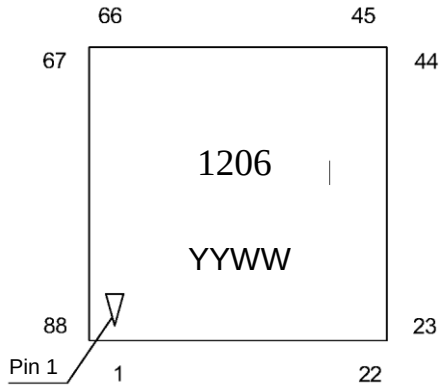
Multifunctional microcontroller
IRL1206FI



YY – year of manufacture
WW – week of manufacture

Key characteristics of the integrated circuit:

- Supply voltage:
  – 3.0 to 3.6 V (with AFE);
  – 1.8 to 3.6 V (without AFE);
- 32-bit RISC-V BM-310S0 architecture with RV32IMCNZX instruction set;
- On-chip flash memory: 512 KB;
- On-chip RAM: 112 KB;
- Integrated circuit junction-to-case thermal resistance:
  − IRL1206FI no more than 3.5 °C/W;
- Integrated circuit weight:
  − IRL1206FI no more than 0.26 g;
  Moisture Sensitivity Level (MSL):
  − IRL1206FI - 2a;
- Operating temperatures range:

Designation	Temperature range, °C
IRL1206FI	-50 to 85

Package type:

- The IRL1206FI are supplied in a 88-lead plastic QFN88 package, 10×10×0.75 (0.4).

Important:

The integrated circuits are moisture sensitive.

The specification is valid when used in conjunction with the document IRL1206FI Errata Notice.

It is recommended **to use the latest version of the bootloader software**:

https://irlabs.in/our-products/microcontrollers-and-microprocessors/ir51206fi

IRL1206_Bootloader_v1.1.0

General description and application scope of the integrated circuit

IRL1206FI (the “integrated circuit”) is a multifunctional microcontroller with embedded Flash program memory, based on the BM-310S0 RISC-V processor core. It is intended for use in a range of electronic equipment, including electrical energy meters in single-phase and three-phase household power networks rated at 230 V/50 Hz and 110 V/60 Hz.

Key characteristics:

Core:

- 32-bit BM-310S0 RISC-V core with RV32IMCNZX instruction set;
- Clock frequency up to 50 MHz for IRL1206FI (CHIP_ID = 217);
- Performance of up to 3.70 CoreMark/MHz when executing from integrated TCM memory;
- Performance of up to 2.15 CoreMark/MHz when executing from Flash memory for IRL1206FI;
- Two-cycle multiplication.
- Instructions for accelerating cryptographic algorithms such as SHA2, AES, etc.;
- Physical memory protection hardware.

Memory:

- On-chip non-volatile Flash program memory with a capacity of 2×256 KB (main area) + 2×8 KB (information area);
- On-chip RAM with a capacity of 112 KB.

Power supply and clock frequency:

- External power supply:

- 3.0 to 3.6 V (with AFE);
- 1.8 to 3.6 V (without AFE);
- On-chip core power supply regulator;
- On-chip power supply monitoring circuits;
- On-chip battery-powered domain;
- On-chip adjustable 8 MHz RC oscillator;
- On-chip adjustable 32 kHz RC oscillator;
- External oscillator, 8 to 16 MHz;
- External oscillator, 32 kHz;
- On-chip PLL clock frequency multiplier for the core.

Low-power mode:

- Battery domain including real-time clock, calendar, three tamper detectors and 512 bytes of RAM.

Analog modules:

- 24-bit $\Delta\sum$ ADC (seven independent channels with a programmable gain amplifier (PGA));
- 10-bit ADC (three external multiplexed channels and an internal temperature sensor channel).

Peripherals:

- Direct memory access controller with Peripheral-to-Memory, Memory-to-Memory transfers;
- Interface controllers: 4 x UART, 3 x SSP, 1 x I2C, 1 x ISO7816;
- Random number generator;
- Protection grid, power supply voltage detectors;
- Up to 55 user-defined IO lines;
- Four 32-bit timer blocks with four event capture channels and a pulse-width modulation (PWM) block;
- Two watchdog timers;
- CRC calculation block with a configurable polynomial.

Debug mode:

- Serial debug interface (JTAG).

Table of contents

1 Introduction 15
2 Block diagram 16
3 Conventional graphical representation 17
4 Pin assignment diagram 18
5 Pin description 19
6 Application and operating guidelines 27
7 Power supply system 28
7.1 Power supply block diagrams 29
7.2 Main power-on and power-off reset circuit 32
8 Memory organization 34
8.1 Integrated circuit base addresses 35
9 Integrated circuit startup modes 36
9.1 Startup modes of IRL1206FI 36
9.1.1 UART+J mode 37
9.1.2 UART bootloader commands 38
10 Program Flash memory controller 42
10.1 General description and structure of Flash memory in IRL1206FI (CHIP_ID = 217) 42
10.1.1 Operation of program Flash memory in normal mode 43
10.1.2 Operation of program Flash memory in programming mode 43
10.2 Manufacturing information, calibration values, and Flash memory protection 48
10.3 Use of calibration values 50
10.4 Program Flash memory controller control registers 50
10.4.1 FLASH_CMD for IRL1206FI (CHIP_ID = 217) 51
10.4.2 FLASH_ADR for IRL1206FI (CHIP_ID = 217) 52
10.4.3 FLASH_DI 53
10.4.4 FLASH_DO 53
10.4.5 FLASH_KEY 53
10.4.6 CHIP_ID_CTRL 54
10.4.7 OTP_KEY 54
11 Instruction set 55
11.1 Wait for Interrupt support 58
12 BM-310S0 processor core 59
12.1 Processor block diagram 60

12.1.1 TCM interface 61
12.1.2 TCM memory block access arbiter 61
12.1.3 AHB I/O interface 61
12.1.4 Interrupt requests 61
12.1.5 JTAG interface 61
12.1.6 LIO crossbar 62
12.1.7 Physical Memory Protection (PMP) 62
13 ADC block for measuring voltages and currents in the electrical power network 67
13.1 Description of the control registers of the seven-channel ADC block 74
13.1.1 ADCUI_CTRL 1 77
13.1.2 ADCUI_CTRL2 81
13.1.3 ADCUI_CTRL3 82
13.1.4 ADCUI_F0CTR 83
13.1.5 ADCUI_F0WC 84
13.1.6 ADCUI_F0WATTP 85
13.1.7 ADCUI_F0WATTN 85
13.1.8 ADCUI_F0VC 85
13.1.9 ADCUI_F0VARP 86
13.1.10 ADCUI_F0VARN 86
13.1.11 ADCUI_F0AC 86
13.1.12 ADCUI_F0VR 87
13.1.13 ADCUI_F0MD0 87
13.1.14 ADCUI_F0MDI 89
13.1.15 ADCUI_F0VPEAK 89
13.1.16 ADCUI_F0IPEAK 89
13.1.17 ADCUI_F0VDAT 90
13.1.18 ADCUI_F0I0DAT 90
13.1.19 ADCUI_F0I3DAT 91
13.1.20 ADCUI_F0VRMS 91
13.1.21 ADCUI_F0VRMS2 91
13.1.22 ADCUI_F0IRMS 92
13.1.23 ADCUI_F0IRMS2 92
13.1.24 ADCUI_F0STAT 92
13.1.25 ADCUI_F0MASK 94
13.1.26 ADCUI_F1CTR 95
13.1.27 ADCUI_F1WC 97
13.1.28 ADCUI_F1WATTP 97
13.1.29 ADCUI_F1WATTN 97
13.1.30 ADCUI_F1VC 98
13.1.31 ADCUI_F1VARP 98
13.1.32 ADCUI_F1VARN 98
13.1.33 ADCUI_F1AC 99
13.1.34 ADCUI_F1VR 99
13.1.35 ADCUI_F1MD0 100
13.1.36 ADCUI_F1MD1 101
13.1.37 ADCUI_F1MD2 101
13.1.38 ADCUI_F1VPEAK 102
13.1.39 ADCUI_F1IPEAK 102

13.1.40 ADCUI_F1VDAT 102
13.1.41 ADCUI_F1IDAT 103
13.1.42 ADCUI_F1VRMS 103
13.1.43 ADCUI_F1VRMS2 104
13.1.44 ADCUI_F1IRMS 104
13.1.45 ADCUI_F1IRMS2 104
13.1.46 ADCUI_F1STAT 105
13.1.47 ADCUI_F1MASK 106
13.1.48 ADCUI_F2CTR 107
13.1.49 ADCUI_F2WC 109
13.1.50 ADCUI_F2WATTP 109
13.1.51 ADCUI_F2WATTN 109
13.1.52 ADCUI_F2VC 110
13.1.53 ADCUI_F2VARP 110
13.1.54 ADCUI_F2VARN 110
13.1.55 ADCUI_F2AC 111
13.1.56 ADCUI_F2VR 111
13.1.57 ADCUI_F2MD0 111
13.1.58 ADCUI_F2MD1 112
13.1.59 ADCUI_F2MD2 113
13.1.60 ADCUI_F2VPEAK 114
13.1.61 ADCUI_F2IPEAK 114
13.1.62 ADCUI_F2VDAT 114
13.1.63 ADCUI_F2IDAT 115
13.1.64 ADCUI_F2VRMS 115
13.1.65 ADCUI_F2VRMS2 115
13.1.66 ADCUI_F2IRMS 116
13.1.67 ADCUI_F2IRMS2 116
13.1.68 ADCUI_F2STAT 116
13.1.69 ADCUI_F2MASK 118
13.1.70 ADCUI_CCAL1 119
13.1.71 ADCUI_CCAL2 119
13.1.72 ADCUI_CCAL3 120
13.1.73 ADCUI_CCAL4 120
13.1.74 ADCUI_F0IRMS_INACTIVE 120
13.1.75 ADCUI_F0WATTP_L 121
13.1.76 ADCUI_F0WATTN_L 121
13.1.77 ADCUI_F0VARP_L 121
13.1.78 ADCUI_F0VARN_L 122
13.1.79 ADCUI_F0VR_L 122
13.1.80 ADCUI_F1WATTP_L 122
13.1.81 ADCUI_F1WATTN_L 123
13.1.82 ADCUI_F1VARP_L 123
13.1.83 ADCUI_F1VARN_L 123
13.1.84 ADCUI_F1VR_L 124
13.1.85 ADCUI_F2WATTP_L 124
13.1.86 ADCUI_F2WATTN_L 124
13.1.87 ADCUI_F2VARP_L 125

13.1.89 ADCUI_F2VARN_L 125
13.1.90 ADCUI_F2VR_L 126
13.1.91 ADCUI_LOAD 126
13.1.92 ADCUI_F0VRMS_TRUE 127
13.1.93 ADCUI_F1VRMS_TRUE 127
13.1.94 ADCUI_F2VRMS_TRUE 128
13.1.95 ADCUI_MSC 129
13.1.96 ADCUI_FILTERCFG 130
13.1.97 ADCUI_CLKPHASE1 131
14 Algorithms for calculating final results and their correspondence to external signals 132
14.1 Typical connection diagram for three-phase electric power metering 139
14.2 Typical connection scheme for single-phase energy metering 141
14.3 Typical connection scheme for single-phase energy metering using a shunt 142
15 Hardware CRC computation block 143
15.1 Description of CRC block control registers 143
15.1.1 CRC_CTRL 143
15.1.2 CRC_STAT 144
15.1.3 CRC_DATAI 145
15.1.4 CRC_VAL 145
15.1.5 CRC_POL 145
16 Clock signals 146
16.1 Internal RC HSI oscillator 147
16.2 Internal RC LSI oscillator 147
16.3 External HSE oscillator 147
16.4 External LSE oscillator 149
16.5 On-chip system clock frequency multiplication block 150
16.6 Clock controller register description 151
16.6.1 CLOCK_STATUS 151
16.6.2 PLL_CONTROL 152
16.6.3 HS_CONTROL 152
16.6.4 CPU_CLOCK 153
16.6.5 PER1_CLOCK 153
16.6.6 ADC_CLOCK 154
16.6.7 RTC_CLOCK 155
16.6.8 PER2_CLOCK 157
16.6.9 DMA_DONE_STICK 158
16.6.10 TIM_CLOCK 158
16.6.11 UART_CLOCK 159
16.6.12 SSP_CLOCK 161
16.6.13 DIV_SYS_TIM 162
17 Battery domain and real-time clock 163
17.1 Real time clock 163
17.1.1 Temperature compensation of the real-time clock SEC_CLK frequency 166
17.2 Register memory and cryptographic keys 169

17.3 Description of the battery domain block registers 169
17.3.1 BKP_MEM (128x32) 170
17.3.2 BKP_WPR 170
17.3.3 BKP_LDO 170
17.3.4 BKP_CLK 172
17.3.5 BKP_RTC_CR 173
17.3.6 BKP_RTC_WUTR 175
17.3.7 BKP_RTC_PREDIV_S 175
17.3.8 BKP_RTC_PRL 176
17.3.9 BKP_RTC_ALRM 176
17.3.10 BKP_RTC_CS 177
17.3.11 BKP_RTC_TR 178
17.3.12 BKP_RTC_DR 179
17.3.13 BKP_RTC_ALRMAR, BKP_RTC_ALRMBR 180
17.3.14 BKP_RTC_TSTR1-BKP_RTC_TSTR3, BKP_RTC_TSDR1-BKP_RTC_TSDR3 181
17.3.15 BKP_RTC_TAFCR 181
17.3.16 BKP_RTC_TMPCAL1 183
17.3.17 BKP_RTC_TMPCAL2 184
17.3.18 BKP_RTC_TMPCAL3 185
18 Input/output ports 186
18.1 Description of I/O port registers 188
18.1.1 PORT_RXTX 189
18.1.2 PORT_OE 189
18.1.3 PORT_FUNC 190
18.1.4 PORT_ANALOG 190
18.1.5 PORT_PULL 191
18.1.6 PORT_PWR 191
18.1.7 PORT_SETTX 192
18.1.8 PORT_CLRTX 192
18.1.9 PORT_RDTX 193
19 Power supply voltage detector 194
19.1 Description of the PVD block registers 195
19.1.1 PVDCS 195
20 General-purpose timers 197
20.1 Main characteristics 197
20.1.1 Block diagram 198
20.2 Basic timer block 199
20.2.1 Timer clock initialization 199
20.2.2 Initialization of the main timer counter 199
20.2.3 Counting modes 200
20.2.4 F_{DTS} clock frequency 202
20.3 Event sources for counting 203
20.3.1 Internal clock signal (TIM_CLKd) 204
20.3.2 Event in another timer (CNT == ARR) 205
20.3.3 External clock signal, “Mode 1”: rising-edge event on the CHyi channel input 206

20.3.4 External clock signal, “Mode 2”: rising- or falling-edge event on the ETR input 208
20.4 Capture mode 209
20.5 PWM mode 211
20.5.1 Reference signal generator REF 211
20.5.2 Dead-time generator 213
20.5.3 Output blocks 214
20.6 Digital filter block 215
20.7 Status flags, interrupts, and DMA requests 217
20.7.1 Status flags 217
20.7.2 Interrupts 217
20.7.3 DMA requests 217
20.8 Examples 218
20.8.1 Basic counter 218
20.8.2 Capture mode 218
20.8.3 PWM mode 219
20.9 Timer block register description 221
20.9.1 CNT 222
20.9.2 PSG 222
20.9.3 ARR 222
20.9.4 CNTRL 224
20.9.5 CHy_CCR 224
20.9.6 CHy_CCR_1 224
20.9.7 CHy_CNTRL0 225
20.9.8 CHy_CNTRL1 227
20.9.9 CHy_CNTRL2 228
20.9.10 CHy_DTG 228
20.9.11 BRKETR_CNTRL 229
20.9.12 STATUS 230
20.9.13 IE 232
20.9.14 DMA_RE 233
21 SAR ADC controller 235
21.1 External channel conversion 236
21.2 Sequential conversion of multiple channels 237
21.3 Conversion with range checking 237
21.4 Temperature sensor 238
21.4.1 Temperature calculation formula 238
21.5 Internal capacitance charge time 238
21.6 ADC internal capacitance charge time and conversion time 239
21.7 Description of the ADC controller block registers 240
21.7.1 ADC1_CFG 241
21.7.2 ADC1_H_LEVEL 243
21.7.3 ADC1_L_LEVEL 243
21.7.4 ADC1_RESULT 243
21.7.5 ADC1_STATUS 244
21.7.6 ADC1_CHSEL 245
21.7.7 ADC_TRIM 245
21.8 SAR_AINx pins 246

22 I2C interface controller 247
22.1 I2C controller configuration 247
22.2 System configuration 248
22.3 I2C protocol 248
22.3.1 START signal 249
22.3.2 Address transfer 249
22.3.3 Data transfer 250
22.3.4 STOP signal 250
22.4 Description of I2C controller registers 250
22.4.1 PRL 251
22.4.2 PRH 251
22.4.3 CTR 251
22.4.4 RXD 252
22.4.5 STA 252
22.4.6 TXD 253
22.4.7 CMD 254
23 SSP controller 255
23.1 Key characteristics of the SSP module 255
23.1.1 Programmable parameters 256
23.1.2 SPI interface characteristics 257
23.1.3 Microwire interface characteristics 257
23.1.4 SSI interface characteristics 257
23.2 SSP module overview 257
23.2.1 Clock generation block 258
23.2.2 Transmit FIFO buffer 258
23.2.3 Receive FIFO buffer 259
23.2.4 Data transmit/receive block 259
23.2.5 Interrupt generation block 259
23.3 Direct Memory Access interface 260
23.4 Transceiver configuration 260
23.5 Transceiver operation permission 260
23.6 Clock frequency signals relationships 260
23.7 Programming the CR0 control register 261
23.8 Programming the CR1 control register 262
23.9 Data clock signal generation 262
23.10 Data frame format 263
23.10.1 Texas Instruments SSI synchronous transfer format 264
23.10.2 Motorola SPI synchronous transfer format 264
23.10.3 Motorola SPI synchronous transfer format, SPO=0, SPH=0 265
23.10.4 Motorola SPI synchronous transfer format, SPO=0, SPH=1 266
23.10.5 Motorola SPI synchronous transfer format, SPO=1, SPH=0 267
23.10.6 Motorola SPI synchronous transfer format, SPO=1, SPH=1 268
23.10.7 National Semiconductor Microwire synchronous transfer format 269
23.11 Examples of module configuration in master and slave modes 272
23.12 Direct Memory Access interface 274
23.13 Interrupts 275
23.13.1 SSPRXINTR 276

23.13.2 SSPTXINTR 276
23.13.3 SSPRORINTR 276
23.13.4 SSPRTINTR 277
23.13.5 SSPRNEINTR 277
23.13.6 SSPTFEINTR 277
23.13.7 SSPTNBSYINTR 277
23.13.8 SSPINTR 277
23.14 Software control of the module 277
23.14.1 General 277
23.14.2 Description of SSP controller registers 278
24 UART controller 286
24.1 Key characteristics of the UART module 286
24.2 Programmable parameters 287
24.3 Differences from the 16C650 UART controller 287
24.4 Functional capabilities 288
24.5 Description of UART operation 290
24.5.1 Transceiver clock generator 290
24.5.2 Transmit FIFO buffer 290
24.5.3 Receive FIFO buffer 291
24.5.4 Transmitter block 291
24.5.5 Receiver block 291
24.5.6 Interrupt generation block 291
24.5.7 Direct Memory Access interface 292
24.5.8 Synchronization block and registers 292
24.6 IrDA SIR codec operation 292
24.6.1 IR transmitter encoder 292
24.6.2 IR receiver decoder 293
24.7 UART operation description 293
24.7.1 Module reset 293
24.7.2 Clock signals 294
24.7.3 Universal asynchronous receiver-transmitter operation 294
24.7.4 Clock division factor 295
24.7.5 Transmission and reception of data 296
24.7.6 Error bits 297
24.7.7 Buffer overrun bit 297
24.7.8 FIFO buffer disable 297
24.7.9 Operation of the IrDA SIR infrared data codec 298
24.8 Modem control lines 300
24.8.1 Hardware flow control 300
24.8.2 RTS-based hardware flow control 301
24.8.3 CTS-based hardware flow control 301
24.9 Direct Memory Access interface 302
24.10 Interrupts 304
24.10.1 UARTMSINTR 305
24.10.2 UARTRXINTR 305
24.10.3 UARTTXINTR 306
24.10.4 UARTRTINTR 306
24.10.5 UARTEINTR 306

24.10.6 UARTINTR 307
24.11 Software control of the module 307
24.12 Description of UART controller registers 308
24.12.1 Data register DR 308
24.12.2 Receive status/error clear register RSR_ECR 310
24.12.3 Flag register FR 311
24.12.4 Low-power IR transfer control register ILPR 311
24.12.5 IBRD integer baud rate division register 312
24.12.6 FBRD fractional baud rate division register 312
24.12.7 LCR_H line control register 314
24.12.8 Control Register CR 316
24.12.9 Interrupt FIFO Level Select Register (IFLS) 318
24.12.10 Interrupt Mask Set/Clear Register IMSC 319
24.12.11 Raw Interrupt Status Register RIS 320
24.12.12 Masked Interrupt Status register MIS 322
24.12.13 Interrupt Clear register ICR 323
24.12.14 Direct Memory Access Control register DMACR 324
24.12.15 Test control register TCR 325
25 Direct Memory Access (DMA) controller 326
25.1 Key features of the DMA controller 326
25.2 Terms and definitions 326
25.3 Functional description 328
25.3.1 DMA channel assignment 328
25.3.2 Block connected to the APB bus 329
25.3.3 Block connected to the AHB bus 329
25.3.4 DMA control block 329
25.3.5 Transfer types 330
25.3.6 Data width of transfers 330
25.3.7 Data protection control 330
25.3.8 Address increment 331
25.4 DMA control 332
25.4.1 Data transfer rules 332
25.4.2 DMA controller operation diagrams 334
25.4.3 DMA arbitration rules 340
25.4.4 Priority 340
25.4.5 DMA cycle types 342
25.5 Channel control data structure 353
25.5.1 Source end pointer 357
25.5.2 Destination end pointer 358
25.5.3 Control bits 358
25.6 Description of DMA controller register 364
25.6.1 DMA status register STATUS 365
25.6.2 DMA configuration register CFG 366
25.6.3 Channel control data base address register CTRL_BASE_PTR 367
25.6.4 Alternate channel control data base address register ALT_CTRL_BASE_PTR 368
25.6.5 Channel wait-on-request status register WAITONREQ_STATUS 368
25.6.6 Channel software request register CHNL_SW_REQUEST 369
25.6.7 Channel use-burst set register CHNL_USEBURST_SET 369

25.6.8 Channel use-burst clear register CHNL_USEBURST_CLR 371
25.6.9 Channel request mask set register CHNL_REQ_MASK_SET 371
25.6.10 Channel request mask clear register CHNL_REQ_MASK_CLR 372
25.6.11 Channel enable set register CHNL_ENABLE_SET 373
25.6.12 Channel enable clear register CHNL_ENABLE_CLR 374
25.6.13 Channel primary/alternate control data structure select register CHNL_PRI_ALT_SET 374
25.6.14 Channel primary/alternate control data structure clear register CHNL_PRI_ALT_CLR 376
25.6.15 Channel priority set register CHNL_PRIORITY_SET 377
25.6.16 Channel priority clear register CHNL_PRIORITY_CLR 377
25.6.17 Error flag clear register ERR_CLR 378
26 Interrupts 379
27 Core local interrupt controller CLIC 380
27.1 Interrupt preemption 380
27.2 Interaction of the CLIC controller with other local interrupts 380
27.3 Description of CLIC controller registers 381
27.3.1 cliccfg configuration register 381
27.3.2 Information register clicinfo 384
27.3.3 Interrupt pending register clicintip 385
27.3.4 Interrupt enable register clicintie 386
27.3.5 Interrupt attribute register clicintattr 387
27.3.6 Interrupt input control register clicintctl 388
27.4 Control and status registers (CSR) 389
27.4.1 mtvec, utvec registers 390
27.4.2 mtvt, utvt registers 391
27.4.3 mnxti, unxti registers 391
27.4.4 mintstatus, uintstatus registers 392
27.4.5 mintthresh, uintthresh registers 393
27.4.6 mclicbase register 394
27.4.7 mscratchcswl, uscratchcswl registers 395
27.5 Local interrupt map 396
28 Debug subsystem 398
28.1 DTM (Debug Transport Module) block 398
28.1.1 DTM registers 399
28.2 DM (Debug Module) block 401
28.2.1 DM registers 401
28.3 Connecting a JTAG adapter to the chip 403
29 Watchdog timers 405
29.1 Watchdog timer block registers 405
29.1.1 IWDG watchdog timer block 405
29.1.2 WWDG watchdog timer registers 410
30 Random number generator RANDOM 414
30.1 Purpose and properties 414
30.2 Module registers 414
30.2.1 Status and control register STAT_CTRL 415
30.2.2 Interrupt control register INT_CTRL 415
30.2.3 Generator clock divider register CLK_DIV 416

30.2.4 Enable-pause register PAUSE 416
30.2.5 Random value register OUTPUT 416
30.2.6 Pause counter register PAUSE_CNT 417
30.2.7 Random-collection register TEMP 417
31 USART controller (ISO7816) 418
31.1 Introduction 418
31.2 USART module features 418
31.3 USART functional description 419
31.3.1 Description of USART characters 421
31.3.2 Transmitter 422
31.3.3 Receiver 425
31.3.4 Fractional baud rate generator 430
31.3.5 USART receiver clock-error tolerance 433
31.3.6 Multiprocessor communication 434
31.3.7 Parity check 436
31.3.8 Synchronous USART mode 437
31.3.9 Single-wire half-duplex communication 439
31.3.10 Smartcard mode 440
31.3.11 Hardware flow control 442
31.4 USART interrupts 444
31.5 USART registers 445
31.5.1 USART_SR 445
31.5.2 USART_DR 448
31.5.3 USART_BRR 449
31.5.4 USART_CR1 450
31.5.5 USART_CR2 453
31.5.6 USART_CR3 454
31.5.7 USART_GTPR 456
32 Electrical parameters of the integrated circuits 457
33 Maximum allowable parameters and limits 459
34 Reference parameters 461
34.1 Dynamic characteristics of the ΔΣ ADC 463
34.2 Performance in tests 463
34.3 IC pin characteristics 464
35 Typical dependencies 466
36 Integrated circuit outline 468
37 Ordering information 469

1 Introduction

The integrated circuit belongs to the family of microcontrollers with on-chip Flash program memory, and is based on the BM-310S0 RISC-V processor core. It operates at clock frequencies of up to 48 MHz and includes 512+16 KB of Flash memory and 112 KB of RAM. The integrated circuit includes advanced peripherals required to implement an electric energy meter for single-phase and three-phase power networks. The peripheral set includes seven channels for a three-phase network (or 3 channels for a single-phase network), each with an independent 24-bit $\Delta\sum$ ADC. Each ADC channel includes an amplifier with phase adjustment (phase correction of at least 0.1) and a hardware unit for calculating the root-mean-square (RMS) value of the signal. ADC channels can be enabled or disabled independently and each has a dedicated direct memory access channel. In addition, a 10-bit successive-approximation ADC is provided for monitoring the voltage of the main or battery domain, measuring temperature, or capturing an external signal. The integrated circuit includes 5 x UART, 3 x SSP, and 1 x I2C interfaces. Cryptographic protection is implemented through a xgost instruction set, a random number generator, and a protection grid. The integrated circuit includes 4 x 32-bit timers with 4 x capture and PWM channels, featuring dead-time insertion and hardware interlock functions. It also contains a system 64-bit timer, and two watchdog timers. One of the watchdog timers operates from the core supply voltage, while the other operates from the main power supply.

On-chip RC oscillators HSI (8 MHz) and LSI (32 kHz), external oscillators HSE (8-16 MHz) and LSE (32 kHz), and a PLL core clock multiplier allow flexible configuration of the IC operating speed.

The system bus architecture, through adjustment of peripheral block clock frequencies, makes it possible to reduce overall system power consumption. The DMA controller enables accelerated data transfer between RAM and peripherals without involvement of the processor core.

The on-chip LDO regulator for powering the digital domain does not require additional external components. Operation of the IC requires only a single external supply voltage in the range of 1.8 to 3.6 V. The IC also implements a battery domain that operates from an external battery in the absence of the main power supply. Cryptographic keys can be stored in the battery domain, and the real-time clock continues to operate. On-chip power supply voltage detectors can monitor the level of the external main supply voltage and the battery supply voltage. Hardware brown-out reset circuits minimize the probability of IC malfunction when the supply voltage exceeds permissible limits. In the event of a failure, the on-chip watchdog timers allow the device to recover operation.

2 Block diagram

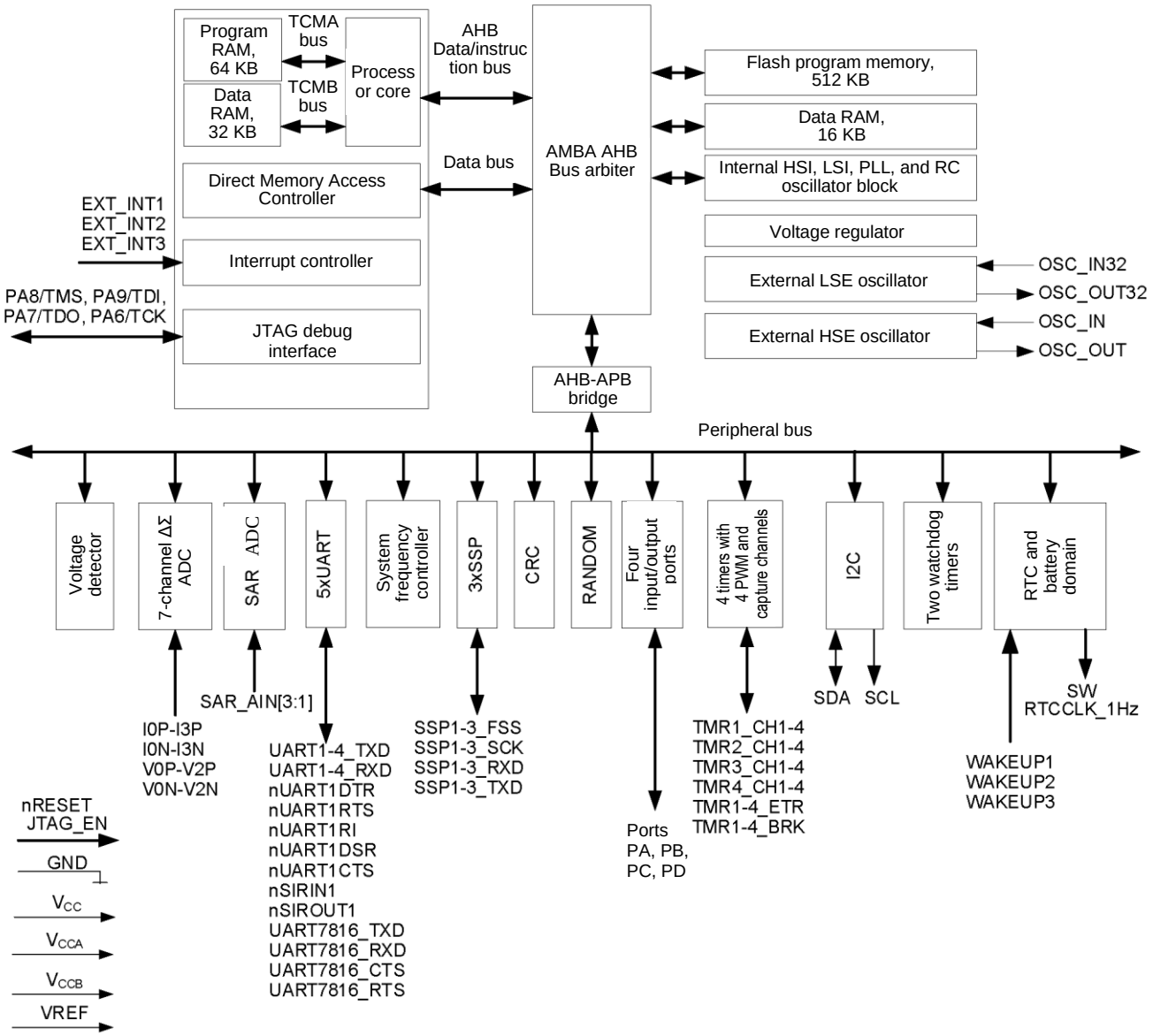


Figure 1 – Block diagram of IRL1206FI

3 Conventional graphical representation

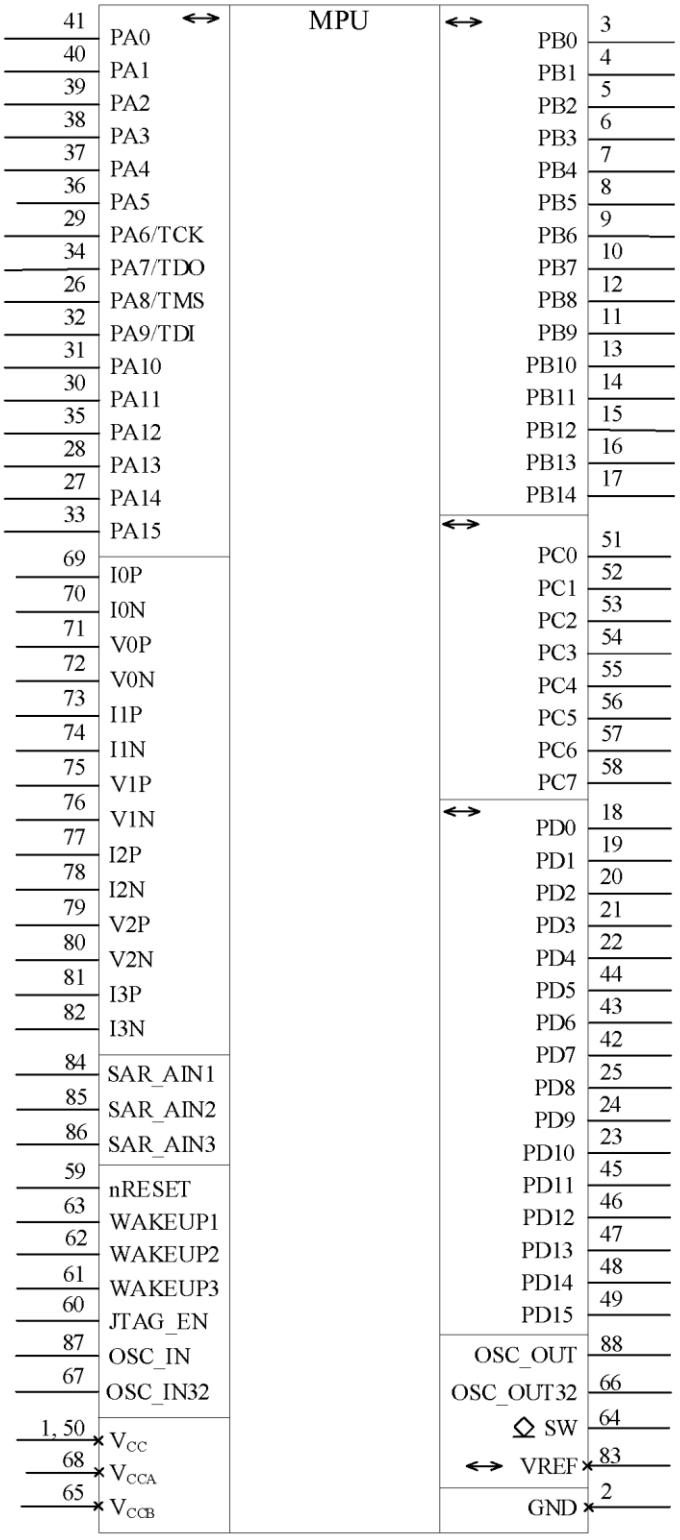


Figure 2 – Conventional graphical representation of IRL1206FI

4 Pin assignment diagram

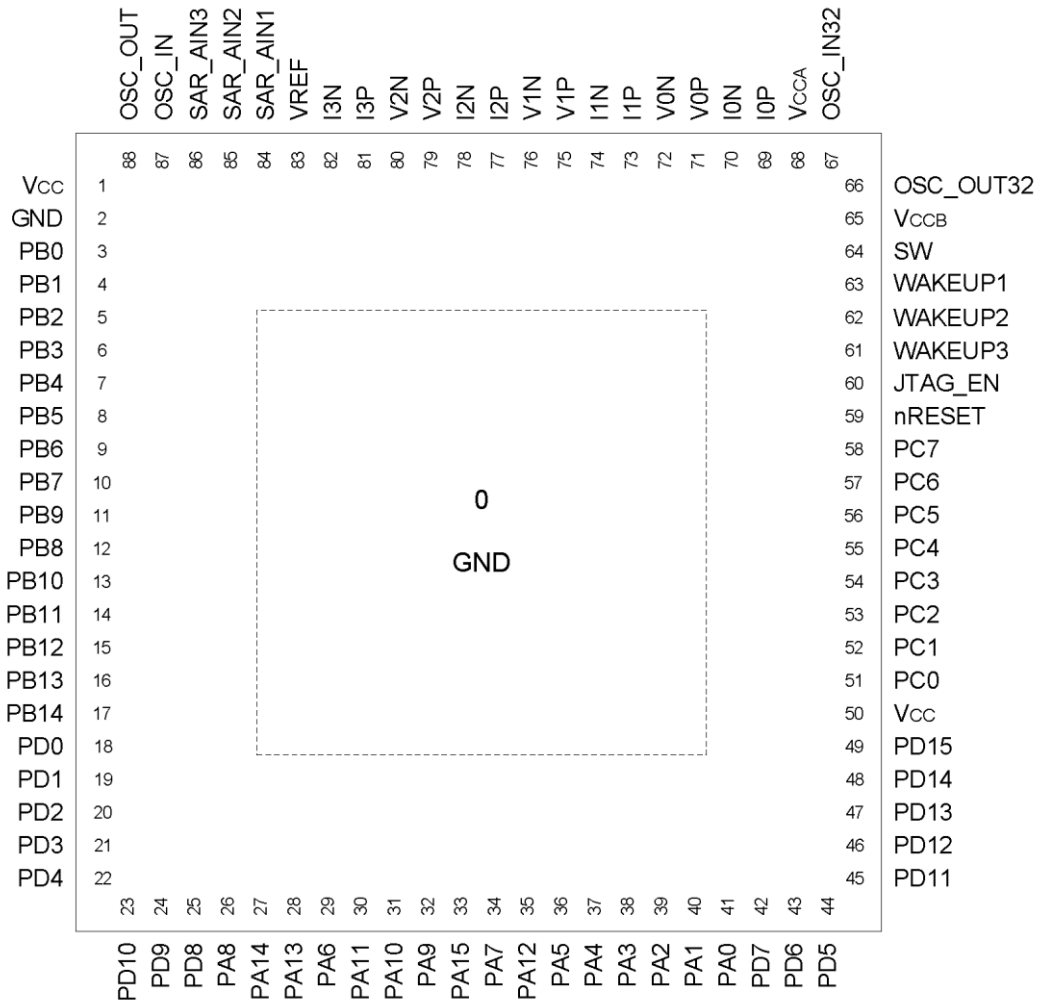


Figure 3 – Pin assignment diagram for IRL1206FI

5 Pin description

Table 1 – Pin description of IRL1206FI

Pin No.	Pin designation	Pin function
1	V_{CC}	Power supply
2	GND	Common
3	PB0	I/O port B bit 0
4	PB1	I/O port B bit 1
5	PB2	I/O port B bit 2
6	PB3	I/O port B bit 3
7	PB4	I/O port B bit 4
8	PB5	I/O port B bit 5
9	PB6	I/O port B bit 6
10	PB7	I/O port B bit 7
11	PB9	I/O port B bit 9
12	PB8	I/O port B bit 8
13	PB10	I/O port B bit 10
14	PB11	I/O port B bit 11
15	PB12	I/O port B bit 12
16	PB13	I/O port B bit 13
17	PB14	I/O port B bit 14
18	PD0	I/O port D bit 0
19	PD1	I/O port D bit 1
20	PD2	I/O port D bit 2
21	PD3	I/O port D bit 3
22	PD4	I/O port D bit 4
23	PD10	I/O port D bit 10
24	PD9	I/O port D bit 9
25	PD8	I/O port D bit 8
26	PA8/TMS	I/O Port A Bit 8/JTAG Control Input
27	PA14	I/O port A bit 14
28	PA13	I/O port A bit 13
29	PA6/TCK	I/O Port A Bit 6/JTAG Clock Input
30	PA11	I/O port A bit 11
31	PA10	I/O port A bit 10
32	PA9/TDI	I/O Port A Bit 9/JTAG Data Input
33	PA15	I/O port A bit 15
34	PA7/TDO	I/O Port A Bit 7/JTAG Data Output
35	PA12	I/O port A bit 12
36	PA5	I/O port A bit 5
37	PA4	I/O port A bit 4
38	PA3	I/O port A bit 3
39	PA2	I/O port A bit 2
40	PA1	I/O port A bit 1
41	PA0	I/O port A bit 0
42	PD7	I/O port D bit 7
43	PD6	I/O port D bit 6

Pin No.	Pin designation	Pin function
44	PD5	I/O port D bit 5
45	PD11	I/O port D bit 11
46	PD12	I/O port D bit 12
47	PD13	I/O port D bit 13
48	PD14	I/O port D-bit 14
49	PD15	I/O port D bit 15
50	V_{CC}	Power supply
51	PC0	I/O port C bit 0
52	PC1	I/O port C bit 1
53	PC2	I/O port C bit 2
54	PC3	I/O port C bit 3
55	PC4	I/O port C bit 4
56	PC5	I/O port C bit 5
57	PC6	I/O port C bit 6
58	PC7	I/O port C bit 7
59	nRESET	External reset signal
60	JTAG_EN	Enable input for the debug TAP interface on pins PB[9:6]. Connect to the Common bus in functional mode
61	WAKEUP3	Tamper input 3
62	WAKEUP2	Tamper input 2
63	WAKEUP1	Tamper input 1
64	SW	Output for controlling external power switching from U_{CCB} to U_{CC}
65	V_{CCB}	Battery supply
66	OSC_OUT32	LSE oscillator output
67	OSC_IN32	LSE oscillator input
68	V_{CCA}	Analog supply for the ΔΣ ADC and SAR ADC
69	I0P	Current channel 0 input, non-inverting
70	I0N	Current channel 0 input, inverting
71	V0P	Voltage channel 0 input, non-inverting
72	V0N	Voltage channel 0 input, inverting
73	I1P	Current channel 1 input, non-inverting
74	I1N	Current channel 1 input, inverting
75	V1P	Voltage channel 1 input, non-inverting
76	V1N	Voltage channel 1 input, inverting
77	I2P	Current channel 2 input, non-inverting
78	I2N	Current channel 2 input, inverting
79	V2P	Voltage channel 2 input, non-inverting
80	V2N	Voltage channel 2 input, inverting
81	I3P	Current channel 3 input, non-inverting
82	I3N	Current channel 3 input, inverting
83	VREF	2.4 V reference voltage input/output
84	SAR_AIN1	SAR ADC input 1
85	SAR_AIN2	SAR ADC input 2
86	SAR_AIN3	SAR ADC input 3
87	OSC_IN	HSE oscillator input
88	OSC_OUT	HSE oscillator output

Pin No.	Pin designation	Pin function
0 (backside package metallization)	GND	Common
Note: the functions of the pins of ports A, B, C, D are provided in Table 2		

Table 2 – Functions of ports A, B, C, and D pins of IRL1206FI

Pin No.	Pin designation	Pin application and function		
		Pin function type	Pin function designation	Pin function
Port A				
41	PA0	I/O port A bit 0		
		Main	TMR1_CH1	Timer 1 channel 1 Input/output
		Alternate	SSP2_FSS	SPI 2 Enable Input/Output
		Overridden	–	–
40	PA1	I/O port A bit 1		
		Main	TMR1_CH1N	Inverted output of Timer 1 channel 1
		Alternate	SSP2_CLK	SPI 2 clock signal input/output
		Overridden	–	–
39	PA2	I/O port A bit 2		
		Main	TMR1_CH2	Timer 1 channel 2 Input/output
		Alternate	SSP2_RXD	SPI 2 Input Data
		Overridden	–	–
38	PA3	I/O port A bit 3		
		Main	TMR1_CH2N	Inverted output of Timer 1 channel 2
		Alternate	SSP2_TXD	SPI 2 Output Data
		Overridden	–	–
37	PA4	I/O port A bit 4		
		Main	TMR1_CH3	Timer 1 channel 3 Input/output
		Alternate	–	–
		Overridden	–	–
36	PA5	I/O port A bit 5		
		Main	TMR1_CH3N	Inverted output of Timer 1 channel 3
		Alternate	–	–
		Overridden	–	–
29	PA6/TCK	I/O Port A Bit 6/JTAG Clock Input		
		Main	TMR1_CH4	Timer 1 channel 4 Input/output
		Alternate	–	–
		Overridden	–	–

Pin No.	Pin designation	Pin application and function		
		Pin function type	Pin function designation	Pin function
34	PA7/TDO	I/O Port A Bit 7/JTAG Data Output		
		Main	TMR1_CH4N	Inverted output of Timer 1 channel 4
		Alternate	–	–
		Overridden	–	–
26	PA8/TMS	I/O Port A Bit 8/JTAG Control Input		
		Main	TMR1_ETR	Timer 1 External Event Input
		Alternate	–	–
		Overridden	–	–
32	PA9/TDI	I/O Port A Bit 9/JTAG Data Input		
		Main	TMR1_BRK	Timer 1 External Event Input
		Alternate	–	–
		Overridden	–	–
31	PA10	I/O port A bit 10		
		Main	EXT_INT1	Interrupt input 1
		Alternate	UART3_RXD	UART3 Input Data
		Overridden	TMR4_CH1	Timer 4 channel 1 Input/output
30	PA11	I/O port A bit 11		
		Main	TMR2_CH4N	Inverted output of Timer 2 channel 4
		Alternate	UART3_TXD	UART3 output data
		Overridden	TMR4_CH1N	Inverted output of Timer 4 channel 1
35	PA12	I/O port A bit 12		
		Main	SSP1_FSS	SPI 1 Enable Input/Output
		Alternate	–	–
		Overridden	TMR4_CH2	Timer 4 channel 2 Input/output
28	PA13	I/O port A bit 13		
		Main	SSP1_CLK	SPI 1 clock signal input/output
		Alternate	–	–
		Overridden	TMR4_CH2N	Inverted output of Timer 4 channel 2
27	PA14	I/O port A bit 14		
		Main	SSP1_RXD	SPI 1 Input Data
		Alternate	–	–
		Overridden	TMR4_CH3	Timer 4 channel 3 Input/output
33	PA15	I/O port A bit 15		
		Main	SSP1_TXD	SPI 1 Output Data
		Alternate	–	–
		Overridden	TMR4_CH3N	Inverted output of Timer 4 channel 3

Pin No.	Pin designation	Pin application and function		
		Pin function type	Pin function designation	Pin function
Port B				
3	PB0	I/O port B bit 0		
		Main	UART1_TXD	UART1 output data
		Alternate	UART7816_TXD	UART7816 output data
		Overridden	TMR3_CH1	Timer 3 channel 1 Input/output
4	PB1	I/O port B bit 1		
		Main	UART1_RXD	UART1 Input Data
		Alternate	UART7816_RXD	UART7816 Input Data
		Overridden	TMR3_CH1N	Inverted output of Timer 3 channel 1
5	PB2	I/O port B bit 2		
		Main	nSIROUT1	IRDA UART1 Input Data
		Alternate	UART7816_CLK	UART7816 clock signal input
		Overridden	TMR3_CH2	Timer 3 channel 2 Input/output
6	PB3	I/O port B bit 3		
		Main	nSIRIN1	IRDA UART1 Input Data
		Alternate	UART7816_CTS	UART7816 control input
		Overridden	TMR3_CH2N	Inverted output of Timer 3 channel 2
7	PB4	I/O port B bit 4		
		Main	nUART1DTR	UART1 control output
		Alternate	–	–
		Overridden	TMR3_CH3	Timer 3 channel 3 Input/output
8	PB5	I/O port B bit 5		
		Main	nUART1RTS	UART1 control output
		Alternate	UART7816_RTS	UART7816 control output
		Overridden	TMR3_CH3N	Inverted output of Timer 3 channel 3
9	PB6	I/O port B bit 6		
		Main	nUART1RI	UART1 control output
		Alternate	EXT_INT2	Interrupt input 2
		Overridden	TMR3_CH4	Timer 3 channel 4 Input/output
10	PB7	I/O port B bit 7		
		Main	nUART1DCD	UART1 control output
		Alternate	EXT_INT3	Interrupt input 3
		Overridden	TMR3_CH4N	Inverted output of Timer 3 channel 4
12	PB8	I/O port B bit 8		
		Main	nUART1DSR	UART1 control output
		Alternate	TMR2_ETR	Timer 2 External Event Input
		Overridden	TMR3_ETR	Timer 3 External Event Input

Pin No.	Pin designation	Pin application and function		
		Pin function type	Pin function designation	Pin function
11	PB9	I/O port B bit 9		
		Main	nUART1CTS	UART1 control output
		Alternate	TMR2_BRK	Timer 2 External Event Input
		Overridden	TMR3_BRK	Timer 3 External Event Input
13	PB10	I/O port B bit 10		
		Main	TMR2_CH2	Timer 2 channel 2 Input/output
		Alternate	UART4_RXD	UART4 Input Data
		Overridden	–	–
14	PB11	I/O port B bit 11		
		Main	TMR2_CH2N	Inverted output of Timer 2 channel 2
		Alternate	UART4_TXD	UART4 output data
		Overridden	–	–
15	PB12	I/O port B bit 12		
		Main	TMR2_CH3	Timer 2 channel 3 Input/output
		Alternate	–	–
		Overridden	–	–
16	PB13	I/O port B bit 13		
		Main	TMR2_CH3N	Inverted output of Timer 2 channel 3
		Alternate	UART2_TXD	UART2 output data
		Overridden	–	–
17	PB14	I/O port B bit 14		
		Main	TMR2_CH4	Timer 2 channel 4 Input/output
		Alternate	UART2_RXD	UART2 output data
		Overridden	–	–
Port C				
51	PC0	I/O port C bit 0		
		Main	TMR4_ETR	Timer 4 External Event Input
		Alternate	–	–
		Overridden	TMR4_CH4	Timer 4 channel 4 Input/output
52	PC1	I/O port C bit 1		
		Main	TMR4_BRK	Timer 4 External Event Input
		Alternate	–	–
		Overridden	TMR4_CH4N	Inverted output of Timer 4 channel 4
53	PC2	I/O port C bit 2		
		Main	TMR2_CH1	Timer 1 channel 2 Input/output
		Alternate	SSP3_FSS	SPI 3 Enable Input/Output
		Overridden	SCL	I2C clock signal input/output
54	PC3	I/O port C bit 3		
		Main	TMR2_CH1N	Inverted output of Timer 2 channel 1
		Alternate	SSP3_CLK	SPI 3 clock signal input/output
		Overridden	SDA	I2C data input/output

Pin No.	Pin designation	Pin application and function		
		Pin function type	Pin function designation	Pin function
55	PC4	I/O port C bit 4		
		Main	EXT_INT2	Interrupt input 2
		Alternate	SSP3_RXD	SPI 3 Input Data
		Overridden	–	–
56	PC5	I/O port C bit 5		
		Main	EXT_INT3	Interrupt input 3
		Alternate	SSP3_TXD	SPI 3 Output Data
		Overridden	–	–
57	PC6	I/O port C bit 6		
		Main	TMR2_ETR	Timer 2 External Event Input
		Alternate	SCL	I2C clock signal input/output
		Overridden	–	–
58	PC7	I/O port C bit 7		
		Main	TMR2_BRK	Timer 2 External Event Input
		Alternate	SDA	I2C data input/output
		Overridden	–	–
Port D				
18	PD0	I/O port D bit 0		
		Main	SSP2_FSS	SPI 2 Enable Input/Output
		Alternate	SD_V0_EXT	Data in channel V0 SINC3
		Overridden	–	–
19	PD1	I/O port D bit 1		
		Main	SSP2_CLK	SPI 2 clock signal input/output
		Alternate	SD_V1_EXT	Data in SINC3 channel V1
		Overridden	–	–
20	PD2	I/O port D bit 2		
		Main	SSP2_RXD	SPI 2 Input Data
		Alternate	SD_V2_EXT	Data in SINC3 channel V2
		Overridden	–	–
21	PD3	I/O port D bit 3		
		Main	SSP2_TXD	SPI 2 Output Data
		Alternate	–	–
		Overridden	–	–
22	PD4	I/O port D bit 4		
		Main	UART2_TXD	UART2 output data
		Alternate	–	–
		Overridden	–	–
44	PD5	I/O port D bit 5		
		Main	UART2_RXD	UART2 output data
		Alternate	–	–
		Overridden	–	–

Pin No.	Pin designation	Pin application and function		
		Pin function type	Pin function designation	Pin function
43	PD6	I/O port D bit 6		
		Main	UART3_RXD	UART3 Input Data
		Alternate	–	–
		Overridden	–	–
42	PD7	I/O port D bit 7		
		Main	UART3_TXD	UART3 output data
		Alternate	–	–
		Overridden	–	–
25	PD8	I/O port D bit 8		
		Main	UART4_RXD	UART4 Input Data
		Alternate	–	–
		Overridden	–	–
24	PD9	I/O port D bit 9		
		Main	UART4_TXD	UART4 output data
		Alternate	–	–
		Overridden	–	–
23	PD10	I/O port D bit 10		
		Main	UART7816_TXD	UART7816 output data
		Alternate	SD_I0_EXT	Data in SINC3, channel I0
		Overridden	–	–
45	PD11	I/O port D bit 11		
		Main	UART7816_RXD	UART7816 Input Data
		Alternate	SD_I1_EXT	Data in SINC3, channel I1
		Overridden	–	–
46	PD12	I/O port D bit 12		
		Main	UART7816_CLK	UART7816 clock output
		Alternate	SD_I2_EXT	Data in SINC3, channel I2
		Overridden	–	–
47	PD13	I/O port D bit 13		
		Main	UART7816_CTS	UART7816 control input
		Alternate	SD_I3_EXT	Data in SINC3, channel I3
		Overridden	–	–
48	PD14	I/O port D-bit 14		
		Main	UART7816_RTS	UART7816 control output
		Alternate	CLK_SD_EXT	Clock output for external delta-sigma modulators
		Overridden	–	–
49	PD15	I/O port D bit 15		
		Main	EXT_INT1	Interrupt input 1
		Alternate	RTCCLK_1Hz	Output of the SEC_CLK clock signal generated by the real-time clock
		Overridden	–	–

6 Application and operating guidelines

The integrated circuit leads provide solderability at a temperature of 220 °C.

The integrated circuits withstand thermal exposure at a soldering temperature of (250 ± 5) °C for no longer than (10 ± 1) s.

Application of any electrical signals (including power supply and common) to integrated circuit outputs that are not used in accordance with Tables 1 and 3 is prohibited.

Unused input pins of the integrated circuits shall be connected to the power supply bus or to the common bus.

The sequence for applying and removing supply voltage and input signals to the integrated circuits is as follows:

- Application (power-up of the integrated circuits): common, supply voltage, input signals, or simultaneously;
- Removal (power-down of the integrated circuits): simultaneously or in the reverse order.

7 Power supply system

The integrated circuit includes several types of power supply pins:

V_{CC} – main power supply of the integrated circuit, supplies the user pins, the on-chip voltage regulator, the PLL, and the oscillators;

V_{CCB} – battery-domain supply, used in the absence of the main supply U_{CC} to power the battery domain and the LSE oscillator. Automatic switching from the main supply to the battery supply occurs when the U_{CC} level drops below ~1.5 V. Automatic switching from the battery supply back to the main supply occurs approximately 2 ms after the U_{CC} level exceeds 1.6 V. If battery power is not required in the system, the V_{CCB} pin shall be connected to V_{CC};

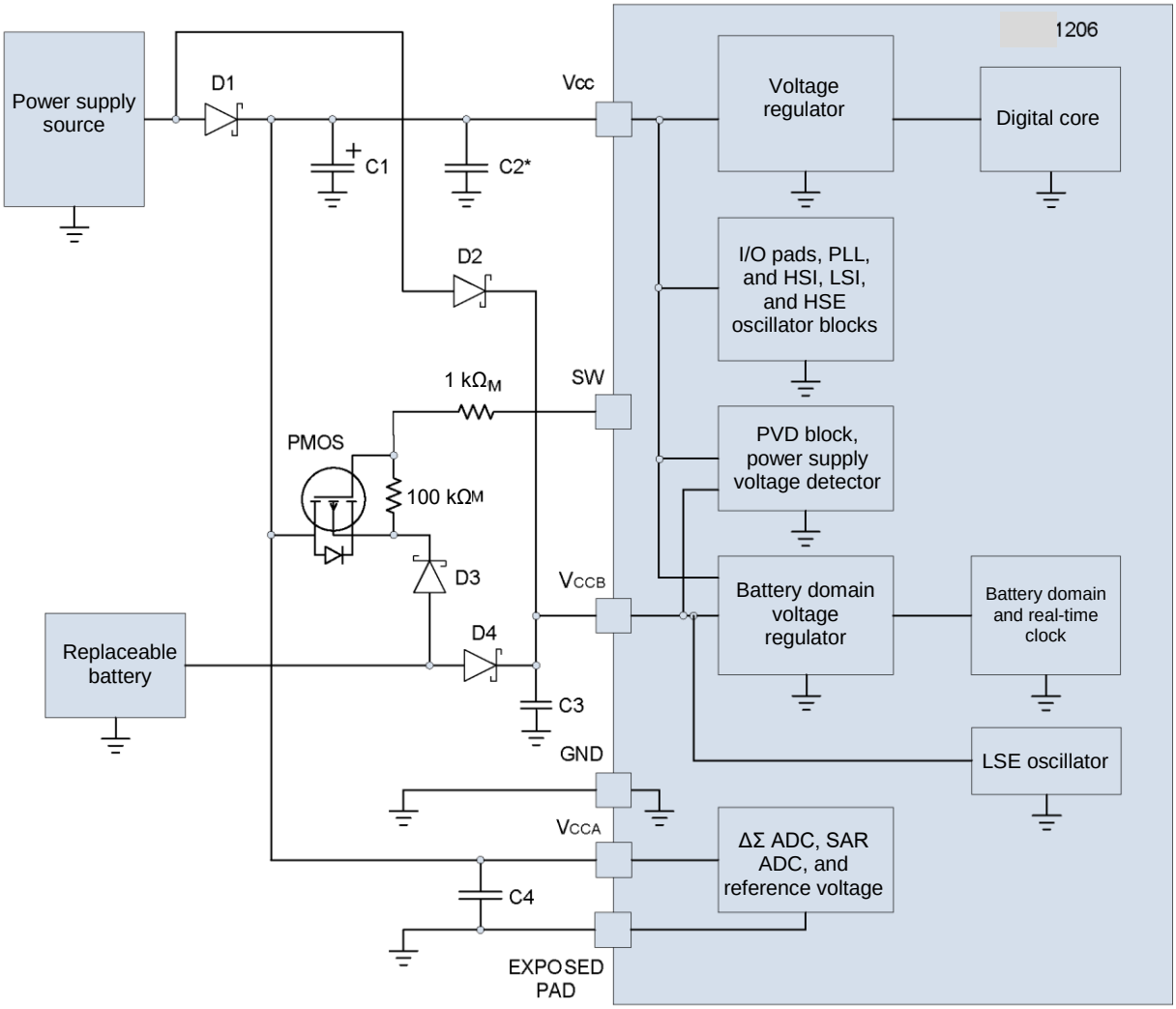
V_{CCA} – the supply for the sigma-delta ADC analog blocks, the successive-approximation ADC, and the reference voltage generator is routed to separate pins to reduce interference generated by the operation of other blocks. Voltage from the same source as V_{CC} shall be applied to these pins, however, noise-reduction measures shall be implemented on the printed circuit board. For correct operation of the ADC and the $\Delta\sum$ ADC, the supply voltage U_{CCA} shall be within the range of 3.0 to 3.6 V. If the supply voltage U_{CCA} is within the range of 1.8 to 3.0 V, correct operation of the ADC and the $\Delta\sum$ ADC is not guaranteed.

GND – main power ground;

EXPOSED PAD – ground for the analog supply U_{CCA} and the digital supply U_{CC}. These pins shall be connected to GND.

The integrated circuit includes on-chip power supply voltage detectors PVD and PVDB; for details, see section 19 “Power supply voltage detector”.

7.1. **Power supply block diagrams**



C1 – capacitor with a capacitance of 22 µF;

C2 – capacitors with a capacitance of 1 µF;

C3 – capacitor with a capacitance of 10 µF;

C4 – capacitors with a capacitance of 0.1 µF;

D1 – D4 – Schottky diodes. It is recommended to select Schottky diodes with the lowest possible voltage drop;

PMOS – P-channel field-effect transistor. It is recommended to select a device with the lowest possible on-state resistance, for example, IRLML6401.

* Capacitors shall be installed close to each power supply pin.

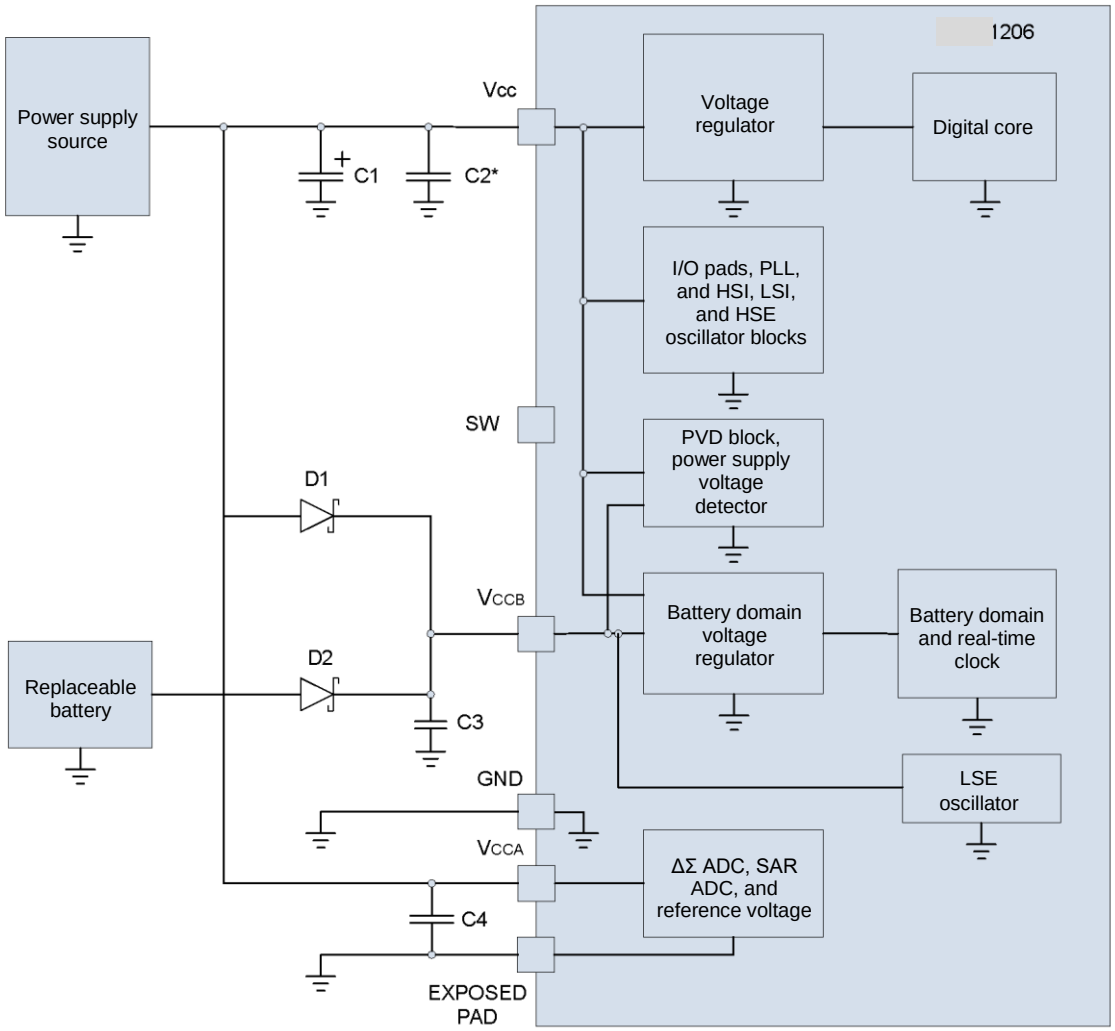
Figure 4 – Power supply block diagram of IRL1206FI

Diodes D2 and D4, in the event of battery power being disconnected (for example, during battery replacement or if the battery is discharged), allow the integrated circuit to remain fully operational when only the U_{CC} supply is present. In addition, if the voltage of the main power source exceeds the battery voltage, this circuit makes it possible to reduce battery consumption, thereby extending battery operating time. In this case, diode D4 protects the battery against reverse current.

Diode D4 and capacitor C3, when the battery backup power function is used (the switch controlled by the SW pin), protect the battery supply U_{CCB} from a voltage drop at the

moment when battery power is applied to the main supply (the switch is closed). When the switch is closed, a short-term inrush current occurs to charge capacitors C1, C2, and others. If the battery is unable to provide the required current, a brief supply voltage sag will occur, and without components D4 and C3, a reset of the battery-domain bit values may occur.

Diode D3 serves to protect the battery from reverse current. When the entire integrated circuit operates from the battery via the "emergency switch" based on the PMOS transistor, at the moment the main power supply is restored and before the "emergency switch" is turned off, reverse current will flow into the battery unless D3 is present. In some cases, installation of the optional diode D1 may also be required. It prevents current from flowing from the battery into the powered-down main supply while the "emergency switch" remains closed.



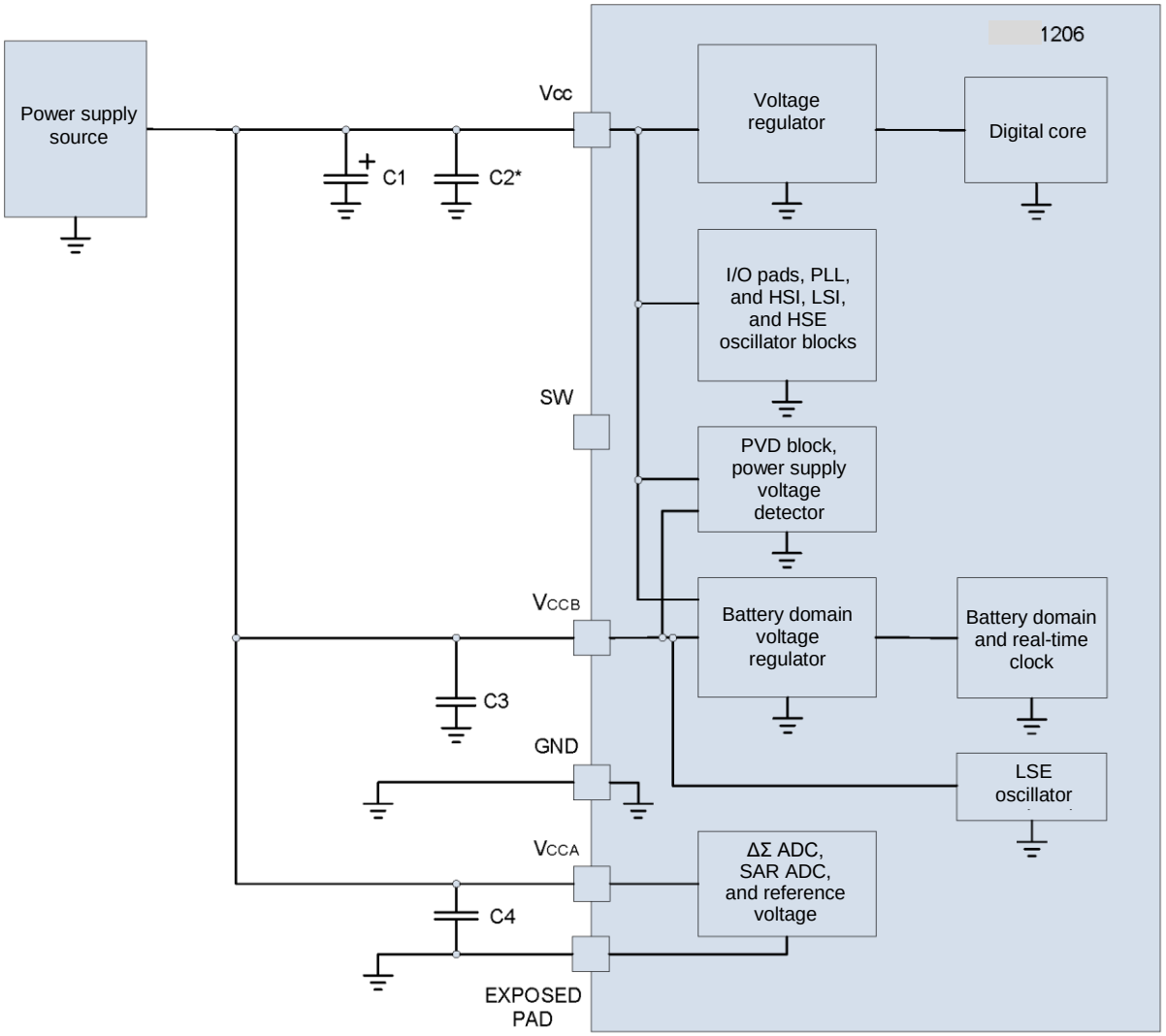
C1 – capacitor with a capacitance of 22 µF;
C2 – capacitors with a capacitance of 1 µF;
C3 – capacitor with a capacitance of 10 µF;
C4 – capacitors with a capacitance of 0.1 µF;
D1, D2 – Schottky diodes. It is recommended to select Schottky diodes with the lowest possible voltage drop.
* Capacitors shall be installed close to each power supply pin.

Figure 5 – Power supply block diagram of IRL1206FI without a switch between U_{CC} and U_{CCB}

If emergency power is not required in the system, the emergency power switch may be omitted, and the SW output may remain unconnected.

Diodes D1 and D2 are optional components. In the event the battery is disconnected (for example, during replacement or if the battery is discharged), they allow the integrated circuit to remain fully operational when only the U_{CC} supply is present. In addition, if the voltage of the main power source exceeds the battery voltage, this circuit makes it possible to reduce battery consumption, thereby extending battery operating time. In this case, diode

D2 protects the battery from reverse current.



C1 – capacitor with a capacitance of 22 µF;

C2 – capacitors with a capacitance of 1 µF;

C3 – capacitor with a capacitance of 0.1 µF;

C4 – capacitor with a capacitance of 0.1 µF;

* Capacitors shall be installed close to each power supply pin.

Figure 6 – Power supply block diagram of IRL1206FI without a battery

Note: When there is no battery supply, the U_{CCB} pin shall be connected to U_{CC}.

7.2. Main power-on and power-off reset circuit

When power is applied, an internal Power-On Reset (POR) signal is generated for the digital domain, the U_{CC} supply voltage begins to rise, and while it remains below 1.6 V, the POR signal is held active; once the voltage exceeds this level, the POR signal remains active for ~2 ms to ensure that the supply has fully stabilized. After this, the POR signal is deasserted, and the circuit can begin operation.

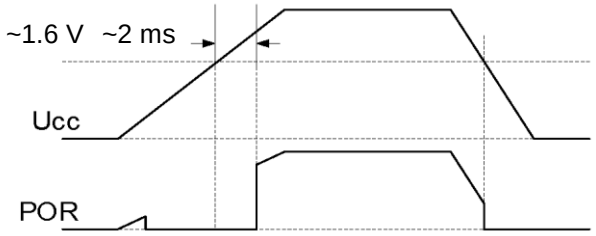


Figure 7 – Reset signal generation diagram of the integrated circuit

When the supply voltage U_{CC} drops below 1.5 V, the POR signal is generated without delay.

The POR signal also serves to switch the battery domain power supply between U_{CCB} and U_{CC}.

When the main supply voltage U_{CC} is applied, the on-chip voltage regulator is automatically enabled to generate the supply voltage for the digital core.

Initial IC reset can be triggered by an external nRESET signal, internal watchdog reset signals, or the internal ndmreset[1] signal generated by the debugger. The nRESET signal is generated by a dedicated reset circuit that includes a glitch filter and a monostable multivibrator to extend the duration of the signal.

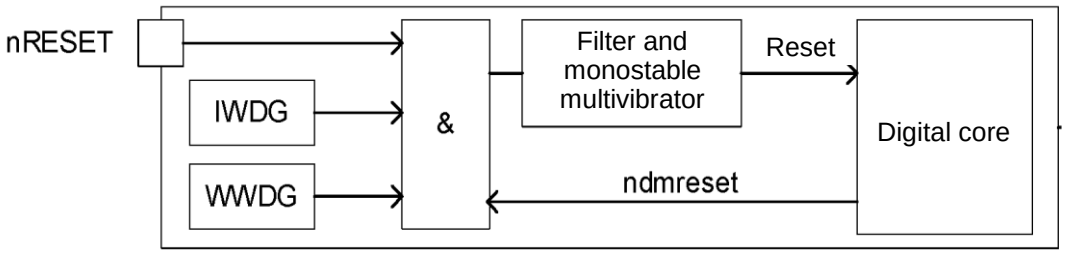


Figure 8 – Reset generation block diagram

When reset pulses shorter than 10 ns are applied to the nRESET input, they are filtered out and do not trigger a processor reset. If the pulse duration exceeds 200 ns, a reset signal is generated. In this case, the duration of the generated reset signal will be no less than 20 μs.

[1] ndmreset (non-debug module reset) – reset signal from the debug subsystem; see the RISC-V External Debug Support Version 0.13 specification.

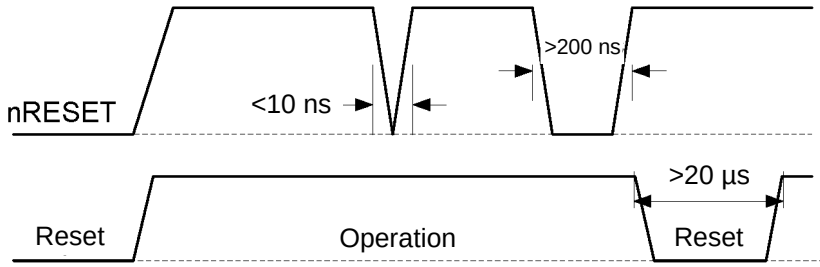


Figure 9 – Glitch filtering diagram for reset signal generation

8 Memory organization

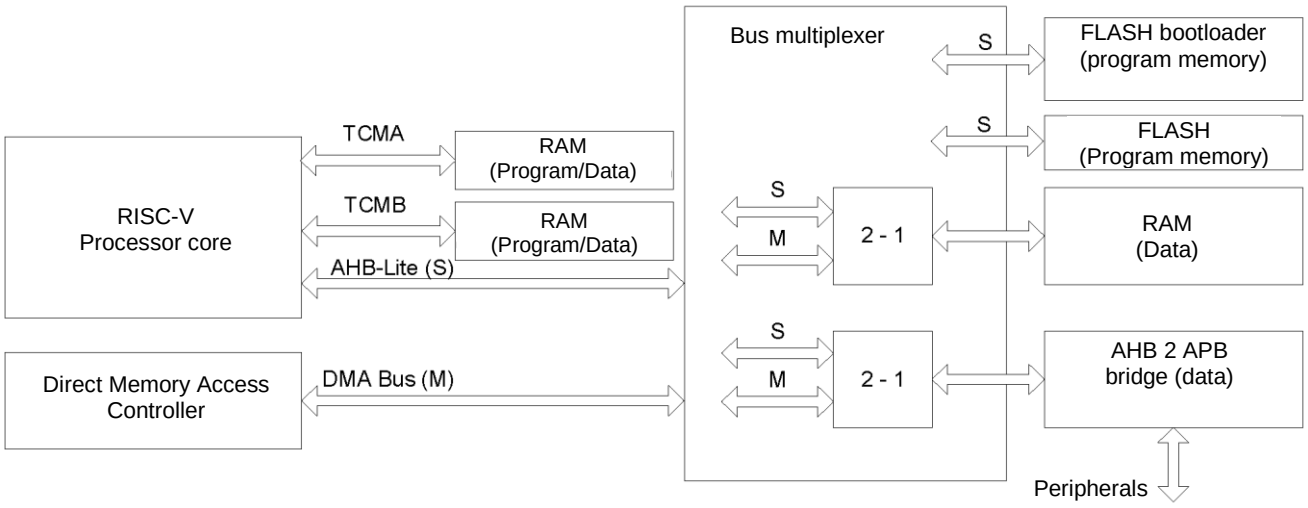


Figure 10 – Memory organization block diagram

The processor core includes two system buses:

– AHB-Lite – instruction and data fetch bus.

– TCMA, TCMB – instruction/data fetch buses.

The integrated circuit also features a Direct Memory Access (DMA) controller that performs data transfers via the DMA Bus.

The entire address space of the integrated circuit is divided into five main sections: Debug, CLINT, CLIC, I/O, and TCM. Various memory and peripheral modules are mapped into this address space.

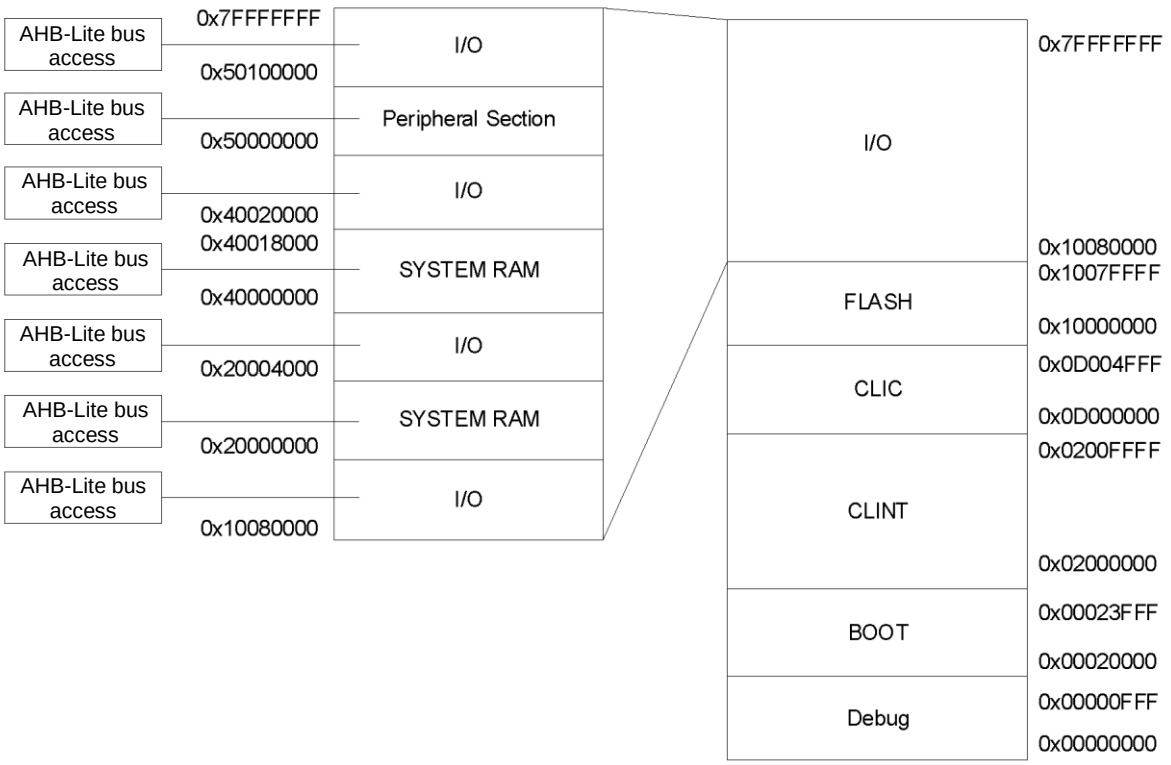


Figure 11 – Address space structure of the integrated circuit

8.1. **Integrated circuit base addresses**

Table 3 – Integrated circuit base addresses

Address	Size	Block	Note
Program memory			
0x0002_0000		BOOT	Bootloader
0x1000_0000		FLASH	Flash program memory area containing the user application
Data memory			
0x2000_0000		SYSTEM RAM	Internal RAM area on the AHB bus
Peripherals			
0x5000_0000		SSP1	SSP1 interface controller registers
0x5000_8000		UART1	UART1 interface controller registers
0x5001_0000		UART2	UART2 interface controller registers
0x5001_8000		FLASH_CNTRL	Flash memory controller registers
0x5002_0000		RST_CLK	Clock signal controller registers
0x5002_8000		DMA	Direct Memory Access controller registers
0x5003_0000		I2C	I2C interface controller registers
0x5003_8000		UART3	UART3 interface controller registers
0x5004_0000		ADC	SAR ADC control registers
0x5004_8000		WWDG	WWDG watchdog timer controller registers
0x5005_0000		IWDG	IWDG watchdog timer controller registers
0x5005_8000		POWER	Power supply voltage detector registers
0x5006_0000		BKP	Battery domain access and control registers
0x5006_8000		ADCUI	$\Delta\sum$ ADC control registers
0x5007_0000		TIMER1	Timer 1 control registers
0x5007_8000		TIMER2	Timer 2 control registers
0x5008_0000		PORTA	Port A control registers
0x5008_8000		PORTB	Port B control registers
0x5009_0000		PORTC	Port C control registers
0x5009_8000		CRC	Hardware CRC calculation block control registers
0x500A_0000		-	-
0x500A_8000		-	-
0x500B_0000		-	-
0x500B_8000		RANDOM	Random number generator block registers
0x500C_0000		USART_CNTR	USART (ISO7816) interface controller registers
0x500C_8000		SSP2	SSP2 interface controller registers
0x500D_0000		SSP3	SSP3 interface controller registers
0x500D_8000		TIMER3	Timer 3 control registers
0x500E_0000		TIMER4	Timer 4 control registers
0x500E_8000		UART4	UART4 interface controller registers
0x500F_0000		PORTD	Port D control registers
SYSTEM REGION			
0x4000_0000		SYSTEM RAM	Internal RAM area on the TCMA bus
0x4001_0000		SYSTEM RAM	Internal RAM area on the TCMB bus

9 Integrated circuit startup modes

9.1. Startup modes of IRL1206FI

The startup mode of the IC is determined at power-up or after any type of reset.

At initial power-up, the startup mode is defined by the levels on the external MODE[1:0] pins (PC[0], PC[1]), which are configured as digital inputs with internal pull-down resistors enabled. After successful execution of the bootloader program,[1] the FPOR bit is set in the BKP_LDO register.

On subsequent restarts of the IC, the levels on the MODE[1:0] pins do not affect the startup mode if the FPOR bit is set, as in this case the startup mode is determined from the MODE[1:0] bits in the BKP_LDO register.

The bootloader also applies calibration values according to the algorithm described in subsection 10.4 “Using calibration values”.

If the FPOR bit is not set, the bootloader additionally executes the recommended workaround algorithm for Errata #0002, as described in the document “IRL1206FI Errata Notice”, provided that the main supply voltage U_{CC} is not below the minimum threshold for comparison by the PVD block (PLS[2:0] = 0). If an exception occurs during bootloader execution, the IC enters an infinite loop in the “Trap_IRQHandler” exception handler.

Table 4 – Integrated circuit startup modes

MODE[1:0]	Mode	Startup address/interrupt vector table	Description
00	FLASH+J	0x10000000	The IC begins program execution from the information Flash memory, and then transfers control to the main Flash program memory. During this process, the JTAG debug interface is enabled, provided it is not locked in the Flash memory or disabled in the BKP register.
01 or 10	UART+J	User defined	The IC receives the program code for execution into RAM via the UART1 interface on pins PB[1] and PB[0]. During this process, the JTAG debug interface is enabled, provided it is not locked in the Flash memory or disabled in the BKP register.
11	RAM+J	0x40000000	The IC begins program execution from the information Flash memory, and then transfers control to the RAM program memory. During this process, the JTAG debug interface is enabled, provided it is not locked in the Flash memory or disabled in the BKP register.

If the JTAG debug interface is enabled, a JTAG adapter can be connected to the IC, allowing development tools to operate the IC in debug mode.

In debug mode, the following operations are supported:

– Erasing, writing, and reading the internal Flash program memory;

– Reading and writing the contents of RAM and peripherals;

[1] The source code of this version of the bootloader is available at https://irlabs.in/our-products/microcontrollers-and-microprocessors/ir51206fi or may be provided upon request by contacting techsupport@irlabs.in It is recommended to use the latest version of the bootloader.

- Executing the program in step-by-step mode;
- Running the program in normal mode;
- Halting program execution at breakpoints;
- Viewing variables of the executing program;
- Tracing the execution flow of the software.

9.1.1. UART+J mode

The UART+J mode provides a sufficient set of operations required to load a program into RAM (such as a Flash memory programmer), verify it, and execute it. Additionally, an external device can configure the communication baud rate. In addition to accessing RAM, access to other address ranges (FLASH, Peripherals) is also supported.

The internal HSI RC oscillator with a frequency of 8 MHz is used as the clock source for UART1. Due to frequency variation in the HSI oscillator, a calibration step is required to select the appropriate UART1 clock divider value for synchronization with the external device.

9.1.1.1. UART communication parameters

The following channel parameters are used for UART communication:

- Initial baud rate: 9600 baud;
- Data bits: 8;
- Parity: none;
- Stop bits: 1;
- The bootloader does not use the UART1 FIFO;
- The bootloader always operates as a Slave, while the external device issuing commands acts as the Master;
- Data is transmitted least significant bit first.

9.1.1.2. **UART communication protocol**

After synchronization with the Master, the bootloader enters the command manager mode. In this mode, the Master has access to the commands listed in Table 5.

9.1.1.3. **Synchronization with the external device**

Initial conditions.

During the synchronization stage with the external device (Master), the Rx pin is used as an input. The Master continuously transmits a sync symbol (0) over the channel. The bootloader adjusts its baud rate to minimize communication errors. Once the bootloader completes the adjustment, it enters command manager mode and sends a prompt to the Master (3 bytes: 0x0D (line feed), 0x0A (carriage return), 0x3E (“>”)).

The Master then stops sending sync symbols and can begin issuing commands according to the communication protocol.

9.1.2. **UART bootloader commands**

Table 5 – UART bootloader commands

Command	Code	ASCII Symbol	Description
CMD_SYNC	0x00		Empty command. The bootloader accepts it but performs no action.
CMD_CR	0x0D		Sends prompt to Master
CMD_BAUD	0x42	“B”	Sets the communication baud rate.
CMD_LOAD	0x4C	“L”	Loads byte array
CMD_VFY	0x59	“Y”	Sends byte array
CMD_RUN	0x52	“R”	Starts program execution

9.1.2.1. **CMD_SYNC command**

Empty command.

The bootloader (Slave) accepts the command but performs no action. The command code matches the synchronization symbol.

Table 8 – CMD_SYNC Command

Command code	**CMD_SYNC = 0x00**
ASCII symbol corresponding to the command code	No
Number of command parameters	0
Command format:	
Master: Sends the CMD_SYNC command code	Slave: If the command is received with errors, it returns an error message 0x45 (“E”) and terminates processing of the current command

9.1.2.2. **CMD_CR command**

Sends prompt to the Master.

Table 6 – CMD_CR command

Command code	**CMD_CR = 0x0D**
ASCII symbol corresponding to the command code	No
Number of command parameters	0
Command format:	
Master: Sends the CMD_CR command code	Slave: If the command is received with errors, it returns an error message 0x45 (“E”) and terminates processing of the current command. Otherwise, it returns 3 bytes: – CMD_CR command code; – code 0x0A; – code 0x3E (ASCII symbol “>”)

9.1.2.3. **CMD_BAUD command**

Sets the communication baud rate.

Table 7 – CMD_BAUD Command

Command code	**CMD_BAUD = 0x42**
ASCII symbol corresponding to the command code	“B”
Number of command parameters	1
Parameter	New baud rate value [baud]
Command format:	
Master: Sends the CMD_BAUD command code	Slave: If the command is received with errors, it returns an error message 0x45 (“E”) and terminates processing of the current command
Master: Sends the parameter	Slave: If the parameter is received with errors, it returns an error message 0x45 (“E”) and terminates processing of the current command. Otherwise: – Returns the CMD_BAUD command code; – After sending the CMD_BAUD code, it sets the new communication baud rate.

9.1.2.4. **CMD_LOAD command**

Loads a byte array into the IC memory.

Table 8 – CMD_LOAD command

Command code	**CMD_LOAD = 0x4C**
ASCII symbol corresponding to the command code	"L"
Number of command parameters	2
Parameter 1	Destination memory address
Parameter 2	Size of the array in bytes
Command format:	
Master: Sends the CMD_LOAD command code	Slave: If the command is received with errors, it returns an error message 0x45 ("E") and terminates processing of the current command
Master: Sends parameter 1	Slave: Waits to receive all parameters
Master: Sends parameter 2	Slave: If either parameter is received with errors, it returns an error message 0x45 ("E") and terminates processing of the current command. Otherwise, it returns the CMD_LOAD command code.
Master: Sends the byte array, least significant byte first	Slave: Receives the byte array. If any byte is received with errors, it returns an error message 0x45 ("E") and terminates processing of the current command without waiting for the rest of the array. After the array is received, it returns the response code REPLY_OK = 0x4B ("K").

9.1.2.5. **CMD_VFY command**

Sends a byte array from the IC memory.

Table 9 – CMD_VFY command

Command code	**CMD_VFY = 0x59**
ASCII symbol corresponding to the command code	"Y"
Number of command parameters	2
Parameter 1	Memory address of the data source
Parameter 2	Size of the array in bytes
Command format:	
Master: Sends the CMD_VFY command code.	Slave: If the command is received with errors, it returns an error message 0x45 ("E") and terminates processing of the current command
Master: Sends parameter 1	Slave: Waits to receive all parameters
Master: Sends parameter 2	Slave: If either parameter is received with errors, it returns an error message 0x45 ("E") and terminates processing of the current command. Otherwise: – Returns the CMD_VFY command code; – Returns the byte array, starting with the least significant byte; – Upon completion of the array transmission, returns the response code REPLY_OK = 0x4B ("K").

9.1.2.6. **The CMD_RUN command**

Starts program execution.

Table 10 – CMD_RUN Command

Command code	**CMD_RUN = 0x52**
ASCII symbol corresponding to the command code	“R”
Number of command parameters	1
Parameter	Address of the first instruction of the loaded program
Command format:	
Master: Sends the CMD_RUN command code	Slave: If the command is received with errors, it returns an error message 0x45 (“E”) and terminates processing of the current command
Master: Sends the parameter	Slave: If the parameter is received with errors, it returns an error message 0x45 (“E”) and terminates processing of the current command. Otherwise: – Returns the CMD_RUN command code; – Sets the FPOR flag; – Sets the PC value according to the received address (SP and CLIC are not reprogrammed), and thus the Slave terminates its execution. If control is transferred to a prohibited address space for code execution (for example, the address space of peripheral controllers), the error symbol 0x66 (“f”) is returned, followed by the microcontroller entering an infinite loop in the exception handler

9.1.2.7. **Receiving command parameters**

Command parameters are 4-byte values. Parameters are transmitted with the least significant byte first.

The value 0xFFFFFFFF is not allowed as a parameter.

If a hardware error is detected during parameter reception (i.e., the UART sets any error flag to “1”), parameter reception continues.

The bootloader analyzes all types of transmission errors only after all command parameters have been received.

9.1.2.8. **Error message**

The error message is represented by the character 0x45 (“E”).

After sending the error message, the bootloader enters a state where it waits for the next command. Therefore, upon receiving this message, the Master shall stop transmitting any further bytes related to the current command.

Once the error message has been received, the Master shall continue sending the CMD_CR command until it receives a valid response corresponding to that command.

10 Program Flash memory controller

The description and structure of the program Flash memory controller is provided:

– for IRL1206FI (CHIP_ID = 217), see subsection 10.1;

10.1. General description and structure of Flash memory in IRL1206FI (CHIP_ID = 217)

The integrated circuit includes on-chip program Flash memory consisting of 2 x 256 KB main memory units and 2 x 8 KB information memory units.

The memory structure is shown in Figure 12.

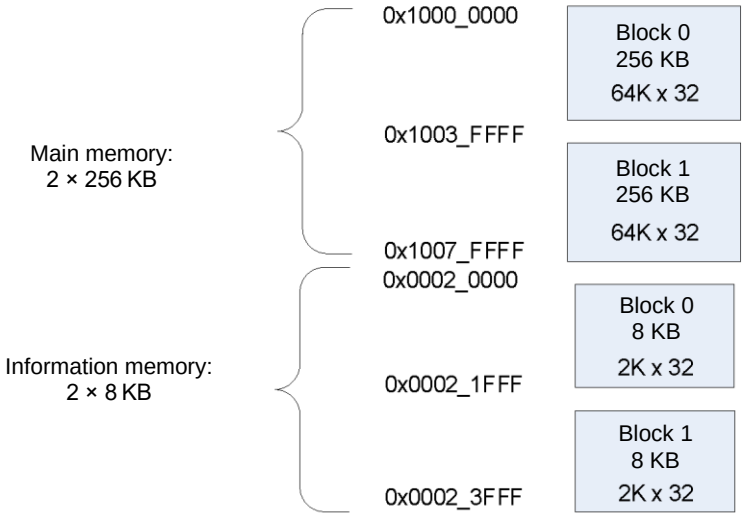


Figure 12 – Memory structure

In normal mode (bit CON = 0 in the FLASH_CMD register), the main program memory is accessible via the system bus for instruction and code data fetches.

In programming mode (bit CON = 1 in the FLASH_CMD register), both the main and information memory units are accessible as peripherals and can be used by application developers. The Flash memory cells at address 0x0002_0000 and in the range from 0x0000_3FE0 to 0x0000_3FFF contain identification data, calibration values, and Flash-memory address space protection words. Therefore, the last 4 KB of information memory are not recommended for storing user data or for use as a boot area. When overwriting the boot area, the cells at addresses 0x0002_3FE0 to 0x0002_3FFF shall be protected from erase and write operations, or their contents shall be saved in advance for later restoration at the addresses specified in Table 12.

In programming mode, program code shall be executed from the system bus area or from RAM. It is not possible to execute program code from the program Flash memory in programming mode.

10.1.1. Operation of program Flash memory in normal mode

The access time of the program Flash memory in the IRL1206FI (CHIP_ID = 217) is limited to approximately 40 ns; as a result, new data can be read from the Flash memory at a frequency of no more than 25 MHz. To allow the processor core to fetch new instructions at higher frequencies, a one-cycle processor wait state is inserted to ensure that data can be read from the Flash memory in time. When operating at frequencies below 25 MHz, no wait state is required, since the Flash memory is able to output new data within a single cycle; at frequencies from 25 to 48 MHz, one wait-state cycle is required. The number of wait-state cycles is configured in the FLASH_CMD register using the Delay[2:0] bits. Table 11 presents the required wait-state characteristics for operation of the program Flash memory in IRL1206FI (CHIP_ID = 217).

Table 11 – Wait-state characteristics for operation of the program Flash memory in IRL1206FI (CHIP_ID = 217).

Delay [2:0]	Wait-state cycles	Clock frequency
0b000	0	up to 25 MHz
0b001	1	up to 50 MHz

The number of wait-state cycles is set before increasing the clock frequency or after decreasing the clock frequency.

10.1.2. Operation of program Flash memory in programming mode

Before switching the memory to programming mode, the value 0x8AAA5551 shall be written to the FLASH_KEY register. In programming mode, the program Flash memory cannot supply instructions or data to the processor; therefore, switching the memory to programming mode (setting the CON bit = 1) is possible only from code executed from RAM.

In programming mode, the following operations are available for both the main memory (IFREN bit = 0, FLASH_CON register) and the information memory (IFREN bit = 1):

- Erasing the entire memory or the 2 × 256 KB main memory;
- Erasing a 4 KB memory page;
- Writing a 32-bit word to memory;
- Reading a 32-bit word from memory.

10.1.2.1. Erasing the entire memory or the 2 × 256 KB main memory

Memory erasure is possible only in programming mode.

Note: When erasing the information memory area, the main memory is erased automatically.

To erase the entire memory:

- Set the IFREN bit to the required value:
- 0 – for main memory;
- 1 – for main and information memory;

– Select FLASH block 0 or 1 by setting ADR[18] and ADR[13];
– Then set the XE, MAS1, and ERASE bits to “1”;
– After a time Tnvs (not less than 5 μs), set the NVSTR bit to “1”;
– Complete memory erasure takes a time Tme (20 to 40 ms). After this time, the ERASE bit shall be cleared, and after a time Tnvh1 (not less than 100 μs), the XE, MAS1, and NVSTR bits shall be cleared.
– Repeat the procedure for the second FLASH bank by setting inverted values of ADR[18] and ADR[13].

Subsequent memory operations may be performed after a time Trcv (not less than 10 μs).

The memory erase timing diagram is shown in Figure 13.

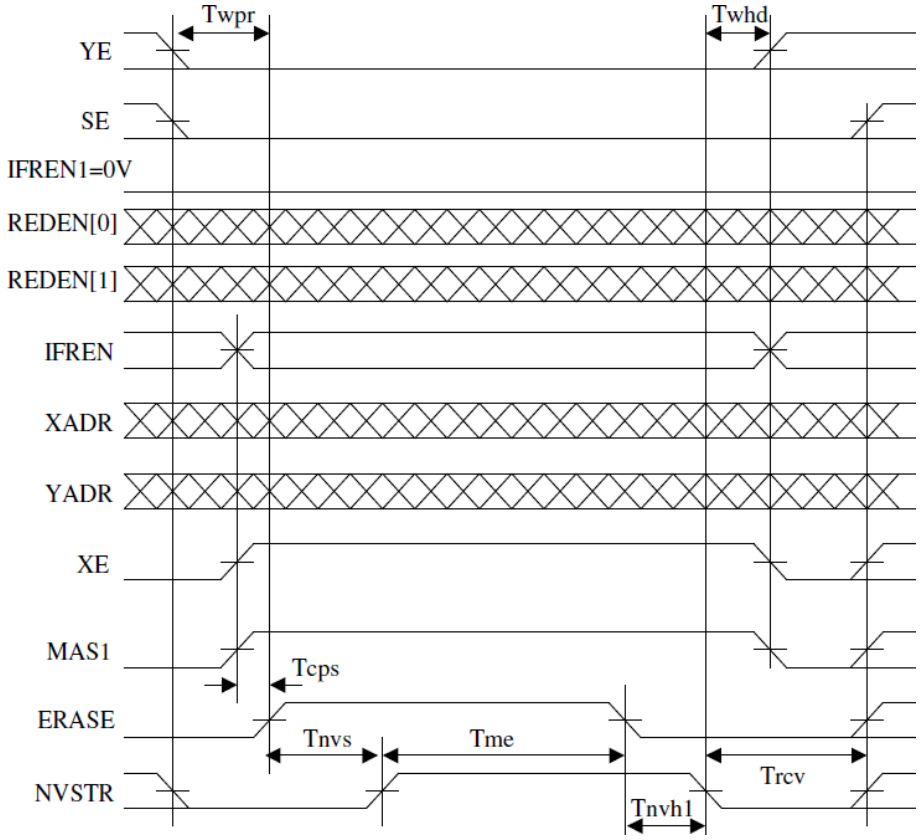


Figure 13 – Memory erase timing diagram

10.1.2.2. **Erasing a 4 KB memory page**

Memory page erasure is possible only in programming mode. To erase a memory page:

– Set the IFREN bit to the required value:
– 0 – for main memory;
– 1 – for information memory;
– Then set the address of the page to be erased in the FLASH_ADR register and set the XE and ERASE bits to “1”;
– After a time Tnvs (not less than 5 μs), set the NVSTR bit to “1”;

– Page erasure takes a time Terase (20 to 40 ms). After this time, the ERASE bit shall be cleared, and after a time Tnvh (not less than 5 µs), the XE and NVSTR bits shall be cleared.

Subsequent memory operations may be performed after a time Trcv (not less than 10 µs).

The memory page erase timing diagram is shown in Figure 14.

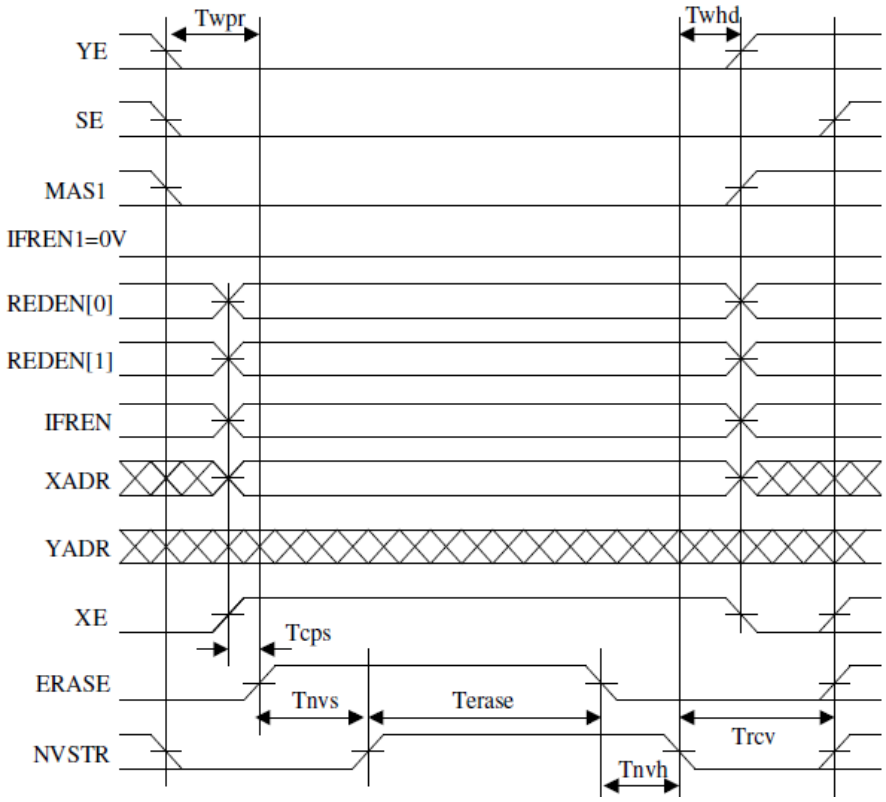


Figure 14 – Memory page erase timing diagram

10.1.2.3. **Writing a 32-bit word to memory**

Writing to memory is possible only in programming mode. The memory write timing diagram is shown in Figure 15.

To write to memory:

– Set the IFREN bit to the required value:

– 0 – for main memory;

– 1 – for information memory;

– Then set the address at which the write operation is to be performed in the FLASH_ADR register;

– Write the word to be written in the memory into the FLASH_DI register and set the XE and PROG bits to “1”;

– After a time Tnvs (not less than 5 µs), set the NVSTR bit to “1”;

– After a time Tnvs (not less than 10 µs), set the NVSTR bit to “1”;

– The memory write operation takes a time Tprog (from 20 to 40 µs). After this time, the YE bit shall be cleared.

If it is necessary to write the next word to memory, then:

– After a time Tadh (not less than 20 ns) following clearing of the YE bit, set a new address YADR (ADR[8:2]) and the value to be written to another memory cell;

– After a time Tads (not less than 20 ns), set the YE bit to “1”;

– The memory write operation takes a time Tprog (from 20 to 40 µs). After this time, the YE bit shall be cleared.

If no further write operations are required:

– After a time Tpgh (not less than 20 ns) following clearing of the YE bit, clear the PROG bit;

– After a time Tnvh (not less than 5 µs), clear the XE and NVSTR bits.

– Subsequent memory operations may be performed after a time Trcv (not less than 10 µs).

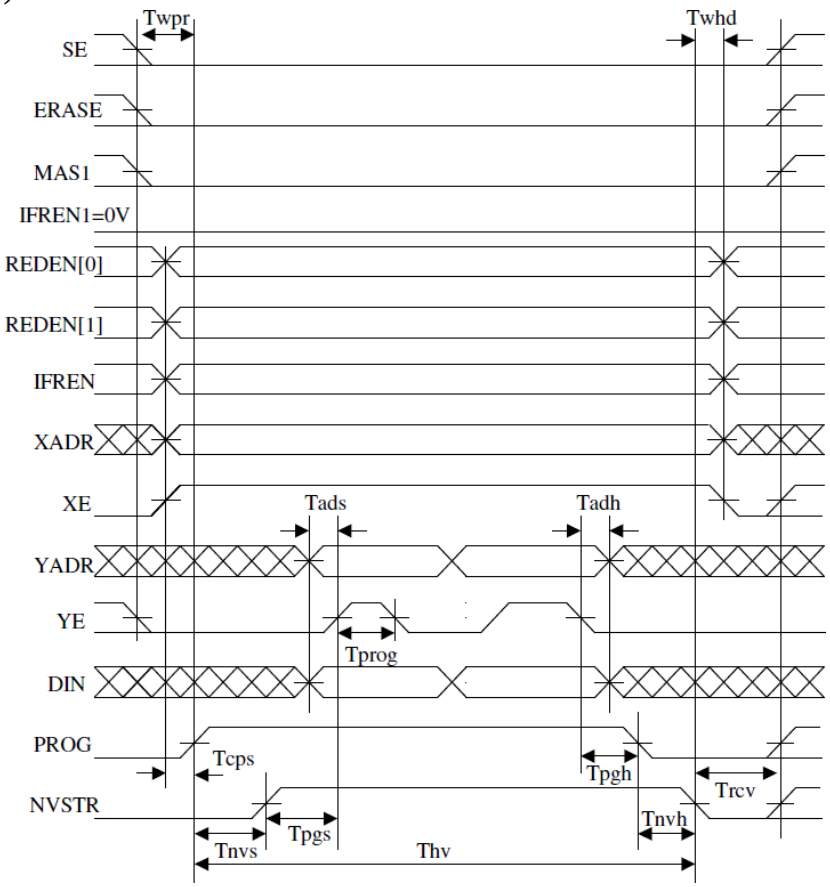


Figure 15 – Memory write timing diagram

Notes

1 The Flash memory supports at least 10,000 rewrite cycles.

2 The Thv time is the cumulative duration of application of the high programming voltage to the same row (the same XADR) before the next erase operation of that row. The Thv time shall not exceed 16 ms.

3 Data may be programmed to the same address more than twice only if an erase operation has been performed after the second programming operation.

4 “0” cannot be written to the same bit cell more than twice unless an erase operation has been performed after the second write.

10.1.2.4. **Reading a 32-bit word from memory**

In normal operating mode, both the main memory and the information memory are available for reading. To do this, it is sufficient to read the required memory address. In programming mode, both the main memory and the information memory are available for reading.

To read from memory:

- Set the required value of the IFREN bit:
- 0 – for main memory;
- 1 – for information memory;
- Write the address from which the data is to be read into the FLASH_ADR register;
- Set the XE and YE bits to “1”, and then, after a time Txys/Tas (not less than 1 ns), set the SE bit to “1”;
- After a time Tacc (not less than 40 ns), the data can be read from the FLASH_DO register.

If it is necessary to read the next word, then:

- Clear the SE bit to “0” for a duration of Tnws (not less than 5 ns);
- Write the new address to the FLASH_ADR register;
- Set the SE bit to “1”;
- After a time Tacc (not less than 40 ns), the next data can be read from the FLASH_DO register.

If no further reading is required, all control bits may be cleared.

The memory read timing diagram is shown in Figure 16.

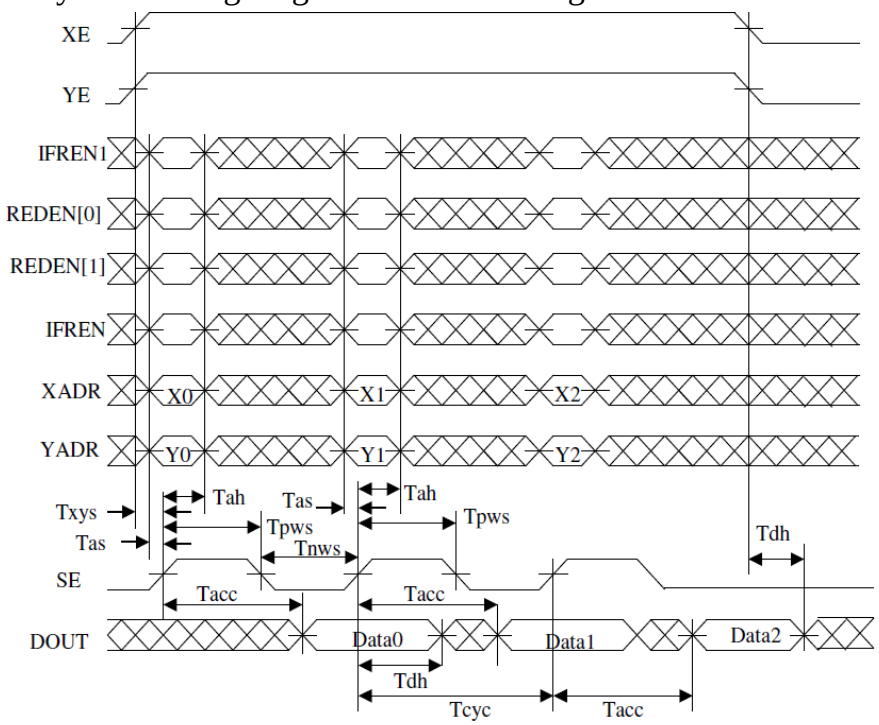


Figure 16 – Memory read timing diagram

10.2. Manufacturing information, calibration values, and Flash memory protection

After power is applied and the internal (POR) and external (nRESET) reset signals are released, the integrated circuit begins executing the program from the BOOT area. The BOOT area is an address space of the Flash memory.

Flash memory cells at address 0x0002_0000 and at offsets from 0x0000_3FE0 to 0x0000_3FFF for IRL1206FI contain identification information, calibration values, and a word used to protect the Flash memory address space. Therefore, these areas of the information memory are not recommended for storing user data or for use as a boot area. When overwriting the BOOT area, it is necessary to protect the cells with addresses from 0x0002_3FE0 to 0x0002_3FFF for IRL1206FI from erase and write operations, or to save their contents in advance for subsequent restoration at the addresses specified in Table 12.

Table 12 – Description of special fields in Flash memory

Base address	Offset for IRL1206FI (CHIP_ID = 217)	Name	Description
0x0002_0000	0x0000_3FE0-0x0000_3FF3	UNIQUE_ID	
	0x0000_3FF4	TRIM_HSI	8 bits – HSI frequency trimming
	0x0000_3FF5	TRIM_BG	8 bits – internal reference voltage (ION) trimming
	0x0000_3FF6	TRIM_LDO	8 bits – LDO voltage trimming
	0x0000_3FF7	TRIM_TS	8 bits – TS voltage offset
	0x0000_3FF8	TRIM_CS	Checksum – bits [7:0] of the arithmetic sum of all preceding 24 bytes
	0x0000_3FFC-0x0000_3FFF	-	Flash memory address space protection word

The Flash memory address space protection word consists of two fields:

- Bits 7:0 – protection bits that define which Flash memory regions have protection enabled;
- Bits 31:8 – protection activation key; when the value is 0x1234BC, enabling protection of Flash memory regions and locking the JTAG interface is permitted.

The protection word is read when the main power supply is applied. If the key value matches 0x1234BC, protection of the Flash memory address space is enabled in accordance with the configured protection bits, and the JTAG interface is also locked.

The protection bits located at address 0x0002_3FFC for IRL1206FI allow protection to be configured for 8 Flash memory regions. The active state of a protection bit is logic “0”. Each protection bit corresponds to selection of a specific protected Flash memory address range:

- Bit 0 – address range 0x10000000–0x1000FFFF;
- Bit 1 – address range 0x10010000–0x1001FFFF;

- Bit 2 – address range 0x10020000–0x1002FFFF;
- Bit 3 – address range 0x10030000–0x1003FFFF;
- Bit 4 – address range 0x10040000–0x1004FFFF;
- Bit 5 – address range 0x10050000–0x1005FFFF;
- Bit 6 – address range 0x10060000–0x1006FFFF;
- Bit 7 – address range 0x10070000–0x1007FFFF.

When at least one protected Flash memory address space is selected, protection of the BOOT area (the information Flash memory area) is also enabled. Protection of the Flash memory address space is implemented as follows:

- In programming mode, for the selected protected Flash memory address spaces and the BOOT area, execution of page erase operations, word write operations for IRL1206FI, as well as word read operations, is blocked;
- In programming mode, execution of the full memory erase operation is blocked;
- In normal operating mode, data read access via the system bus to the protected Flash memory address spaces and the BOOT area is restricted as follows:
- If the program is executed from a protected Flash memory address space or from the BOOT area, read access to all Flash memory areas is allowed;
- If the program is executed from an unprotected Flash memory address space or from any RAM area, read access to the protected Flash memory address spaces and the BOOT area is blocked (read operations return zeros).

It should be noted that after protection is enabled for at least one Flash memory address space, rewriting the protection word is no longer possible.

If a value is written to the protection word in which the key value is correct but no protected Flash memory address space is selected (0x1234BCFF), the JTAG interface will be locked; however, access to all Flash memory areas will remain available in both normal operating mode and programming mode.

Flash memory protection applies only to the 8 regions within the address range from 0x10000000 to 0x1007FFFF, at the same time, read access to Flash memory is also mirrored in the address range from 0x10080000 to 0x1FFFFFFF, in which Flash memory protection is not applied. To restrict read access to Flash memory via the address range from 0x10080000 to 0x1FFFFFFF, the Physical Memory Protection (PMP) block shall be used.

10.3. **Use of calibration values**

The values of the UNIQUE_ID, TRIM_HSI, TRIM_BG, TRIM_LDO, TRIM_TS, and TRIM_CS fields are not processed by the IC; they can only be used by software. If the TRIM_CS checksum matches the checksum calculated from TRIM_HSI,

TRIM_BG, TRIM_LDO, and TRIM_TS, then at the start of bootloader execution the following actions are performed:

- The TRIM_HSI value is written to the HSI_LSI_TRIM bits of the BKP_CLK register;
- the TRIM_BG value is written to the BG_TRIM bits of the ADC_TRIM register;
- The TRIM_LDO value is written to the LDO_BOOST bits of the BKP_LDO register.

If the TRIM_CS checksum does not match the calculated value, then:

- The following value is written to the HSI_LSI_TRIM bits of the BKP_CLK register:
- “01 1110” for IRL1206FI (CHIP_ID = 217).
- The following value is written to the BG_TRIM bits of the ADC_TRIM register:
- “1 0100” for IRL1206FI (CHIP_ID = 217).

Writing “median” values in the absence/erasure of calibration values programmed during manufacturing does not improve IC characteristics and does not guarantee stable operation of the IC in all operating modes. Therefore, in such cases it is recommended to perform calibration independently.

Calibration of TRIM_TS is not implemented.

For integrated circuits without a bootloader, the trimming write algorithm described above shall be implemented independently.

10.4. **Program Flash memory controller control registers**

Table 13 – Program Flash memory controller control registers

Base Address	Name	Description
0x5001_8000	FLASH_CNTRL	Flash memory controller registers
Offset		
0x00	FLASH_CMD	Flash memory control register
0x04	FLASH_ADR	Address register
0x08	FLASH_DI	Write data register
0x0C	FLASH_DO	Read data register
0x10	FLASH_KEY	FLASH key register
0x14	FLASH_CTRL	Flash memory control and test register in IRL1206AFI (CHIP_ID = 215)
0x18	CHIP_ID_CTRL	CHIP_ID and OTP control register
0x1C	OTP_KEY	OTP key register

10.4.1. **FLASH_CMD for IRL1206FI (CHIP_ID = 217)**

Table 14 – FLASH_CMD Register

Number	31…15	14	13	12	11	10	9	8
Access	U	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	1	0	0	0	0	0	0
		TMR	NVSTR	PROG	MAS1	ERASE	IFREN	SE

Number	7	6	5…3	2, 1	0
Access	R/W	R/W	R/W	U	R/W
Reset	0	0	001	0	0
	YE	XE	Delay[2:0]		CON

Access designations:

R/W – the bit is available for read and write access;

RO – the bit is available for read-only access;

U – the bit is not physically implemented or is reserved.

Table 15 – Description of the FLASH_CMD register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...15	-	Reserved
14	TMR	Reset of Flash memory test mode: 0 – test enabled; 1 – reset. It is recommended to always write “1”
13	NVSTR	Write or erase operations: 0 – during read; 1 – during write or erase
12	PROG	Write data from the FLASH_DI register: 0 – no write; 1 – write enabled
11	MAS1	Erase the entire block when ERASE = 1: 0 – no erase; 1 – erase. When IFREN = 0, the main memory is erased. When IFREN = 1, both the main and information memory are erased
10	ERASE	Erase a 4 KB memory page: 0 – no erase; 1 – erase
9	IFREN	Operation with the information block: 0 – main memory; 1 – information memory
8	SE	Sense amplifier: 0 – disabled; 1 – enabled
7	YE	Output of address ADR[8:2]: 0 – not enabled; 1 – enabled
6	XE	Output of address ADR[17:9]:

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
		0 – not enabled; 1 – enabled
5…3	Delay[2:0]	Program memory read delay in cycles (in normal operating mode): 000 – 0 cycles; 001 – 1 cycle
2, 1	-	Reserved
0	CON	Switching the Flash memory controller to register-based control; this shall not be performed while executing code from the FLASH area: 0 – FLASH controlled by the core, normal operating mode; 1 – register-based control, programming mode

10.4.2. **FLASH_ADR for IRL1206FI (CHIP_ID = 217)**

Table 16 – FLASH_ADR Register

Number	31…0
Access	R/W
Reset	0
	ADR [31:0]

Table 17 – Description of the FLASH_ADR register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...0	ADR[31:0]	Memory access address. When IFREN = 0 (main memory): – ADR[18] – block number; – ADR[17:12] – page address (XADR[8:3]); – ADR[11:9] – row address (XADR[2:0]); – ADR[8:2] – word address (YADR[6:0]); – ADR[1:0] – no value. The minimum addressable unit is 32 bits. When IFREN = 1 (information memory): – ADR[13] – block number; – ADR[12] – page address (XADR[3]); – ADR[11:9] – row address (XADR[2:0]); – ADR[8:2] – word address (YADR[6:0]); – ADR[1:0] – no value. The minimum addressable cell is 32 bits

10.4.3. **FLASH_DI**

Table 18 – FLASH_DI Register

Number	31…0
Access	R/W
Reset	0
	DATA [31:0]

Table 19 – Description of the FLASH_DI register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...0	DATA[31:0]	Data to write to FLASH

10.4.4. **FLASH_DO**

Table 20 – FLASH_DO Register

Number	31…0
Access	R/W
Reset	0
	DATA [31:0]

Table 21 – Description of the FLASH_DO register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...0	DATA[31:0]	Data read from FLASH

10.4.5. **FLASH_KEY**

Table 22 – FLASH_KEY Register

Number	31…0
Access	R/W
Reset	0
	KEY [31:0]

Table 23 – Description of the FLASH_KEY register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...0	KEY[31:0]	Key for enabling access to Flash memory via register access. Before switching the memory to programming mode, the value 0x8AAA5551 shall be written to the FLASH_KEY register. After completion of Flash memory write or erase operations, the key shall be cleared to disable register-based access

10.4.6. **CHIP_ID_CTRL**

Table 24 – CHIP_ID_CTRL Register

Number	31…10	9…2	1	0
Access	U	R	R	R/W
Reset	0	CHIP_ID	0/1	0
		CHIP_ID	OTPTST2	OTP_PROG

Table 25 – Description of the CHIP_ID_CTRL register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...10	-	Reserved
9...2	CHIP_ID	11011001 – ID of IRL1206FI
1	OTPTST2	Test bit; may be in any state
0	OTP_PROG	When writing the sequence 1 – delay(1 ms) – 0 – delay(1 ms) – 1 – delay(1 ms) – 0 – delay(1 ms) – 1 – delay(1 ms) – 0 the OTP memory cells are programmed. After this, upon the first application of power to the IC, the JTAG interface is locked (implemented in ICs starting from revision 2) with no possibility of recovery. By default, write access to this register is blocked and is unlocked by writing the key to OTP_KEY

10.4.7. **OTP_KEY**

Table 26 – OTP_KEY Register

Number	31…0
Access	R/W
Reset	0
	OTP_KEY [31:0]

Table 27 – Description of the OTP_KEY register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...0	OTP_KEY[31:0]	Key for enabling programming of OTP memory cells. To unlock write access to the CHIP_ID_CTRL register, the key 0x1555AAA8 shall be written

11 Instruction set

The integrated circuit implements the RISC-V RV32IMCNZX instruction set, including:

- The standard RV32I extension;
- The standard M extension (integer multiplication and division);
- The standard C extension (16-bit instructions to increase code density);
- The standard N extension (support for interrupts in user privilege mode);
- The standard ZBA, ZBB, ZBC, ZBS extensions (bit-manipulation instructions);
- The standard ZBKB, ZBKC, ZBKX, ZKND, ZKNE, ZKNH, ZKSED, ZKSH extensions for scalar cryptography;
- The proprietary XGOST extension.

The supported instructions are listed in Table 28.

Table 28 – Instruction set of the BM-310S0 processor

Instruction mnemonic	Format	Operands	Brief description
lb	I	rd, rs1, imm	Load byte
lh	I	rd, rs1, imm	Load halfword
lw	I, Cx	rd, rs1, imm	Load word
lbu	I	rd, rs1, imm	Load byte unsigned
lhu	I	rd, rs1, imm	Load word unsigned
sb	S	rs1, rs2, imm	Store byte
sh	S	rs1, rs2, imm	Store halfword
sw	S, Cx	rs1, rs2, imm	Store word
add	R, Cx	rd, rs1, rs2	Arithmetic addition
addi	I, Cx	rd, rs1, imm	Arithmetic addition with immediate value
sub	R, Cx	rd, rs1, rs2	Arithmetic subtraction
lui	U	rd, imm	Load upper immediate
auipc	U	rd, imm	Add upper immediate to PC
xor	R	rd, rs1, rs2	Logical exclusive OR
xori	I	rd, rs1, imm	Logical exclusive OR with immediate
or	R, Cx	rd, rs1, rs2	Logical OR
ori	I	rd, rs1, imm	Logical OR with immediate
and	R, Cx	rd, rs1, rs2	Logical AND
andi	I	rd, rs1, imm	Logical AND with immediate
sll	R	rd, rs1, rs2	Logical shift left
slli	I, Cx	rd, rs1, shamt	Logical shift left by immediate
srl	R	rd, rs1, rs2	Logical shift right
srli	I	rd, rs1, shamt	Logical shift right by immediate
sra	R	rd, rs1, rs2	Arithmetic shift right
srai	I	rd, rs1, shamt	Arithmetic shift right by immediate
slt	R	rd, rs1, rs2	Comparison “<”
slti	I	rd, rs1, imm	Comparison “<” with immediate
sltu	R	rd, rs1, rs2	Comparison “<” unsigned
sltiu	I	rd, rs1, imm	Comparison “<” unsigned with immediate
beq	SB, Cx	rs1, rs2, imm	Branch if equal
bne	SB, Cx	rs1, rs2, imm	Branch if not equal

Instruction mnemonic	Format	Operands	Brief description
blt	SB	rs1, rs2, imm	Branch if less than
bge	SB	rs1, rs2, imm	Branch if greater than or equal
bltu	SB	rs1, rs2, imm	Branch if less than, unsigned
bgeu	SB	rs1, rs2, imm	Branch if greater than or equal, unsigned
jal	UJ, Cx	rd, imm	Jump and link
jalr	UJ, Cx	rd, rs1, imm	Jump and link register
fence.i	I	-	Synchronization of the instruction fetch stream and the data read/write stream
rdinstret	I	rd	Pseudo-instruction (alias) for the instruction that reads the executed-instructions counter
ecall	I	-	Request to enter the operating system execution environment
ebreak	I	-	Transfer of control to the debug execution environment
rdcycle	I	rd	Pseudo-instruction (alias) for the instruction that reads the processor cycle counter
rdcycleh	I	rd	Pseudo-instruction (alias) for the instruction that reads the most significant bits of the processor cycle counter
rdtime	I	rd	Pseudo-instruction (alias) for the instruction that reads the value of the core low-frequency system timer
rdtimeh	I	rd	Pseudo-instruction (alias) for the instruction that reads the most significant bits of the core low-frequency system timer
rdinstret	I	rd	Pseudo-instruction (alias) for the instruction that reads the executed-instructions counter
rdinstreth	I	rd	Pseudo-instruction (alias) for the instruction that reads the most significant bits of the executed-instructions counter
mul	R	rd, rs1, rs2	Multiplication
mulh	R	rd, rs1, rs2	Multiplication with return of the most significant bits of the result
mulhsu	R	rd, rs1, rs2	Signed-by-unsigned multiplication with return of the most significant bits of the result
mulhu	R	rd, rs1, rs2	Unsigned-by-unsigned multiplication with return of the most significant bits of the result
div	R	rd, rs1, rs2	Division
divu	R	rd, rs1, rs2	Unsigned division
rem	R	rd, rs1, rs2	Remainder
remu	R	rd, rs1, rs2	Unsigned remainder
andn	R	rd, rs1, rs2	Bitwise AND of rs1 and the bitwise inversion of rs2
clmul	R	rd, rs1, rs2	Carry-less multiplication (lower part)
clmulh	R	rd, rs1, rs2	Carry-less multiplication (upper part)
clmulr	R	rd, rs1, rs2	Carry-less multiplication of bit-reversed operands followed by bit-reversal of the result
clz	R	rd, rs	Count leading zeros in rs1
cpop	R	rd, rs	Count set bits in rs1
ctz	R	rd, rs	Count trailing zeros in rs1
max	R	rd, rs1, rs2	Maximum of two signed integers
maxu	R	rd, rs1, rs2	Maximum of two unsigned integers
min	R	rd, rs1, rs2	Minimum of two signed integers
minu	R	rd, rs1, rs2	Minimum of two unsigned integers
orc.b	R	rd, rs1, rs2	Bitwise OR across all bits of each byte of the word
orn	R	rd, rs1, rs2	Bitwise OR of rs1 and the bitwise inverted rs2
rev8	R	rd, rs	Byte reversal

Instruction mnemonic	Format	Operands	Brief description
rol	R	rd, rs1, rs2	Rotate rs1 left
ror	R	rd, rs1, rs2	Rotate rs1 right
rori	I	rd, rs1, shamt	Rotate right by immediate
bclr	R	rd, rs1, rs2	Clear the bit with the index specified in rs2
bclri	I	rd, rs1, imm	Clear the bit with the index specified by an immediate value
bext	R	rd, rs1, rs2	Extract the bit with the index specified in rs2
bexti	I	rd, rs1, imm	Extract the bit with the index specified by an immediate value
binv	R	rd, rs1, rs2	Invert the bit with the index specified in rs2
binvi	I	rd, rs1, imm	Invert the bit with the index specified by an immediate value
bset	R	rd, rs1, rs2	Set the bit with the index specified in rs2
bseti	I	rd, rs1, imm	Set the bit with the index specified by an immediate value
sext.b	R	rd, rs	Sign extension of the least significant byte of rs
sext.h	R	rd, rs	Sign extension of the halfword rs
sh1add	R	rd, rs1, rs2	Shift rs1 left by 1 and add rs2
sh2add	R	rd, rs1, rs2	Shift rs1 left by 2 and add rs2
sh3add	R	rd, rs1, rs2	Shift rs1 left by 3 and add rs2
xnor	R	rd, rs1, rs2	Exclusive OR inversion
zext.h	R	rd, rs	Zero extension of a halfword to word size
Note: The following notation is used in the Operands column: – rs – source register; – rd – destination register; – imm or shamt – immediate value			

Detailed descriptions and instruction formats are provided in the following documents:

- RISC-V Instruction Set Manual Volume I: User-Level ISA Document Version 2.2;
- RISC-V Bit-Manipulation ISA-extensions. Version 1.0.0;
- RISC-V Cryptography Extensions Volume I. Scalar and Entropy Source Instructions. Version 1.0.0.

11.1. **Wait for Interrupt support**

The processor core supports the WFI (Wait for Interrupt) instruction. When executing the WFI instruction, the processor core stops all activity on external interfaces and enters interrupt wait mode. In interrupt wait mode, to reduce power consumption, the processor core disables the clock signal until an interrupt occurs.

A detailed description and the instruction format of the WFI instruction are provided in the document RISC-V Instruction Set Manual, Volume II: Privileged Architecture. Version 1.11.

12 BM-310S0 processor core

The BM-310S0 supports two privilege modes: machine and user. User mode provides a mechanism for isolating processes from each other and from trusted code executed in machine mode. A detailed description of the privilege modes is provided in the document RISC-V Instruction Set Manual Volume II: Privileged Architecture.

The BM-310S0 pipeline consists of three stages, which perform the following operations:

Stage 1 – generation of a request to the program memory subsystem (PMS);

Stage 2 – reading a code fragment from the PMS and decoding the instruction;

Stage 3 – execution of the instruction.

The instruction front-end (FE) subsystem organizes execution of the first two pipeline stages.

The FE unit consists of the following sub-units:

- IFU – Instruction Fetch Unit – contains the logic for generating the address of the next code fragment;
- IDU – Instruction Decode Unit;
- RAS – Return Address Stack – a stack of function return addresses.

At stage 1, the IFU generates requests to fetch code fragments from the program memory subsystem (PMS), using fetch redirection information from the IDU, the RAS, and the instruction execution subsystem (BE).

At stage 2, a code fragment is fetched from program memory. The code fragment may contain one or two instructions. After that, one instruction is decoded, and the result is placed into the decoded instruction queue (consisting of two entries) or into its bypass path to the instruction execution subsystem (BE).

At stage 3, the instruction execution subsystem (BE) receives the decoded instruction and its operands from the register file, checks whether it can be executed, and executes it on the appropriate execution unit.

Instructions executed by the processor core have different execution latencies. Instructions of the I extension are executed in one clock cycle, except for read instructions, which are executed in two clock cycles. Instructions that access CSR registers are executed with an empty pipeline. The division execution unit is iterative and is responsible for executing the DIV[U] and REM[U] instructions. The execution latencies of the M extension instructions are given in Table 29. The CLMUL, CLMULH, and CLMULR instructions from the ZBC extension are executed in eight clock cycles. All other bit-manipulation extension instructions are executed in one clock cycle. Instructions from the scalar cryptography extension are executed in one clock cycle.

Table 29 – Instruction execution latency

Instruction	Latency, cycles
MUL	1
MULH	1
MULHS	1
MULHSU	1
DIV	2 to 16
DIVU	2 to 16
REM	2 to 16
REMU	2 to 16

The program memory subsystem (PMS) and the data memory subsystem (DMS) provide access to the integrated memory (TCM) and to the I/O address range, part of which is reserved for internal blocks of the processor complex, while the remaining part can be used to access devices on the AHB bus. Access to the TCM is provided through dedicated interfaces. The core uses a strongly ordering memory access model on the AHB bus, meaning that the next AHB bus request is not issued until execution of the current request is completed. If an error occurs on the AHB bus upon completion of request processing, an exception (access fault) is generated by the core.

12.1. **Processor block diagram**

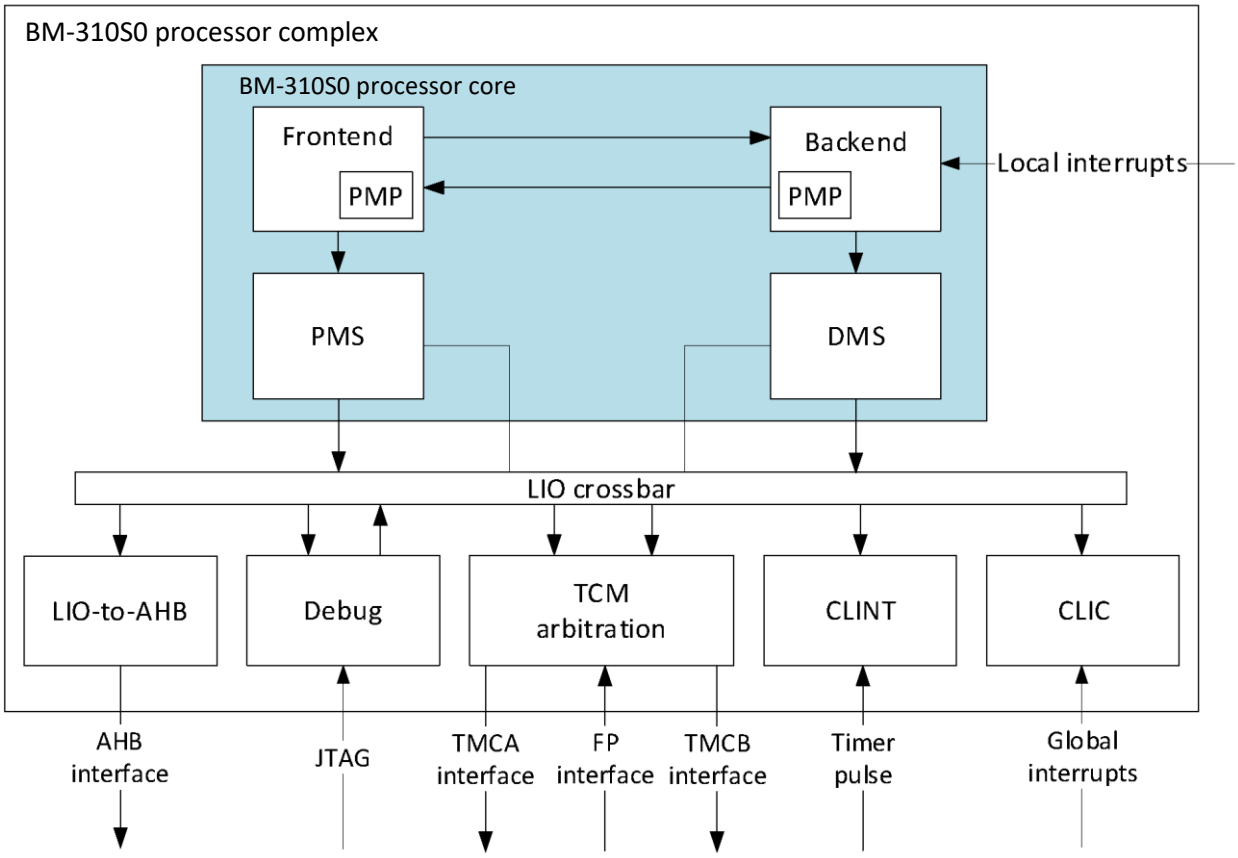


Figure 17 – Processor block diagram

12.1.1. **TCM interface**

The TCM interface is used to access the integrated program/data memory. The TCM interface is divided into two parts: the PMS interface and the DMS interface. The TCM interface address bus width is 14 bits, word addressing is used, and memory capacities of up to 128 KB are supported. The data bus width is 32 bits.

12.1.2. **TCM memory block access arbiter**

External bus master blocks on the AHB bus (for example, a DMA block) can use the AHB slave interface of the processor complex to access the TCM memory blocks. Requests from an external bus master are forwarded to the TCM memory block access arbiter, where arbitration is performed between these requests and requests from the program memory subsystem (PMS) and the data memory subsystem (DMS). If requests from PMS/DMS and from an external block in the same cycle target different TCMA and TCMB blocks, access occurs simultaneously (without conflict). If requests target the same TCM memory block, the request is selected according to the following rules:

- A request from the DMS has higher priority than a request from the PMS;
- Arbitration between the selected PMS/DMS request and the external bus master request is performed using a round-robin scheme.

Atomicity between accesses of the processor core and external bus master blocks is not guaranteed.

12.1.3. **AHB I/O interface**

The processor complex uses the AHB I/O interface to access peripheral device registers or non-cacheable memory. The AHB I/O interface uses a 32-bit physical address. The data bus width of the AHB I/O interface is 32 bits. Execution of code from the I/O address range is supported.

12.1.4. **Interrupt requests**

Local interrupt requests are connected to the Core-Local Interrupt Controller (CLIC). Signals connected to the local interrupts shall be synchronous with the core clock. Interrupt requests are edge-sensitive signals.

12.1.5. **JTAG interface**

The implementation shall set the manufid signal to the correct value, which is the integrated circuit manufacturer identifier (JEDEC ID). The JTAG interface is connected directly to the debug subsystem of the processor complex. The Debug block is not reset by the ndmreset signal. The Debug block is reset by the POR reset signal, the external nRESET signal, and internal reset signals from watchdog timers. For correct operation of the Debug block, it shall be reset at system startup.

12.1.6. **LIO crossbar**

Local core bus.

12.1.7. **Physical Memory Protection (PMP)**

To ensure secure data processing and to prevent faults, it is recommended to restrict the physical addresses accessible to software. The optional Physical Memory Protection (PMP) module provides control registers in M-mode that allow access permissions (read, write, execute) to be defined for each physical memory region.

The granularity of PMP access control configuration depends on the platform and may vary between physical memory regions; however, the standard PMP encoding supports regions as small as four bytes. Privileges for specific regions can be configured programmatically; for example, some regions may be accessible only in M-mode and not at lower privilege levels. PMP checks are applied to all accesses when the core is operating in U-mode, as well as to load and store operations when the MPRV bit is set in the mstatus register and the MPP field in the mstatus register contains U. In addition, PMP checks may be applied to accesses in M-mode, in which case the PMP registers themselves are locked, so that even M-mode software cannot modify them without a core reset. In essence, PMP can grant permissions to U-mode that are not present by default and can revoke permissions from M-mode, which by default has full permissions. PMP violations are always detected by the core.

12.1.7.1. **CSR registers for physical memory protection**

PMP entries are defined by an 8-bit configuration register and a single 32-bit address register. In some PMP configurations, an additional address register associated with the preceding PMP entry is also used. Up to 8 PMP entries are supported. All fields of the PMP CSR registers are WARL (write any value, read only valid, legal values) and can be cleared. PMP CSR registers are accessible only in M-mode. PMP configuration registers are tightly packed within CSRs to minimize context switch time. For the presented core architecture, two CSR registers, pmpcfg0–pmpcfg1, are implemented, containing the pmp0cfg–pmp7cfg configurations for the 8 PMP entries.

Table 30 – PMP CSR registers (Machine)

Address	Name	Access	Description
0x3A0	pmpcfg0	R/W	Physical Memory Protection configuration register 0
0x3A1	pmpcfg1	R/W	Physical Memory Protection configuration register 1
0x3B0	pmpaddr0	R/W	Physical Memory Protection address register 0
0x3B1	pmpaddr1	R/W	Physical Memory Protection address register 1
0x3B2	pmpaddr2	R/W	Physical Memory Protection address register 2
0x3B3	pmpaddr3	R/W	Physical Memory Protection address register 3
0x3B4	pmpaddr4	R/W	Physical Memory Protection address register 4
0x3B5	pmpaddr5	R/W	Physical Memory Protection address register 5
0x3B6	pmpaddr6	R/W	Physical Memory Protection address register 6
0x3B7	pmpaddr7	R/W	Physical Memory Protection address register 7

Table 31 – Layout of the PMP configuration CSR pmpcfg0

Number	31...24	23...16	15…8	7…0
Access	WARL	WARL	WARL	WARL
Reset	00000XXX	00000XXX	00000XXX	00000XXX
	pmp3cfg	pmp2cfg	pmp1cfg	pmp0cfg

Table 32 – Layout of the PMP configuration CSR register pmpcfg1

Number	31...24	23…16	15...8	7...0
Access	WARL	WARL	WARL	WARL
Reset	00000XXX	00000XXX	00000XXX	00000XXX
	pmp7cfg	pmp6cfg	pmp5cfg	pmp4cfg

The PMP address registers are CSRs named pmpaddr0–pmpaddr7. Each PMP address register encodes bits 33…2 of the 34-bit physical address, as shown in Table 33. Because not all physical address bits may be implemented, the pmpaddr registers are WARL.

Table 33 – PMP address registers pmpaddr0-pmpaddr7

Number	31..0
Access	WARL
Reset	0
	address

Table 34 – Description of bits of the PMP address registers pmpaddr0-pmpaddr7

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…0	address	Bits 33…2 for 34-bit physical address

Table 35 shows the format of the PMP configuration register. When set, the R, W, and X bits indicate that the PMP entry permits read, write, and instruction execution, respectively. If any of these bits is cleared, the corresponding type of access is prohibited. The combination R = 0 and W = 1 is reserved for future use. The A and L fields are described in the following sections.

Table 35 – PMP configuration registers pmp0cfg-pmp7cfg

Number	7	5…6	4...3	2	1	0
Access	WARL	WARL	WARL	WARL	WARL	WARL
Reset	0	00	00	X	X	X
	L	-	A	X	W	R

Table 36 – Description of bits of the PMP configuration registers pmp0cfg-pmp7cfg

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
7	L	Locking of the corresponding PMP entry: 1 – the PMP entry is locked; the R/W/X permissions apply to accesses in both M-mode and U-mode, and writes to the configuration register and the corresponding address registers are ignored; 0 – the PMP entry is unlocked; access in M-mode is unrestricted, and the R/W/X permissions apply only to accesses in U-mode.
6...5	-	Reserved. It is recommended to leave the value of this field unchanged.
4...3	A	Address matching mode encoding: 00 – null region (entry disabled); 01 – top-of-range (TOR); 10 – naturally aligned four-byte region (NA4); 11 – naturally aligned power-of-two region (NAPOT), ≥ 8 bytes.
2	X	Instruction execution permission: 1 – enabled; 0 – disabled
1	W	Write permission: 1 – enabled; 0 – disabled
0	R	Read permission: 1 – enabled; 0 – disabled

An attempt to fetch an instruction from a PMP region that does not have execute permission causes an instruction access fault exception. An attempt to execute a load instruction whose address lies within a PMP region without read permission causes a load access fault exception. An attempt to execute a store instruction whose address lies within a PMP region without write permission causes a store access fault exception.

12.1.7.2. **Address matching**

The A field in the PMP entry configuration register encodes the address matching mode for the corresponding PMP address register. The encoding of this field is shown in Table 48. When A = 0, the PMP entry is disabled and does not match any addresses. Two other address matching modes are supported: naturally aligned power-of-two regions (NAPOT), including the special case of naturally aligned four-byte regions (NA4), and top-of-range (TOR). These modes support four-byte granularity.

Table 37 – Decoding of the A field in the PMP configuration register

A	Mode designation	Description
0	OFF	Null region (entry disabled)
1	TOR	Top-of-range
2	NA4	Naturally aligned four-byte region
3	NAPOT	Naturally aligned power-of-two region, ≥ 8 bytes.

If the TOR option is selected, the corresponding address register defines the upper bound of the address range, while the preceding PMP address register defines the lower bound of the address range. If the A field of PMP entry *i* is set to TOR, the entry matches any address y such that $pmpaddr_{i-1} \leq y < pmpaddr_i$. If the A field of PMP entry *0* is set to TOR, zero is used as the lower bound, and therefore it matches any address $y < pmpaddr_0$.

When the NAPOT region is selected, the corresponding address register encodes the region size (the least significant bits) and the region base address (the most significant bits), as shown in Table 38. A zero-valued bit serves as the delimiter between the size and the base address.

Table 38 – Decoding of the NAPOT region in PMP address and configuration registers

pmpaddr	pmpcfg.A	Region type and size mapping
yyyy…yyyy	NA4	4 bytes
yyyy…yyy0	NAPOT	8 bytes
yyyy…yy01	NAPOT	16 bytes
yyyy…y011	NAPOT	32 bytes
…	…	
yy01…1111	NAPOT	2^{32} bytes
y011…1111	NAPOT	2^{33} bytes
0111…1111	NAPOT	2^{34} bytes
1111…1111	NAPOT	2^{35} bytes

Table 39 shows examples of address register encoding when using the NAPOT mode.

Table 39 – Examples of address register encoding in NAPOT mode

pmpaddr value	Region base address	Region size
0x1000_0000	0x4000_0000	8 bytes
0x1000_0003	0x4000_0000	32 bytes
0x1000_01FF	0x4000_0000	4 KB

Software can determine the PMP granularity by writing zero to pmp0cfg, then writing all ones to pmpaddr0, and then reading back the value of pmpaddr0. If G is the index of the least significant bit set to one, then the PMP granularity is 2^{G+2} bytes.

12.1.7.3. **Locking and privilege mode**

The L bit indicates that the PMP entry is locked, meaning that writes to the configuration register and the corresponding address registers are ignored. Locked PMP entries can be unlocked only by a core reset. If PMP entry *i* is locked, writes to pmpicfg and pmpaddr*i* are ignored. In addition, if TOR is set in pmp*i*cfg.A, writes to pmpaddr$_{i-1}$ are ignored. In addition to locking the PMP entry, the L bit indicates whether the R/W/X permissions apply to accesses in M-mode. If the L bit is set, these permissions apply to all privilege modes. If the L bit is cleared, any M-mode access matching the PMP entry succeeds, and the R/W/X permissions apply only to U-mode.

12.1.7.4. **Priority and matching logic**

PMP entries have a static priority. The PMP entry with the lowest index that matches any byte alignment of the access determines whether the access succeeds or fails. A PMP entry must match all bytes requested by the access; otherwise, the access fails regardless of the L, R, W, and X bits.

If a PMP entry matches all bytes requested by the access, the L, R, W, and X bits determine whether the access succeeds. If the L bit is cleared and the access privilege mode is M, the access succeeds. Otherwise, if the L bit is set or the access privilege mode is U, the access succeeds only if the R, W, or X bit corresponding to the access type is set.

13 ADC block for measuring voltages and currents in the electrical power network

The integrated circuit includes a block of seven 24-bit ΔΣ ADC channels. All channels are organized into three pairs F0–F2 (one voltage channel and one current channel) for a three-phase network, plus one additional independent current channel. Each of the seven channels digitizes the input signal with an output sampling rate of up to 16 kHz. In addition, each channel pair F0–F2 implements functionality for calculating RMS current/voltage values, computing active and reactive power, calculating consumed active and reactive energy, determining the signal frequency in the voltage channels, detecting peak value exceedance, and detecting signal drops below a specified threshold. These additional blocks reduce the load on the processor, which in turn reduces the overall power consumption of the chip. In addition, each ADC has an independent DMA channel, enabling data to be stored in RAM without processor involvement.

The block diagrams of the seven ADC channels are shown in Figures 20, 21, 22.

To prevent the influence of high-frequency interference on computation results, an external anti-aliasing filter should be used. For example, a simple first-order RC filter with a cutoff frequency of about 10–100 kHz can be used, depending on the application of the integrated circuit.

All digital filters (LPF, HPF, SINC3) are configured for the specified cutoff frequencies when the ADCUI block is clocked from HSE = 8.192 MHz; in this case, the ADC clock frequency may be 1.024 MHz (output sampling rates 1/2/4 kHz depending on the OSR_CONF setting), 2.048 MHz (output sampling rates 2/4/8 kHz depending on the OSR_CONF setting), or 4.096 MHz (output sampling rates 4/8/16 kHz depending on the OSR_CONF setting), depending on the configuration of the CLC_ADC_CFG field in the ADCUI_CLKPHASE1 register. **Compliance with the block parameters is guaranteed only when operating at sampling rates of 4/8/16 kHz.**

The filter transfer characteristics are described by the following formulas:

– for the HPF filter

$$H(z) = \frac{1}{2} \cdot \frac{1-c+(c-1)\cdot z^{-1}}{1+c\cdot z^{-1}}, \quad (1)$$

where c is the two's-complement value of the CHPF_COEFF field of the ADCUI_FILTERCFG register divided by 2^{11}.

For the reset value of CHPF_COEFF, c = 802 h / 2^{11} = −0.999023.

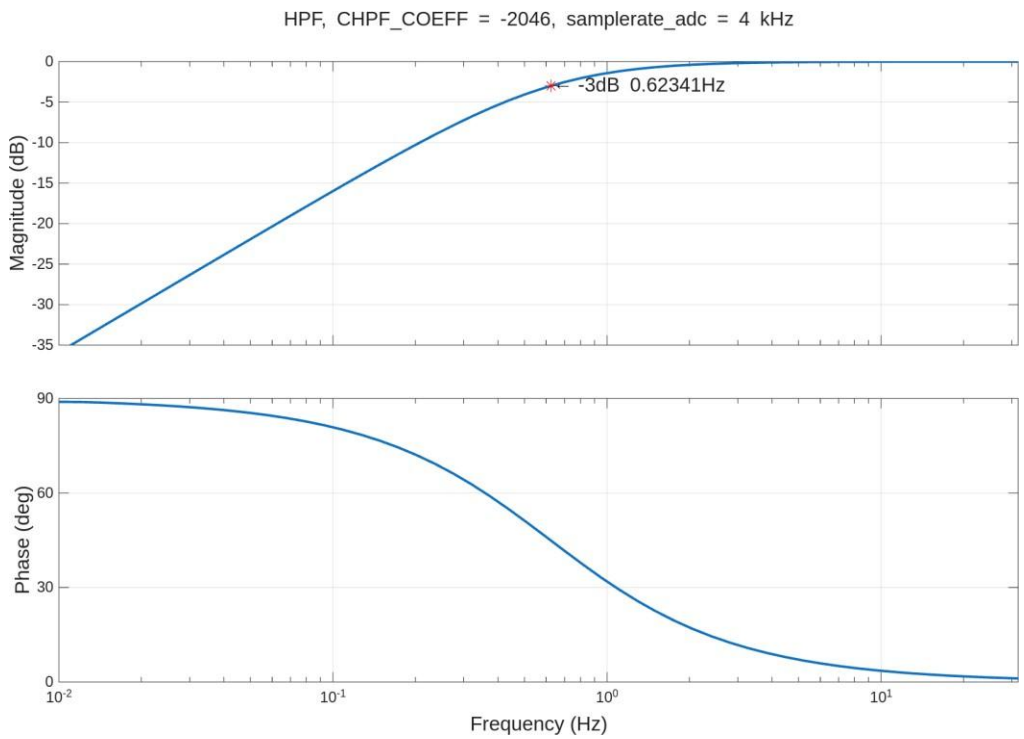


Figure 18 – Amplitude and phase frequency responses of the HPF filter with default settings at 4 kHz

– for the LPF filter

$$H(z) = \frac{1}{2} \cdot \frac{1+c+(c+1)\cdot z^{-1}}{1+c\cdot z^{-1}}, \qquad (2)$$

where c is the two's-complement value of the CLRMS_COEFF field of the ADCUI_FILTERCFG register divided by 2^{11}.

For the reset value of CLRMS_COEFF, c = 803 h / 2^{11} = −0.998535.

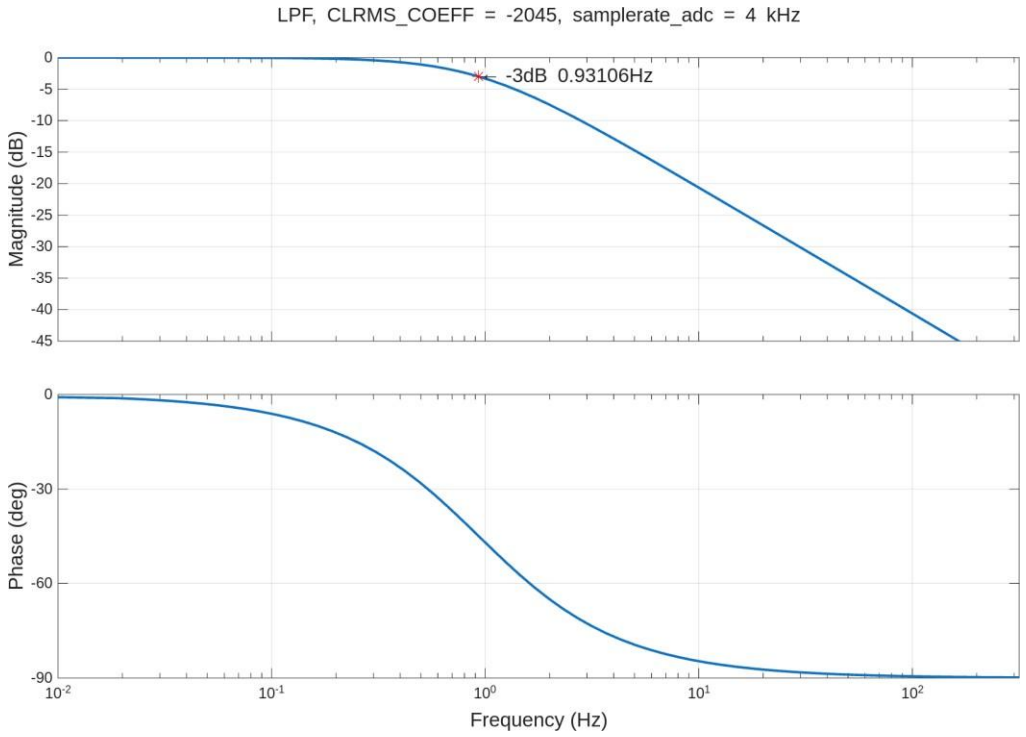


Figure 19 – Amplitude and phase frequency responses of the LPF filter with default settings at 4 kHz

– for the SINC3 filter – a third-order low-pass FIR filter

$$H(z) = \left[\frac{\left(1-z^{-OSR}\right)\cdot z^{-2}}{OSR\cdot\left(1-z^{-1}\right)}\right]^3. \quad (3)$$

It is possible to use external delta-sigma modulators instead of the internal ADCs. External delta-sigma modulators are used by the ADCUI block instead of the internal ADCs when the port D pins are configured for the corresponding SD_Vx_EXT and SD_Ix_EXT functions.

List of parameters calculated by the ADC block:

– Seven independent ADC channels (four current channels and three voltage channels) with an output sampling rate that depends on the ADC clock frequency, which is configured using the CLC_ADC_CFG field of the ADCUI_CLKPHASE1 register; the output sampling rate is configured using the OSR_CONF field of the ADCUI_CTRL1 register. These channels form three blocks for measuring the parameters of each phase F0–F2. The FIFO depth for each $\Delta\sum$ ADC channel is eight samples. Samples in the FIFO buffers (registers F0IxDAT, FxVDAT, and FxIDAT) of each channel are represented in two's-complement format, that is, they are signed values;

– In the F0 block, an option is implemented to select current channel I0 or I3 for subsequent power calculations. Apart from this function, the F0–F2 blocks are otherwise identical;

– All ADC channels have independent slope calibration coefficients;

– Each current channel has an independent integrator. The integrator must be used only when the voltage lags the current in the Fx block by 90 degrees in phase (due to the applied circuit design); when enabled, phase correction is performed. If the current and voltage in the Fx block are in phase, the integrator is not required;

– In each ADC block (F0–F2), the signal period on the voltage channel is calculated independently. The number of periods over which this value is calculated can be set to 1/2/4/8/16/32/64/128 periods;

– Each block includes detection of loss of the periodic signal on the voltage channel;

– Each block checks for voltage sag below a specified level, as well as exceedance of a specified limit in the current and voltage channels;

– It is possible to adjust the phases of the signals in the voltage channels with an accuracy of up to 0.005%;

– RMS values and squared RMS values of currents and voltages are calculated, along with their independent calibration;

– When calculating active and reactive energy, the accumulated energy over a period is stored in separate registers (for positive and negative energy);

– Apparent power and apparent energy are calculated;

- The phase shift relative to phase 0 is calculated;
- All seven channels are available for measuring the DC component of the signal. For voltage channels, the ADCUI_FxMD0 register, FxVSEL field, must be set to “00” for voltage samples (default setting), whereas for current channels I0–I3, the ADCUI_MSC register, FxISEL_HPF field, must be set to “1”.

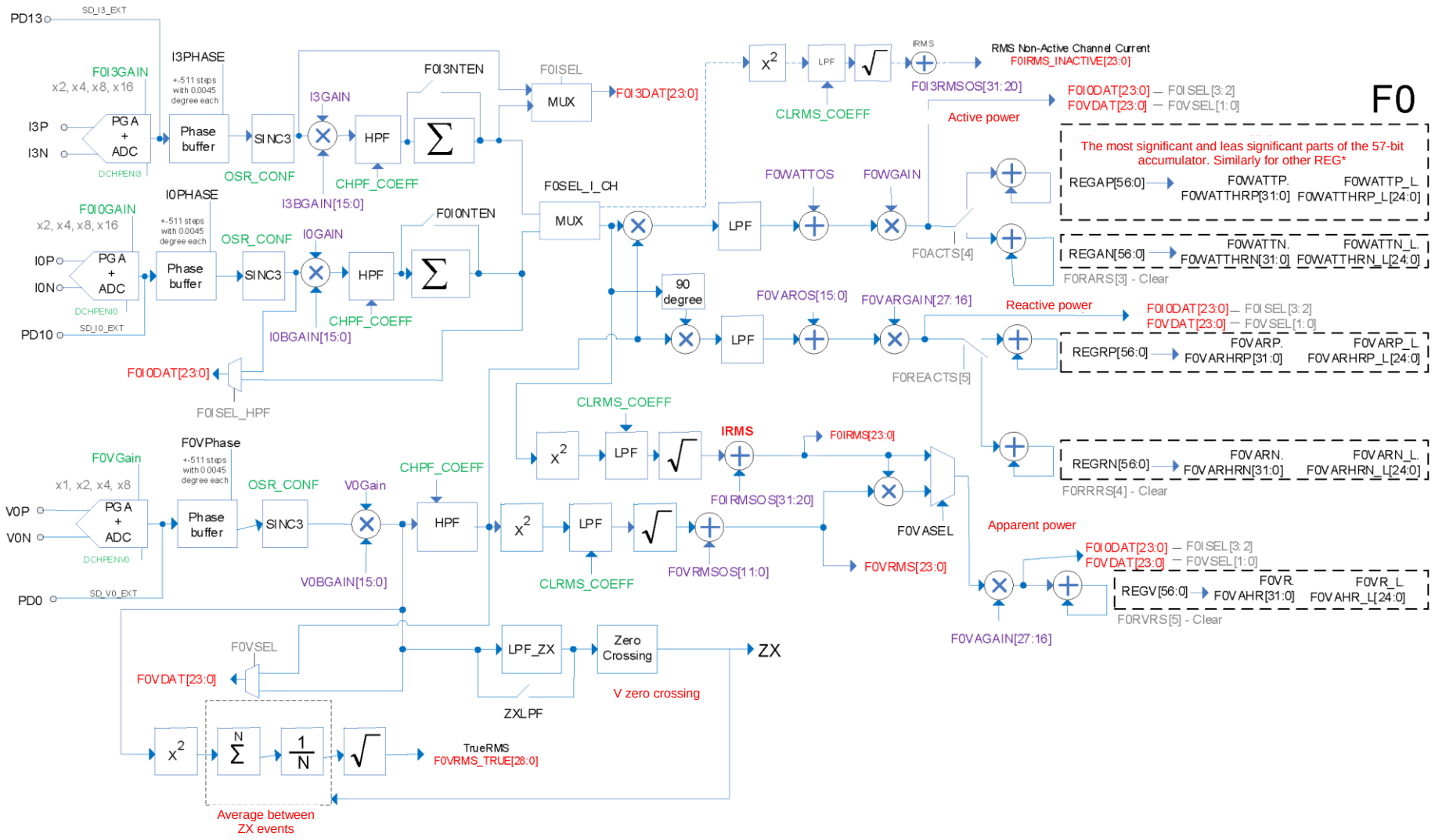


Figure 20 – Block diagram of F0

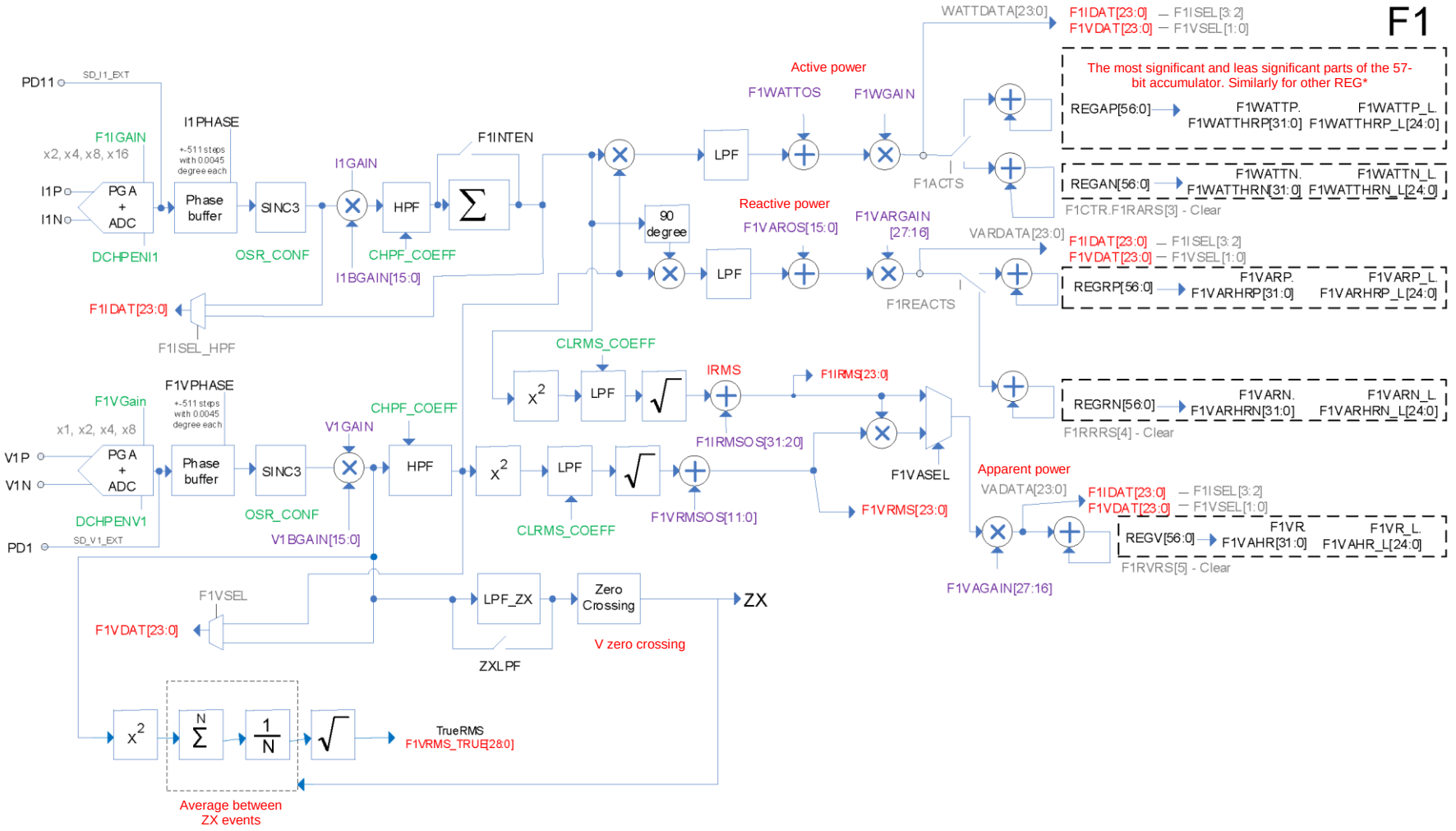


Figure 21 – Block diagram of F1

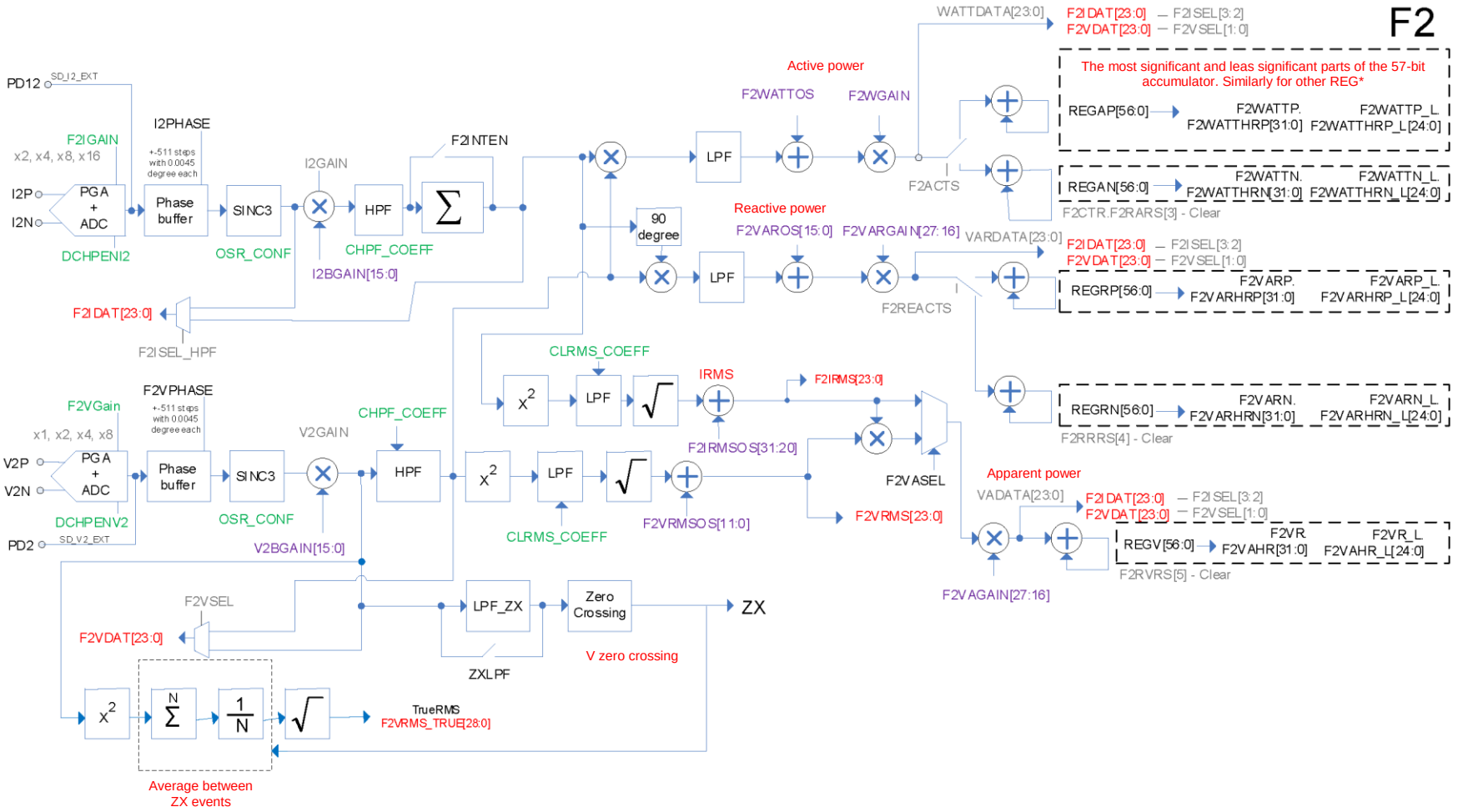


Figure 22 – Block diagram of F2

13.1. **Description of the control registers of the seven-channel ADC block**

Table 40 – Control registers of the seven-channel ADC block

Base Address	Name	Description
0x5006_8000	ADCUI	Voltage/current ADC controller
Offset		
0x000	ADCUI_CTRL1	General control of the ADC controller
0x004	ADCUI_CTRL2	
0x008	ADCUI_CTRL3	
0x00C	ADCUI_F0CTR	Control for channel F0
0x010	ADCUI_F0WC	Active energy calculation control for channel F0
0x014	ADCUI_F0WATTP	Upper part of the positive active energy accumulator for channel F0
0x018	ADCUI_F0WATTN	Upper part of the negative active energy accumulator for channel F0
0x01C	ADCUI_F0VC	Reactive energy calculation control for channel F0
0x020	ADCUI_F0VARP	Upper part of the positive reactive energy accumulator for channel F0
0x024	ADCUI_F0VARN	Upper part of the negative reactive energy accumulator for channel F0
0x028	ADCUI_F0AC	Apparent energy calculation control for channel F0
0x02C	ADCUI_F0VR	Upper part of the apparent energy accumulator for channel F0
0x030	ADCUI_F0MD0	Parameter set 0 for channel F0
0x034	ADCUI_F0MD1	Parameter set 1 for channel F0
0x038	ADCUI_F0VPEAK	Peak value in the voltage channel of F0
0x03C	ADCUI_F0IPEAK	Peak value in the current channel of F0
0x040	ADCUI_F0VDAT	FIFO sample buffer of the voltage channel of block F0
0x044	ADCUI_F0I0DAT	FIFO sample buffer of the current channel (I0) of block F0
0x048	ADCUI_F0I3DAT	FIFO sample buffer of the current channel (I3) of block F0
0x04C	ADCUI_F0VRMS	RMS voltage value of channel F0
0x050	ADCUI_F0VRMS2	Squared RMS value of the voltage channel of F0
0x054	ADCUI_F0IRMS	RMS current value of channel F0
0x058	ADCUI_F0IRMS2	Squared RMS value of the current channel F0
0x05C	ADCUI_F0STAT	Channel F0 status
0x060	ADCUI_F0MASK	Channel F0 interrupt mask
0x064	ADCUI_F1CTR	Control for channel F1
0x068	ADCUI_F1WC	Active energy calculation control for channel F1
0x06C	ADCUI_F1WATTP	Upper part of the positive active energy accumulator for channel F1
0x070	ADCUI_F1WATTN	Upper part of the negative active energy accumulator for channel F1
0x074	ADCUI_F1VC	Reactive energy calculation control for channel F1

Base Address	Name	Description
0x5006_8000	ADCUI	Voltage/current ADC controller
Offset		
0x078	ADCUI_F1VARP	Upper part of the positive reactive energy accumulator for channel F1
0x07C	ADCUI_F1VARN	Upper part of the negative reactive energy accumulator for channel F1
0x080	ADCUI_F1AC	Apparent energy calculation control for channel F1
0x084	ADCUI_F1VR	Upper part of the apparent energy accumulator for channel F1
0x088	ADCUI_F1MD0	Parameter set 0 for channel F1
0x08C	ADCUI_F1MD1	Parameter set 1 for channel F1
0x090	ADCUI_F1MD2	Parameter set 2 for channel F1
0x094	ADCUI_F1VPEAK	Peak value in the voltage channel of F1
0x098	ADCUI_F1IPEAK	Peak value in the current channel of F1
0x09C	ADCUI_F1VDAT	FIFO sample buffer of the voltage channel of block F1
0x0A0	ADCUI_F1IDAT	FIFO sample buffer of the current channel (I) of block F1
0x0A4	ADCUI_F1VRMS	RMS voltage value of channel F1
0x0A8	ADCUI_F1VRMS2	Squared RMS value of the voltage channel F1
0x0AC	ADCUI_F1IRMS	RMS current value of channel F1
0x0B0	ADCUI_F1IRMS2	Squared RMS value of the current channel F1
0x0B4	ADCUI_F1STAT	Channel F1 status
0x0B8	ADCUI_F1MASK	Channel F1 interrupt mask
0x0BC	ADCUI_F2CTR	Control for channel F2
0x0C0	ADCUI_F2WC	Active energy calculation control for channel F2
0x0C4	ADCUI_F2WATTP	Upper part of the positive active energy accumulator for channel F2
0x0C8	ADCUI_F2WATTN	Upper part of the negative active energy accumulator for channel F2
0x0CC	ADCUI_F2VC	Reactive energy calculation control for channel F2
0x0D0	ADCUI_F2VARP	Upper part of the positive reactive energy accumulator for channel F2
0x0D4	ADCUI_F2VARN	Upper part of the negative reactive energy accumulator for channel F2
0x0D8	ADCUI_F2AC	Apparent energy calculation control for channel F2
0x0DC	ADCUI_F2VR	Upper part of the apparent energy accumulator for channel F2
0x0E0	ADCUI_F2MD0	Parameter set 0 for channel F2
0x0E4	ADCUI_F2MD1	Parameter set 1 for channel F2
0x0E8	ADCUI_F2MD2	Parameter set 2 for channel F2
0x0EC	ADCUI_F2VPEAK	Peak value in the voltage channel of F2
0x0F0	ADCUI_F2IPEAK	Peak value in the current channel of F2
0x0F4	ADCUI_F2VDAT	FIFO sample buffer of the voltage channel of block F2
0x0F8	ADCUI_F2IDAT	FIFO sample buffer of the current channel (I) of block F2

Base Address	Name	Description
0x5006_8000	ADCUI	Voltage/current ADC controller
Offset		
0x0FC	ADCUI_F2VRMS	RMS voltage value of channel F2
0x100	ADCUI_F2VRMS2	Squared RMS voltage value channel F2
0x104	ADCUI_F2IRMS	RMS current value channel F2
0x108	ADCUI_F2IRMS2	Squared RMS in the current channel F2
0x10C	ADCUI_F2STAT	Channel F2 status
0x110	ADCUI_F2MASK	Channel F2 interrupt mask
0x114	ADCUI_CCAL1	Calibration register for current (I0) and voltage (V0) channels in block F0
0x118	ADCUI_CCAL2	Calibration register for current and voltage channels in block F1
0x11C	ADCUI_CCAL3	Calibration register for current and voltage channels in block F2
0x120	ADCUI_CCAL4	Calibration register for the current channel (I3) in block F0
0x124	ADCUI_F0IRMS_INACTIVE	RMS current value of the inactive channel F0
0x128	ADCUI_F0WATTP_L	Least significant part of the positive active energy accumulator in channel F0
0x12C	ADCUI_F0WATTN_L	Least significant part of the negative active energy accumulator in channel F0
0x130	ADCUI_F0VARP_L	Least significant part of the positive reactive energy accumulator in channel F0
0x134	ADCUI_F0VARN_L	Least significant part of the negative reactive energy accumulator in channel F0
0x138	ADCUI_F0VR_L	Least significant part of the apparent energy accumulator in channel F0
0x13C	ADCUI_F1WATTP_L	Least significant part of the positive active energy accumulator in channel F1
0x140	ADCUI_F1WATTN_L	Least significant part of the negative active energy accumulator in channel F1
0x144	ADCUI_F1VARP_L	Least significant part of the positive reactive energy accumulator in channel F1
0x148	ADCUI_F1VARN_L	Least significant part of the negative reactive energy accumulator in channel F1
0x14C	ADCUI_F1VR_L	Least significant part of the apparent energy accumulator in channel F1
0x150	ADCUI_F2WATTP_L	Least significant part of the positive active energy accumulator in channel F2
0x154	ADCUI_F2WATTN_L	Least significant part of the negative active energy accumulator in channel F2
0x158	ADCUI_F2VARP_L	Least significant part of the positive reactive energy accumulator in channel F2
0x15C	ADCUI_F2VARN_L	Least significant part of the negative reactive energy accumulator in channel F2

Base Address	Name	Description
0x5006_8000	ADCUI	Voltage/current ADC controller
Offset		
0x160	ADCUI_F2VR_L	Least significant part of the apparent energy accumulator in channel F2
0x164	ADCUI_LOAD	General control of the “no-load” mode
0x168	ADCUI_F0VRMS_TRUE	RMS voltage value over one period for channel F0
0x16C	ADCUI_F1VRMS_TRUE	RMS voltage value over one period for channel F1
0x170	ADCUI_F2VRMS_TRUE	RMS voltage value over one period for channel F2
0x174	ADCUI_MSC	Enable generation of DMA requests and others
0x178	ADCUI_FILTERCFG	Filter configuration control
0x17C	ADCUI_CLKPHASE1	Current channel phase shift control
0x180	ADCUI_CLKPHASE2	Current channel phase shift control

13.1.1. **ADCUI_CTRL1**

Table 41 – ADCUI_CTRL1 Register

Number	31, 30	29	28	27	26, 25	24, 23
Access	R/W	R/W	R/W	R/W	-	R/W
Reset	00	0	0	0	-	00
	OSR_CONF	tst_in[4]	RESET_DIG	ZXRMS	-	tst_in[3:2]

Number	22	21	20	19	18, 17	16, 15
Access	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0
	tst_in[1]	tst_in[0]	VREF_SEL	FREQSEL	VANOLOAD	VARNOLOAD

Number	14	13, 12	11…9	8	7	6
Access	-	R/W	R/W	R/W	R/W	R/W
Reset	-	0	000	0	0	0
	-	APNOLOAD	PER_LENGTH	ZXLPF	RESOL	I3EN

Number	5	4	3	2	1	0
Access	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0
	V2EN	I2EN	V1EN	I1EN	V0EN	I0EN

Table 42 – Description of the ADCUI_CTRL1 register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31, 30	OSR_CONF*	Selection of the oversampling ratio: 00 – OSR = 1024; 01 – OSR = 512; 10 – OSR = 256; 11 – reserved. Note: These ratios apply to the ADC block clock frequency f_{C_ADCD}, which may be 1.024, 2.048, or 4.096 MHz, depending on the setting of the CLC_ADC_CFG field in the ADCUI_CLKPHASE1 register.
29	tst_in[4]	Test output (always write zero).
28	RESET_DIG	Reset of the digital part of the ADC blocks: 0 – no reset; 1 – the digital part is under global reset.
27	ZXRMS**	Control of updates of registers containing RMS values: 0 – continuous update; 1 – update on voltage zero crossing. Note: This setting does not apply to the calculation of True RMS values in the corresponding registers.
26, 25	-	Reserved
24, 23	tst_in[3:2]	Test outputs (always write zero).
22	tst_in[1]	Test output (always write zero).
21	tst_in[0]	Test output (always write zero).
20	VREF_SEL	Selection of the ADC reference voltage: 0 – external reference voltage from the VREF pin; 1 – internal reference voltage VREF_INT 2.4 V (Figure 64). When VREF_SEL == 1 is selected, the internal reference voltage source VREF_INT 2.4 V is enabled, and the VREF_INT 2.4 V output voltage is driven to the VREF pin.
19	FREQSEL	Enable calculation of the period in the voltage channels: 1 – enabled; 0 – the last calculated value is retained.
18, 17	VANOLOAD	“No-load” mode when calculating apparent energy: 00 – all calculated energy is accumulated (both positive and negative energy); 01 – energy below 0.0305% of the full-scale quantum energy is ignored; 10 – energy below 0.0152% of the full-scale quantum energy is ignored; 11 – energy below 0.0076% of the full-scale quantum energy is ignored. Quantum energy is the energy over one period of the input signal (20 ms), as determined by 0-crossing detector events: – at a sampling rate of 4 kHz – the sum of 80 samples; – at a sampling rate of 8 kHz – the sum of 160 samples; – at a sampling rate of 16 kHz – the sum of 320 samples

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
16, 15	VARNOLOAD	"No-load" mode when calculating reactive energy: 00 – all calculated energy is accumulated (both positive and negative energy); 01 – energy below 0.0305% of the full-scale quantum energy is ignored; 10 – energy below 0.0152% of the full-scale quantum energy is ignored; 11 – energy below 0.0076% of the full-scale quantum energy is ignored. Quantum energy is the energy over one period of the input signal (20 ms), as determined by 0-crossing detector events: – at a sampling rate of 4 kHz – the sum of 80 samples; – at a sampling rate of 8 kHz – the sum of 160 samples; – at a sampling rate of 16 kHz – the sum of 320 samples
14	-	Reserved
13, 12	APNOLOAD	"No-load" mode when calculating active energy: 00 – all calculated energy is accumulated (both positive and negative energy); 01 – energy below 0.0305% of the full-scale quantum energy is ignored; 10 – energy below 0.0152% of the full-scale quantum energy is ignored; 11 – energy below 0.0076% of the full-scale quantum energy is ignored. Quantum energy is the energy over one period of the input signal (20 ms), as determined by 0-crossing detector events: – at a sampling rate of 4 kHz – the sum of 80 samples; – at a sampling rate of 8 kHz – the sum of 160 samples; – at a sampling rate of 16 kHz – the sum of 320 samples
11...9	PER_LENGTH	Range for period and phase-shift calculation: 000 – over 1 period; 001 – over 2 periods; … 111 – over 128 periods. Affects the calculation of F1PER_FREQ, F2PER_FREQ, F0PER_FREQ, F1PHASE, and F2PHASE.
8	ZXLPF	Disabling the low-frequency filter before the 0-crossing detector in the voltage channels: 0 – filter enabled; 1 – filter disabled
7	RESOL	Output data width: 0 – 16 bits; 1 – 24 bits The most significant bit in the data register is bit 15 or bit 23. No data register shifting is required.
6	I3EN	Enable operation of channel I3: 0 – channel disabled; 1 – channel enabled***
5	V2EN	Enable operation of channel V2: 0 – channel disabled; 1 – channel enabled***

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
4	I2EN	Enable operation of channel I2: 0 – channel disabled; 1 – channel enabled***
3	V1EN	Enable operation of channel V1: 0 – channel disabled; 1 – channel enabled***
2	I1EN	Enable operation of channel I1: 0 – channel disabled; 1 – channel enabled***
1	V0EN	Enable operation of channel V0: 0 – channel disabled; 1 – channel enabled***
0	I0EN	Enable operation of channel I0: 0 – channel disabled; 1 – channel enabled***

* When the sampling rate is changed, the settings of the digital HPF and LPF filters must be adjusted accordingly to maintain their cutoff frequencies. It should also be taken into account that doubling the sampling rate results in a reduction of SNR by at least 3 dB in the band from 0 Hz to half the sampling rate.

** Since the RMS values of current and voltage are updated simultaneously, the current value depends on the angle between voltage and current. This dependence is shown in Figure 17. Based on these data, the true current value can be corrected.

*** When any channel in block Fx is enabled, the FIFOs of all other channels (FIFO FxVDAT, FxIDAT) in block Fx are also enabled.

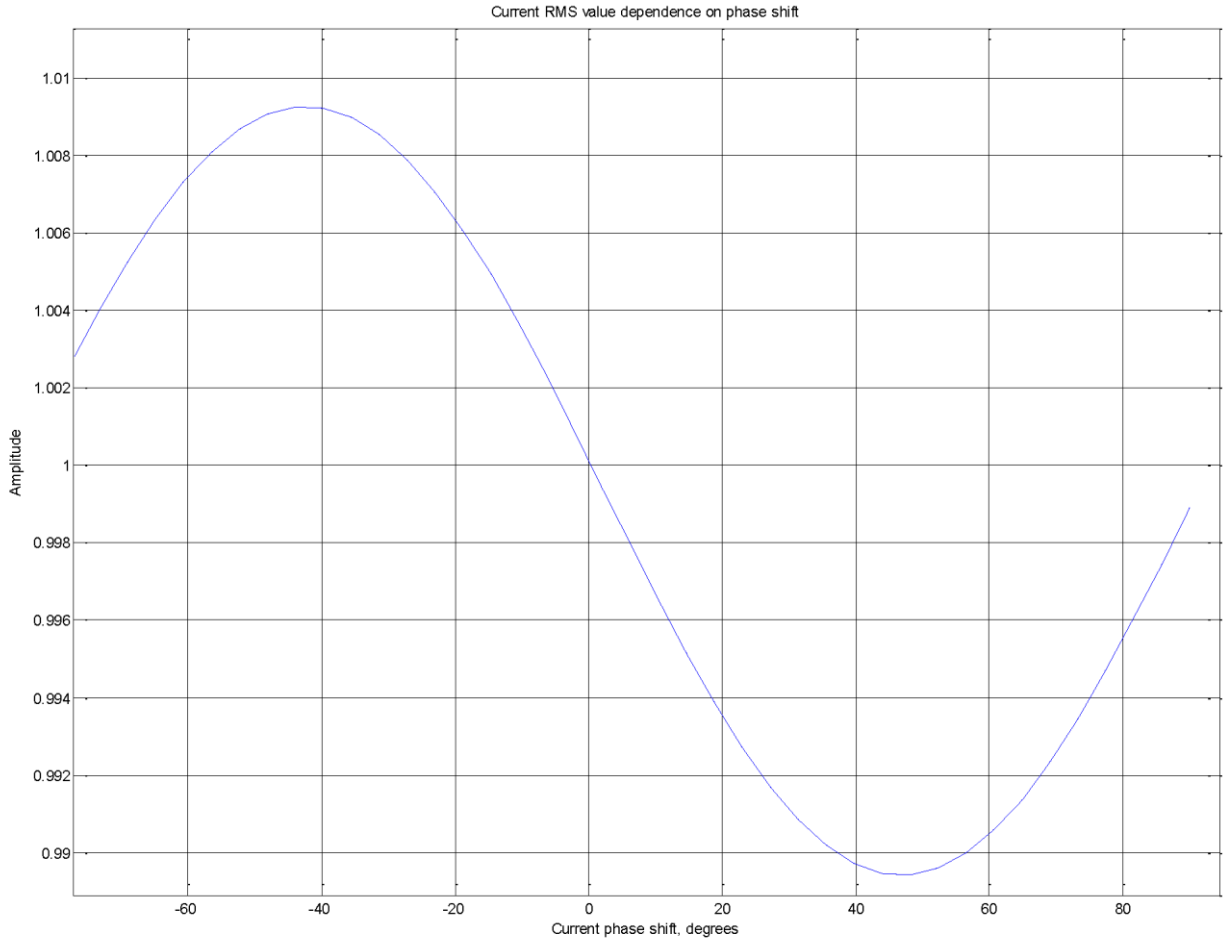


Figure 23 – Dependence of RMS on the phase shift of the input signal

13.1.2. **ADCUI_CTRL2**

Table 43 – ADCUI_CTRL2 Register

Number	31	30	29	28	27	26
Access	–	R/W	R/W	R/W	R/W	R/W
Reset	–	0	0	0	0	0
	–	DCHPENI3	DCHPENI2	DCHPENI1	DCHPENI0	DCHPENV2

Number	25	24	23…16	15…0
Access	R/W	R/W	R/W	R/W
Reset	0	0	FFh	FFFFh
	DCHPENV1	DCHPENV0	SAGCYC	SAGLVL

Table 44 – Description of the ADCUI_CTRL2 register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31	–	Reserved
30	DCHPENI3	0 – chopper for channel I3 disabled; 1 – chopper for channel I3 enabled*
29	DCHPENI2	0 – chopper for channel I2 disabled; 1 – chopper for channel I2 enabled*

28	DCHPENI1	0 – chopper for channel I1 disabled; 1 – chopper for channel I1 enabled*
27	DCHPENI0	0 – chopper for channel I0 disabled; 1 – chopper for channel I0 enabled*
26	DCHPENV2	0 – chopper for channel V2 disabled; 1 – chopper for channel V2 enabled*
25	DCHPENV1	0 – chopper for channel V1 disabled; 1 – chopper for channel V1 enabled*
24	DCHPENV0	0 – chopper for channel V0 disabled; 1 – chopper for channel V0 enabled*
23...16	SAGCYC	Number of voltage half-cycles used to calculate voltage sag
15...0	SAGLVL	Permitted voltage sag level
* Recommended for use when measuring DC signals. When using an external delta-sigma modulator, the chopper must be disabled		

13.1.3. **ADCUI_CTRL3**

Table 45 – ADCUI_CTRL3 Register

Number	31…12	11…0
Access	-	R/W
Reset	-	0FFh
	-	ZXTOUT

Table 46 – Description of the ADCUI_CTRL3 register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31..12	-	Reserved
11..0	ZXTOUT	Timeout counter value for the absence of input-signal zero crossings in channels V0, V1, and V2, which set the F0ZXTOF, F1ZXTOF, and F2ZXTOF flags. The value is expressed in periods of the ADC oversampling frequency of 4/8/16 kHz (depending on OSR_CONF and CLC_ADC_CFG). The reset value is ~64 ms.

13.1.4. **ADCUI_F0CTR**

Table 45 – ADCUI_F0CTR Register

Number	31…20	19, 18	17…10	9, 8	7, 6
Access	R/W	R/W	R/W	R/W	R/W
Reset	0	00	00	0	0
	F0IRMSOS	F0I3GAIN	F0VPHASE	F0VGAIN	F0I0GAIN

Number	5	4	3	2	1	0
Access	WO	WO	WO	R/W	R/W	R/W
Reset	0	0	0	0	0	0
	F0RVRS	F0RRRS	F0RARS	F0VASEL	F0I3NTEN	F0I0NTEN

Table 46 – Description of the ADCUI_F0CTR register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...20	F0IRMSOS	Offset calibration for the RMS calculator of current channel I0. Represented as signed values in two's-complement binary format; when applied in the calculator, the value is shifted left by 4 bits. Example: 007: IRMSOS = 7 · 16 = 112 FF9: IRMSOS = -7 · 16 = -112
19, 18	F0I3GAIN	Preamplifier in current channel I3: 00 – 6 dB (x2); 01 – 12 dB (x4); 10 – 18 dB (x8); 11 – 24 dB (x16)
17...10	F0VPHASE[9:2]	Phase shift of the voltage channel relative to the current channel, encoded in two's-complement format. From −511*(ADC clock period) to +511*(ADC clock period). "0000000000" corresponds to an in-phase signal relative to the current. When DCHPENV0 = 1, it is recommended to adjust the phase shift in steps of 32. The least significant bits F0VPHASE[1:0] are located in ADCUI_MSC.
9, 8	F0V0GAIN	Preamplifier in the voltage channel: 00 – 0 dB (x1); 01 – 6 dB (x2); 10 – 12 dB (x4); 11 – 18 dB (x8)
7, 6	F0I0GAIN	Preamplifier in current channel 0: 00 – 6 dB (x2); 01 – 12 dB (x4); 10 – 18 dB (x8); 11 – 24 dB (x16)
5	F0RVRS	Writing to this register resets the transmitted apparent energy counter.
4	F0RRRS	Writing to this register resets the transmitted reactive energy counter.

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
3	F0RARS	Writing to this register resets the transmitted active energy counter.
2	F0VASEL	Selection of the signal source to be stored in the apparent energy register: 0 – apparent energy; 1 – RMS current value
1	F0I3NTEN	Disabling the integrator in current channel I3: 0 – integrator enabled; 1 – integrator disabled. Correct operation of the integrator is guaranteed when the CHPF_COEFF value provides an HPF cutoff frequency of 1 Hz at the specified sampling rate. For example, if the sampling rate is 4 kHz, the CHPF_COEFF value must be 0x802 (default value).
0	F0I0NTEN	Disabling the integrator in current channel 0: 0 – integrator enabled; 1 – integrator disabled. Correct operation of the integrator is guaranteed when the CHPF_COEFF value provides an HPF cutoff frequency of 1 Hz at the specified sampling rate. For example, if the sampling rate is 4 kHz, the CHPF_COEFF value must be 0x802 (default value).

13.1.5. **ADCUI_F0WC**

Table 47 – ADCUI_F0WC Register

Number	31…28	27…16	15…0
Access	-	R/W	R/W
Reset	-	000 h	0000 h
	-	F0WGAIN	F0WATTOS

Table 48 – Description of the ADCUI_F0WC register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...28	-	Reserved
27…16	F0WGAIN	Active power channel gain calibration
15…0	F0WATTOS	Active power channel offset calibration

13.1.6. **ADCUI_F0WATTP**

Table 49 – ADCUI_F0WATTP Register

Number	31…0
Access	RO
Reset	
	F0WATTHRP

Table 50 – Description of the ADCUI_F0WATTP register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…0	F0WATTHRP	The most significant 32 bits of the internal 57-bit accumulator of positive active energy

13.1.7. **ADCUI_F0WATTN**

Table 51 – ADCUI_F0WATTN Register

Number	31…0
Access	RO
Reset	
	F0WATTHRN

Table 52 – Description of the ADCUI_F0WATTN register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...0	F0WATTHRN	The most significant 32 bits of the internal 57-bit accumulator of negative active energy

13.1.8. **ADCUI_F0VC**

Table 53 – ADCUI_F0VC Register

Number	31…28	27…16	15…0
Access	-	R/W	R/W
Reset	-	000 h	0000 h
	-	F0VARGAIN	F0VAROS

Table 54 – Description of the ADCUI_F0VC register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...28	-	Reserved
27...16	F0VARGAIN	Reactive power channel gain calibration
15...0	F0VAROS	Reactive power channel offset calibration

13.1.9. **ADCUI_F0VARP**

Table 55 – ADCUI_F0VARP Register

Number	31..0
Access	RO
Reset	
	F0VARHRP

Table 56 – Description of the ADCUI_F0VARP register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...0	F0VARHRP	The most significant 32 bits of the internal 57-bit accumulator of positive reactive energy

13.1.10. **ADCUI_F0VARN**

Table 57 – ADCUI_F0VARN Register

Number	31..0
Access	RO
Reset	
	F0VARHRN

Table 58 – Description of the ADCUI_F0VARN register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...0	F0VARHRN	The most significant 32 bits of the internal 57-bit accumulator of negative reactive energy

13.1.11.**ADCUI_F0AC**

Table 59 – ADCUI_F0AC Register

Number	31…28	27…16	15…13	12	11…0
Access	-	R/W	-	R/W	R/W
Reset	-	000 h	-	0	000 h
	-	F0VAGAIN	-	F0I3SEL	F0VRMSOS

Table 60 – Description of the ADCUI_F0AC register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...28	-	Reserved
27...16	F0VAGAIN	Apparent power channel gain calibration
15...13	-	Reserved
12	F0I3SEL	Selection of the signal source for the FIFO of current channel I3 (ADCUI_F0I3DAT register): 0 – current samples after the high-pass filter; 1 – current samples before the high-pass filter

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
11...0	F0VRMSOS	Calibration of the voltage RMS calculator. Represented as signed values in two's-complement binary format; when applied in the calculator, the value is shifted left by 4 bits. Example: 007: VRMSOS = 7 · 16 = 112 FF9: VRMSOS = -7 · 16 = -112

13.1.12. **ADCUI_F0VR**

Table 61 – ADCUI_F0VR Register

Number	31…0
Access	RO
Reset	
	F0VAHR

Table 62 – Description of the ADCUI_F0VR register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…0	F0VAHR	The most significant 32 bits of the internal 57-bit accumulator of apparent energy

13.1.13. **ADCUI_F0MD0**

Table 63 – ADCUI_F0MD0 Register

Number	31, 30	29…9	8	7
Access	R/W	RO	R/W	R/W
Reset	01		0	0
	F0SEL_I_CH	F0PER_FREQ	I3GAIN	V0GAIN

Number	6	5	4	3, 2	1, 0
Access	R/W	R	R	R/W	R/W
Reset	0	0	0	00	00
	I0GAIN	F0REACTS	F0ACTS	F0ISEL	F0VSEL

Table 64 – Description of the ADCUI_F0MD0 register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31, 30	F0SEL_I_CH	Selection of the active current channel for power calculation: 00, 11 – invalid value; 01 – active channel I0; 10 – active channel I3
29...9	F0PER_FREQ	Duration of the input signal period in voltage channel V0. The value is expressed in ticks of the internal counter, averaged over the number of input-frequency periods defined by PER_LENGTH. The input signal period, based on the counter value, is calculated using the following formula: $$Period = \frac{F0PER\_FREQ \cdot 8}{ADCUI\_CLK}\ c.$$ For example, for a 50 Hz input signal, when the sigma-delta ADC block is clocked at 8.192 MHz (the recommended operating frequency), the F0PER_FREQ counter value is 20480; in this case, the value $$Period = \frac{20480 \cdot 8}{ADCUI\_CLK} = 0{,}02\,c$$
8	I3GAIN	Digital gain in current channel I3: 0 – no gain; 1 – +6 dB gain (result shifted left by 1).
7	V0GAIN	Digital gain in voltage channel V0: 0 – no gain; 1 – +6 dB gain (result shifted left by 1).
6	I0GAIN	Digital gain in channel I0: 0 – no gain; 1 – +6 dB gain (result shifted left by 1).
5	F0REACTS	Sign of reactive energy in the last period.
4	F0ACTS	Sign of active energy in the last period.
3, 2	F0ISEL	Selection of the signal source for the FIFO of current channel I0 (ADCUI_F0IDAT register): 00 – current samples (after or before the HPF, see F0ISEL_HPF); 01 – active power samples; 10 – reactive power samples; 11 – apparent power samples.
1, 0	F0VSEL	Selection of the signal source for the FIFO of voltage channel V0 (ADCUI_F0VDAT register): 00 – voltage samples before the high-pass filter; 01 – active power samples; 10 – reactive power samples; 11 – voltage samples after the high-pass filter.

13.1.14. **ADCUI_F0MD1**

Table 65 – ADCUI_F0MD1 Register

Number	31…16	15…0
Access	R/W	R/W
Reset	0000 h	0000 h
	F0VPKLVL	F0IPKLVL

Table 66 – Description of the ADCUI_F0MD1 register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...16	F0VPKLVL	Maximum permitted voltage level. The threshold value set in this register is compared with the most significant 16 bits at the output of the HPF filter of voltage channel V0. When the threshold is exceeded, the F0PEAKVF flag is set.
15...0	F0IPKLVL	Maximum permitted current level. The threshold value set in this register is compared with the most significant 16 bits at the output of the HPF filter of current channel I0. When the threshold is exceeded, the F0PEAKIF flag is set.

13.1.15. **ADCUI_F0VPEAK**

Table 67 – ADCUI_F0VPEAK Register

Number	31…24	23…0
Access		R/W
Reset		000000 h
	-	F0VPEAK

Table 68 – Description of the ADCUI_F0VPEAK register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...24	-	Reserved
23...0	F0VPEAK	Peak voltage value. Cleared by reading from the register. Writing to this register resets the peak value.

13.1.16. **ADCUI_F0IPEAK**

Table 69 – ADCUI_F0IPEAK Register

Number	31…24	23…0
Access		R/W
Reset		000000 h
	-	F0IPEAK

Table 70 – Description of the ADCUI_F0IPEAK register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...24	-	Reserved
23...0	F0IPEAK	Peak current value. Cleared by reading from the register. Writing to this register resets the peak value.

13.1.17. **ADCUI_F0VDAT**

Table 71 – ADCUI_F0VDAT Register

Number	31…24	23…0
Access	RO	RO
Reset	00 h	000000 h
	00 h	F0VDAT

Table 72 – Description of the ADCUI_F0VDAT register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...24	-	Reserved
23...0	F0VDAT	FIFO of voltage samples (or of one of the power values). The FIFO is enabled when any of the bits I3EN, V0EN, or I0EN is set.

13.1.18. **ADCUI_F0I0DAT**

Table 73 – ADCUI_F0I0DAT Register

Number	31…24	23…0
Access	RO	RO
Reset	00 h	000000 h
	00 h	F0I0DAT

Table 74 – Description of the ADCUI_F0I0DAT register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…24	-	Reserved
23...0	F0I0DAT	FIFO of current 0 samples (or of one of the power values). The FIFO is enabled when any of the bits I3EN, V0EN, or I0EN is set.

13.1.19. **ADCUI_F0I3DAT**

Table 75 – ADCUI_F0I3DAT Register

Number	31…24	23…0
Access	RO	RO
Reset	00 h	000000 h
	00 h	F0I3DAT

Table 76 – Description of the ADCUI_F0I3DAT register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...24	-	Reserved
23...0	F0I3DAT	FIFO of current samples 3. The FIFO is enabled when any of the bits I3EN, V0EN, or I0EN is set.

13.1.20. **ADCUI_F0VRMS**

Table 77 – ADCUI_F0VRMS Register

Number	31…24	23…0
Access		RO
Reset		000000 h
	-	F0VRMS

Table 78 – Description of the ADCUI_F0VRMS register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...24	-	Reserved
23...0	F0VRMS	RMS voltage

13.1.21. **ADCUI_F0VRMS2**

Table 79 – ADCUI_F0VRMS2 Register

Number	31…0
Access	RO
Reset	
	F0VRMS2

Table 80 – Description of the ADCUI_F0VRMS2 register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...0	F0VRMS2	Squared RMS voltage

13.1.22. **ADCUI_F0IRMS**

Table 81 – ADCUI_F0IRMS Register

Number	31…24	23…0
Access		RO
Reset		000000 h
	-	F0IRMS

Table 82 – Description of the ADCUI_F0IRMS register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...24	-	Reserved
23...0	F0IRMS	RMS current value

13.1.23. **ADCUI_F0IRMS2**

Table 83 – ADCUI_F0IRMS2 Register

Number	31…0
Access	RO
Reset	
	F0IRMS2

Table 84 – Description of the ADCUI_F0IRMS2 register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...0	F0IRMS2	Squared RMS voltage

13.1.24. **ADCUI_F0STAT**

Table 85 – ADCUI_F0STAT Register

Number	31…27	26	25	24	23	22
Access	-	R/W	R/W	R/W	RO	RO
Reset	-		0	0	0	0
	-	F0VAROVN	F0WATTOVN	C3IF_OVR	C3IF_FLL	C3IF_EMP

Number	21	20	19	18	17	16
Access	R/W	RO	-	RO	R/W	RO
Reset	0	0	-	0	0	0
	F0ZEROCRS	F0VANLDFL	-	F0VARNLDFL	F0VARSIGN	F0APNLDFL

Number	15	14	13	12	11	10	9	8	7
Access	R/W	-	RO	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	-	0	0	0	0	0	0	0
	F0APSIGN	-	F0ICHANNEL	F0ZXTOF	F0VAOV	F0VAROVP	F0WATTOVP	F0PEAKIF	F0PEAKVF

Number	6	5	4	3	2	1	0
Access	R/W	R/W	RO	RO	R/W	RO	RO
Reset	0	0	0	0	0	0	0
	F0SAGF	F0IF_OVR	F0IF_FLL	F0IF_EMP	F0VF_OVR	F0VF_FLL	F0VF_EMP

Table 86 – Description of the ADCUI_F0STAT register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…27	-	Reserved
26	F0VAROVN	Flag indicating that an overrun of the negative reactive energy accumulator register has occurred. Writing “1” clears this flag.
25	F0WATTOVN	Flag indicating that an overrun of the negative active energy accumulator register has occurred. Writing “1” clears this flag.
24	C3IF_OVR	Flag indicating that an overrun of the F0I3DAT FIFO has occurred. Writing “1” clears this flag.
23	C3IF_FLL	Flag indicating that the F0I3DAT FIFO is full.
22	C3IF_EMP	Flag indicating that the F0I3DAT FIFO is empty.
21	F0ZEROCRS	Flag indicating that a voltage 0 crossing has occurred. Writing “1” clears this flag.
20	F0VANLDFL	Apparent power below the comparison level.
19	-	Reserved
18	F0VARNLDFL	Reactive power below the comparison level.
17	F0VARSIGN	Change of sign of reactive power. Writing “1” clears this flag.
16	F0APNLDFL	Active power below the comparison level.
15	F0APSIGN	Change of sign of active power. Writing “1” clears this flag.
14	-	Reserved
13	F0ICHANNEL	Active current channel: 0 – active channel I0; 1 – active channel I3.
12	F0ZXTOF	Flag indicating that no voltage 0 crossing occurred in channel V0 during ZXTOUT. Writing “1” clears this flag.
11	F0VAOV	Flag indicating that an overrun of the apparent energy accumulator register has occurred. Writing “1” clears this flag.
10	F0VAROVP	Flag indicating that an overrun of the positive reactive energy accumulator register has occurred. Writing “1” clears this flag.
9	F0WATTOVP	Flag indicating that an overrun of the positive active energy accumulator register has occurred. Writing “1” clears this flag.
8	F0PEAKIF	Flag indicating that a current threshold exceedance has occurred. Writing “1” clears this flag.
7	F0PEAKVF	Flag indicating that a voltage threshold exceedance has occurred. Writing “1” clears this flag.
6	F0SAGF	Flag indicating that a voltage sag has occurred. Writing “1” clears this flag.

5	F0IF_OVR	Flag indicating that an overrun of the F0IDAT FIFO has occurred. Writing “1” clears this flag.
4	F0IF_FLL	Flag indicating that the F0IDAT FIFO is full.
3	F0IF_EMP	Flag indicating that the F0IDAT FIFO is empty.
2	F0VF_OVR	Flag indicating that an overrun of the F0VDAT FIFO has occurred. Writing “1” clears this flag.
1	F0VF_FLL	Flag indicating that the F0VDAT FIFO is full.
0	F0VF_EMP	Flag indicating that the F0VDAT FIFO is empty.

13.1.25. **ADCUI_F0MASK**

Table 87 – ADCUI_F0MASK Register

Number	31…27	26	25	24	23	22
Access	-	R/W	R/W	R/W	R/W	R/W
Reset	-		0	0	0	0
	-	F0VAROVNM	F0WATTOVNM	C3IF_OVRM	C3IF_FLLM	C3IF_EMPM

Number	21	20	19	18	17	16
Access	R/W	R/W	-	R/W	R/W	R/W
Reset	0	0	-	0	0	0
	F0ZEROCRSM	F0VANLDFLM	-	F0VARNLDFLM	F0VARSIGNM	F0APNLDFLM

Number	15	14	13	12	11	10	9	8	7
Access	R/W	-	-	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	-	-	0	0	0	0	0	0
	F0APSIGNM	-	-	F0ZXTOFM	F0VAOVM	F0VAROVPM	F0WATTOVPM	F0PEAKIFM	F0PEAKVFM

Number	6	5	4	3	2	1	0
Access	R/W	R/W	RO	RO	R/W	RO	RO
Reset	0	0	0	0	0	0	0
	F0SAGFM	F0IF_OVRM	F0IF_FLLM	F0IF_EMPM	F0VF_OVRM	F0VF_FLLM	F0VF_EMPM

Table 88 – Description of the ADCUI_F0MASK register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...27	-	Reserved
26	F0VAROVNM	Mask of the F0VAROVN bit
25	F0WATTOVNM	Mask of the F0WATTOVN bit
24	C3IF_OVRM	Mask of the C3IF_OVR bit
23	C3IF_FLLM	Mask of the C3IF_FLL bit
22	C3IF_EMPM	Mask of the C3IF_EMP bit
21	F0ZEROCRSM	Mask of the F0ZEROCRS bit
20	F0VANLDFLM	Mask of the F0VANLDFL bit
19	-	Reserved
18	F0VARNLDFLM	Mask of the F0VARNLDFL bit
17	F0VARSIGNM	Mask of the F0VARSIGN bit
16	F0APNLDFLM	Mask of the F0APNLDFL bit
15	F0APSIGNM	Mask of the F0APSIGNM bit
14	-	Reserved
13	-	Reserved
12	F0ZXTOFM	Mask of the F0ZXTOF bit
11	F0VAOVM	Mask of the F0VAOV bit
10	F0VAROVPM	Mask of the F0VAROVP bit
9	F0WATTOVPM	Mask of the F0WATTOVP bit
8	F0PEAKIFM	Mask of the F0PEAKIF bit
7	F0PEAKVFM	Mask of the F0PEAKVF bit
6	F0SAGFM	Mask of the F0SAGF bit
5	F0IF_OVRM	Mask of the F0IF_OVR bit
4	F0IF_FLLM	Mask of the F0IF_FLL bit
3	F0IF_EMPM	Mask of the F0IF_EMP bit
2	F0VF_OVRM	Mask of the F0VF_OVR bit
1	F0VF_FLLM	Mask of the F0VF_FLL bit
0	F0VF_EMPM	Mask of the F0VF_EMP bit

13.1.26. **ADCUI_F1CTR**

Table 89 – ADCUI_F1CTR Register

Number	31:20	19, 18	17:10	9, 8	7, 6
Access	R/W	R/W	R/W	R/W	R/W
Reset	0	0	00	0	0
	F1IRMSOS	F1VPHASE	F1VPHASE	F1VGAIN	F1IGAIN

Number	5	4	3	2	1	0
Access	WO	WO	WO	R/W	-	R/W
Reset	0	0	0	0	-	0
	F1RVRS	F1RRRS	F1RARS	F1VASEL	-	F1INTEN

Table 90 – Description of the ADCUI_F1CTR register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...20	F1IRMSOS	Offset calibration for the RMS calculator of current channel I1. Represented as signed values in two's-complement binary format; when applied in the calculator, the value is shifted left by 4 bits. Example: 007: IRMSOS = 7 · 16 = 112 FF9: IRMSOS = -7 · 16 = -112
19, 18	F1VPHASE[1:0]	Least significant bits of F1VPHASE
17...10	F1VPHASE[9:2]	Phase shift of the voltage channel relative to the current channel, encoded in two's-complement format. From −511·(ADC clock period) to +511·(ADC clock period). "0000000000" corresponds to an in-phase signal relative to the current. When DCHPENV1 = 1, it is recommended to adjust the phase shift in steps of 32.
9, 8	F1VGAIN	Preamplifier in the voltage channel: 00 – 0 dB (x1); 01 – 6 dB (x2); 10 – 12 dB (x4); 11 – 18 dB (x8)
7, 6	F1IGAIN	Preamplifier in current channel: 00 – 6 dB (x2); 01 – 12 dB (x4); 10 – 18 dB (x8); 11 – 24 dB (x16)
5	F1RVRS	Writing to this bit resets the transmitted apparent energy counter.
4	F1RRRS	Writing to this bit resets the transmitted reactive energy counter.
3	F1RARS	Writing to this bit resets the transmitted active energy counter.
2	F1VASEL	Selection of the signal source to be stored in the apparent energy register: 0 – apparent energy; 1 – RMS current value
1	-	Reserved
0	F1INTEN	Disabling the integrator in current channel: 0 – integrator enabled; 1 – integrator disabled. Correct operation of the integrator is guaranteed when the CHPF_COEFF value provides an HPF cutoff frequency of 1 Hz at the specified sampling rate. For example, if the sampling rate is 4 kHz, the CHPF_COEFF value must be 0x802 (default value).

13.1.27. **ADCUI_F1WC**

Table 91 – ADCUI_F1WC Register

Number	31…28	27…16	15…0
Access	-	R/W	R/W
Reset	-	000 h	0000 h
	-	F1WGAIN	F1WATTOS

Table 92 – Description of the ADCUI_F1WC register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...28	-	Reserved
27...16	F1WGAIN	Channel gain calibration
15...0	F1WATTOS	Channel offset calibration

13.1.28. **ADCUI_F1WATTP**

Table 93 – ADCUI_F1WATTP Register

Number	31…0
Access	RO
Reset	
	F1WATTHRP

Table 94 – Description of the ADCUI_F1WATTP register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...0	F1WATTHRP	The most significant 32 bits of the internal 57-bit accumulator of positive active energy

13.1.29. **ADCUI_F1WATTN**

Table 95 – ADCUI_F1WATTN Register

Number	31…0
Access	RO
Reset	
	F1WATTHRN

Table 96 – Description of the ADCUI_F1WATTN register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...0	F1WATTHRN	The most significant 32 bits of the internal 57-bit accumulator of negative active energy

13.1.30. **ADCUI_F1VC**

Table 97 – ADCUI_F1VC Register

Number	31…28	27…16	15…0
Access	-	R/W	R/W
Reset	-	000 h	0000 h
	-	F1VARGAIN	F1VAROS

Table 98 – Description of the ADCUI_F1VC register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...28	-	Reserved
27....16	F1VARGAIN	Channel gain calibration
15…0	F1VAROS	Channel offset calibration

13.1.31. **ADCUI_F1VARP**

Table 99 – ADCUI_F1VARP Register

Number	31…0
Access	RO
Reset	
	F1VARHRP

Table 100 – Description of the ADCUI_F1VARP register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...0	F1VARHRP	The most significant 32 bits of the internal 57-bit accumulator of positive reactive energy

13.1.32. **ADCUI_F1VARN**

Table 101 – ADCUI_F1VARN Register

Number	31…0
Access	RO
Reset	
	F1VARHRN

Table 102 – Description of the ADCUI_F1VARN register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...0	F1VARHRN	The most significant 32 bits of the internal 57-bit accumulator of negative reactive energy

13.1.33. **ADCUI_F1AC**

Table 103 – ADCUI_F1AC Register

Number	31…28	27…16	15…12	11…0
Access	-	R/W	-	R/W
Reset	-	000 h	-	000 h
	-	F1VAGAIN	-	F1VRMSOS

Table 104 – Description of the ADCUI_F1AC register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...28	-	Reserved
27...16	F1VAGAIN	Channel gain calibration
15...12	-	Reserved
11...0	F1VRMSOS	Calibration of the voltage RMS calculator. Represented as signed values in two's-complement binary format; when applied in the calculator, the value is shifted left by 4 bits. Example: 007: VRMSOS = 7 · 16 = 112 FF9: VRMSOS = -7 · 16 = -112

13.1.34. **ADCUI_F1VR**

Table 105 – ADCUI_F1VR Register

Number	31…0
Access	RO
Reset	
	F1VAHR

Table 106 – Description of the ADCUI_F1VR register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...0	F1VAHR	The most significant 32 bits of the internal 57-bit accumulator of apparent energy

13.1.35. **ADCUI_F1MD0**

Table 107 – ADCUI_F1MD0 Register

Number	31..29	28..8	7	6	5	4	3, 2	1, 0
Access	-	RO	R/W	R/W	R	R	R/W	R/W
Reset	-		0	0	0	0	00	00
	-	F1PER_FREQ	V1GAIN	I1GAIN	F1REACTS	F1ACTS	F1ISEL	F1VSEL

Table 108 – Description of the ADCUI_F1MD0 register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...29	-	Reserved
28...8	F1PER_FREQ	Duration of the input signal period in voltage channel V1. The value is expressed in ticks of the internal counter, averaged over the number of input-frequency periods defined by PER_LENGTH. The input signal period, based on the counter value, is calculated using the following formula: $$Period = \frac{F1PER\_FREQ \cdot 8}{ADCUI\_CLK} c.$$ For example, for a 50 Hz input signal, when the sigma-delta ADC block is clocked at 8.192 MHz (the recommended operating frequency), the F1PER_FREQ counter value is 20480; in this case, the value $$Period = \frac{20480 \cdot 8}{ADCUI\_CLK} = 0{,}02\,c$$
7	V1GAIN	Digital gain in voltage channel V1: 0 – no gain; 1 – +6 dB gain (result shifted left by 1).
6	I1GAIN	Digital gain in voltage channel V1: 0 – no gain; 1 – +6 dB gain (result shifted left by 1).
5	F1REACTS	Sign of reactive energy in the last period.
4	F1ACTS	Sign of active energy in the last period.
3, 2	F1ISEL	Selection of the signal source for the FIFO of current channel I1 (ADCUI_F1IDAT register): 00 – current samples (after or before the HPF, see F1ISEL_HPF); 01 – active power samples; 10 – reactive power samples; 11 – apparent power samples.
1, 0	F1VSEL	Selection of the signal source for the FIFO of voltage channel V1 (ADCUI_F1VDAT register): 00 – voltage samples before the high-pass filter; 01 – active power samples; 10 – reactive power samples; 11 – voltage samples after the high-pass filter.

13.1.36. **ADCUI_F1MD1**

Table 109 – ADCUI_F1MD1 Register

Number	31…16	15…0
Access	R/W	R/W
Reset	0000 h	0000 h
	F1VPKLVL	F1IPKLVL

Table 110 – Description of the ADCUI_F1MD1 register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...16	F1VPKLVL	Maximum permitted voltage level. The threshold value set in this register is compared with the most significant 16 bits at the output of the HPF filter of voltage channel V1. When the threshold is exceeded, the F1PEAKVF flag is set.
15...0	F1IPKLVL	Maximum permitted current level. The threshold value set in this register is compared with the most significant 16 bits at the output of the HPF filter of current channel I1. When the threshold is exceeded, the F0PEAKIF flag is set

13.1.37. **ADCUI_F1MD2**

Table 111 – ADCUI_F1MD2 Register

Number	31…17	16…0
Access	-	RO
Reset	-	00000 h
	-	F1PHASE

Table 112 – Description of the ADCUI_F1MD2 register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...17	-	Reserved
16...0	F1PHASE	Phase shift of voltage channel V1 relative to V0. The value is expressed in ticks of the internal counter, averaged over the number of input-signal periods defined by PER_LENGTH. The phase difference, based on the counter value, is calculated using the following formula: $$Difference = \frac{F1PHASE \cdot 2^{(3-OSR\_CONF)}}{ADCUI\_CLK} c,$$ where OSR_CONF can take the values 0, 1, or 2. For example, the phase difference between V0 and V1 is 180 degrees for 50 Hz signals. When the sigma-delta ADC block is clocked at 8.192 MHz and the resulting sampling rate is 16 kHz, the counter value is 40960. Then the value $$Difference = \frac{40960 \cdot 2}{ADCUI\ CLK} = 0{,}01c$$

13.1.38. **ADCUI_F1VPEAK**

Table 113 – ADCUI_F1VPEAK Register

Number	31…24	23…0
Access		R/W
Reset		000000 h
	-	F1VPEAK

Table 114 – Description of the ADCUI_F1VPEAK register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...24	-	Reserved
23...0	F1VPEAK	Peak voltage value. Cleared by reading from the register Writing to this register resets the peak value.

13.1.39. **ADCUI_F1IPEAK**

Table 115 – ADCUI_F1IPEAK Register

Number	31…24	23…0
Access		R/W
Reset		000000 h
	-	F1IPEAK

Table 116 – Description of the ADCUI_F1IPEAK register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...24	-	Reserved
23...0	F1IPEAK	Peak current value. Cleared by reading from the register. Writing to this register resets the peak value.

13.1.40. **ADCUI_F1VDAT**

Table 117 – ADCUI_F1VDAT Register

Number	31…24	23…0
Access	RO	RO
Reset	00 h	000000 h
	00 h	F1VDAT

Table 118 – Description of the ADCUI_F1VDAT register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...24	-	Reserved
23...0	F1VDAT	FIFO of voltage samples (or of one of the power values depending on F1VSEL). The FIFO is enabled when any of the bits V1EN, I1EN is set.

13.1.41. **ADCUI_F1IDAT**

Table 119 – ADCUI_F1IDAT Register

Number	31…24	23…0
Access	RO	RO
Reset	00 h	000000 h
	00 h	F1IDAT

Table 120 – Description of the ADCUI_F1IDAT register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...24	-	Reserved
23...0	F1IDAT	FIFO of current samples (or of one of the power values depending on F1ISEL). The FIFO is enabled when any of the bits V1EN, I1EN is set.

13.1.42. **ADCUI_F1VRMS**

Table 121 – ADCUI_F1VRMS Register

Number	31…24	23…0
Access		RO
Reset		000000 h
	-	F1VRMS

Table 122 – Description of the ADCUI_F1VRMS register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...24	-	Reserved
23..0	F1VRMS	RMS voltage

13.1.43. **ADCUI_F1VRMS2**

Table 123 – ADCUI_F1VRMS2 Register

Number	31…0
Access	RO
Reset	
	F1VRMS2

Table 124 – Description of the ADCUI_F1VRMS2 register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…0	F1VRMS2	Squared RMS voltage

13.1.44. **ADCUI_F1IRMS**

Table 125 – ADCUI_F1IRMS Register

Number	31…24	23…0
Access		R/W
Reset		000000 h
	-	F1IRMS

Table 126 – Description of the ADCUI_F1IRMS register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...24	-	Reserved
23...0	F1IRMS	RMS current value

13.1.45. **ADCUI_F1IRMS2**

Table 127 – ADCUI_F1IRMS2 Register

Number	31…0
Access	RO
Reset	
	F1IRMS2

Table 128 – Description of the ADCUI_F1IRMS2 register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...0	F1IRMS2	Squared RMS voltage

13.1.46. **ADCUI_F1STAT**

Table 129 – ADCUI_F1STAT Register

Number	31...27	26	25	24…22	21
Access	-	R/W	R/W	-	R/W
Reset	-		0	-	0
	-	F1VAROVN	F1WATTOVN	-	F1ZEROCRS

Number	20	19	18	17	16	15
Access	RO	-	RO	R/W	RO	R/W
Reset	0	-	0	0	0	0
	F1VANLDFL	-	F1VARNLDFL	F1VARSIGN	F1APNLDFL	F1APSIGN

Number	14, 13	12	11	10	9	8	7
Access	-	R/W	R/W	R/W	R/W	R/W	R/W
Reset	-	0	0	0	0	0	0
	-	F1ZXTOF	F1VAOV	F1VAROVP	F1WATTOVP	F1PEAKIF	F1PEAKVF

Number	6	5	4	3	2	1	0
Access	R/W	R/W	RO	RO	R/W	RO	RO
Reset	0	0	0	0	0	0	0
	F1SAGF	F1IF_OVR	F1IF_FLL	F1IF_EMP	F1VF_OVR	F1VF_FLL	F1VF_EMP

Table 130 – Description of the ADCUI_F1STAT register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...27	-	Reserved
26	F1VAROVN	Flag indicating that an overrun of the negative reactive energy accumulator register has occurred. Writing “1” clears this flag.
25	F1WATTOVN	Flag indicating that an overrun of the negative active energy accumulator register has occurred. Writing “1” clears this flag.
24...22	-	Reserved
21	F1ZEROCRS	Flag indicating that a voltage 0 crossing has occurred. Writing “1” clears this flag.
20	F1VANLDFL	Apparent power below the comparison level.
19	-	Reserved
18	F1VARNLDFL	Reactive power below the comparison level.
17	F1VARSIGN	Change of sign of reactive power. Writing “1” clears this flag.
16	F1APNLDFL	Active power below the comparison level.
15	F1APSIGN	Change of sign of active power. Writing “1” clears this flag.
14, 13	-	Reserved
12	F1ZXTOF	Flag indicating that no voltage 0 crossing occurred in channel V1 during ZXTOUT. Writing “1” clears this flag.

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
11	F1VAOV	Flag indicating that an overrun of the apparent energy accumulator register has occurred. Writing “1” clears this flag.
10	F1VAROVP	Flag indicating that an overrun of the positive reactive energy accumulator register has occurred. Writing “1” clears this flag.
9	F1WATTOVP	Flag indicating that an overrun of the positive active energy accumulator register has occurred. Writing “1” clears this flag.
8	F1PEAKIF	Flag indicating that a current threshold exceedance has occurred. Writing “1” clears this flag.
7	F1PEAKVF	Flag indicating that a voltage threshold exceedance has occurred. Writing “1” clears this flag.
6	F1SAGF	Flag indicating that a voltage sag has occurred. Writing “1” clears this flag.
5	F1IF_OVR	Flag indicating that an overrun of the F1IDAT FIFO has occurred. Writing “1” clears this flag.
4	F1IF_FLL	Flag indicating that the F1IDAT FIFO is full.
3	F1IF_EMP	Flag indicating that the F1IDAT FIFO is full.
2	F1VF_OVR	Flag indicating that an overrun of the F1VDAT FIFO has occurred. Writing “1” clears this flag.
1	F1VF_FLL	Flag indicating that the F1VDAT FIFO is full.
0	F1VF_EMP	Flag indicating that the F1VDAT FIFO is empty.

13.1.47. **ADCUI_F1MASK**

Table 131 – ADCUI_F1MASK Register

Number	31...27	26	25	24...22	21
Access	-	R/W	R/W	-	R/W
Reset	-		0	-	0
	-	F1VAROVNM	F1WATTOVNM	-	F1ZEROCRSM

Number	20	19	18	17	16
Access	R/W	-	R/W	R/W	R/W
Reset	0	-	0	0	0
	F1VANLDFLM	-	F1VARNLDFLM	F1VARSIGNM	F1APNLDFLM

Number	15	14, 13	12	11	10	9	8	7
Access	R/W	-	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	-	0	0	0	0	0	0
	F1APSIGNM	-	F1ZXTOFM	F1VAOVM	F1VAROVPM	F1WATTOVPM	F1PEAKIFM	F1PEAKVFM

Number	6	5	4	3	2	1	0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0
	F1SAGFM	F1IF_OVRM	F1IF_FLLM	F1IF_EMPM	F1VF_OVRM	F1VF_FLLM	F1VF_EMPM

Table 132 – Description of the ADCUI_F1MASK register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...27	-	Reserved
26	F1VAROVNM	Mask of the F1VAROVN bit
25	F1WATTOVNM	Mask of the F1WATTOVN bit
24...22	-	Reserved
21	F1ZEROCRSM	Mask of the F1ZEROCRS bit
20	F1VANLDFLM	Mask of the F1VANLDFL bit
19	-	Reserved
18	F1VARNLDFLM	Mask of the F1VARNLDFL bit
17	F1VARSIGNM	Mask of the F1VARSIGN bit
16	F1APNLDFLM	Mask of the F1VARNLDFL bit
15	F1APSIGNM	Mask of the F1APSIGNM bit
14, 13	-	Reserved
12	F1ZXTOFM	Mask of the F1ZXTOF bit
11	F1VAOVM	Mask of the F1VAOV bit
10	F1VAROVPM	Mask of the F1VAROVP bit
9	F1WATTOVPM	Mask of the F1WATTOVP bit
8	F1PEAKIFM	Mask of the F1PEAKIF bit
7	F1PEAKVFM	Mask of the F1PEAKVF bit
6	F1SAGFM	Mask of the F1SAGF bit
5	F1IF_OVRM	Mask of the F1IF_OVR bit
4	F1IF_FLLM	Mask of the F1IF_FLL bit
3	F1IF_EMPM	Mask of the F1IF_EMP bit
2	F1VF_OVRM	Mask of the F1VF_OVR bit
1	F1VF_FLLM	Mask of the F1VF_FLL bit
0	F1VF_EMPM	Mask of the F1VF_EMP bit

13.1.48. **ADCUI_F2CTR**

Table 133 – ADCUI_F2CTR Register

Number	31…20	19, 18	17…10	9, 8	7, 6
Access	R/W	R/W	R/W	R/W	R/W
Reset	0	0	00	0	0
	F2IRMSOS	F2VPHASE	F2VPHASE	F2VGAIN	F2IGAIN

Number	5	4	3	2	1	0
Access	WO	WO	WO	R/W	-	R/W
Reset	0	0	0	0	-	0
	F2RVRS	F2RRRS	F2RARS	F2VASEL	-	F2INTEN

Table 134 – Description of the ADCUI_F2CTR register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...20	F2IRMSOS	Offset calibration for the RMS calculator of current channel I2. Represented as signed values in two's-complement binary format; when applied in the calculator, the value is shifted left by 4 bits. Example: 007: IRMSOS = 7 · 16 = 112 FF9: IRMSOS = -7 · 16 = -112
19, 18	F2VPHASE[1:0]	Least significant bits of F2VPHASE
17...10	F2VPHASE[9:2]	Phase shift of the voltage channel relative to the current channel, encoded in two's-complement format. From −511·(ADC clock period) to +511·(ADC clock period). "0000000000" corresponds to an in-phase signal relative to the current. When DCHPENV2 = 1, it is recommended to adjust the phase shift in steps of 32.
9, 8	F2VGAIN	Preamplifier in the voltage channel: 00 – 0 dB (x1); 01 – 6 dB (x2); 10 – 12 dB (x4); 11 – 18 dB (x8)
7, 6	F2IGAIN	Preamplifier in current channel: 00 – 6 dB (x2); 01 – 12 dB (x4); 10 – 18 dB (x8); 11 – 24 dB (x16)
5	F2RVRS	Writing to this bit resets the transmitted apparent energy counter.
4	F2RRRS	Writing to this bit resets the transmitted reactive energy counter.
3	F2RARS	Writing to this bit resets the transmitted active energy counter.
2	F2VASEL	Selection of the signal source to be stored in the apparent energy register: 0 – apparent energy; 1 – RMS current value
1	-	Reserved
0	F2INTEN	Disabling the integrator in current channel: 0 – integrator enabled; 1 – integrator disabled. Correct operation of the integrator is guaranteed when the CHPF_COEFF value provides an HPF cutoff frequency of 1 Hz at the specified sampling rate. For example, if the sampling rate is 4 kHz, the CHPF_COEFF value must be 0x802 (default value).

13.1.49. **ADCUI_F2WC**

Table 135 – ADCUI_F2WC Register

Number	31…28	27…16	15…0
Access	-	R/W	R/W
Reset	-	000 h	0000 h
	-	F2WGAIN	F2WATTOS

Table 136 – Description of the ADCUI_F2WC register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...28	-	Reserved
27...16	F2WGAIN	Channel gain calibration
15...0	F2WATTOS	Channel offset calibration

13.1.50. **ADCUI_F2WATTP**

Table 137 – ADCUI_F2WATTP Register

Number	31…0
Access	RO
Reset	
	F2WATTHRP

Table 138 – Description of the ADCUI_F2WATTP register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...0	F2WATTHRP	The most significant 32 bits of the internal 57-bit accumulator of positive active energy

13.1.51. **ADCUI_F2WATTN**

Table 139 – ADCUI_F2WATTN Register

Number	31…0
Access	RO
Reset	
	F2WATTHRN

Table 140 – Description of the ADCUI_F2WATTN register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...0	F2WATTHRN	The most significant 32 bits of the internal 57-bit accumulator of negative active energy

13.1.52. **ADCUI_F2VC**

Table 141 – ADCUI_F2VC Register

Number	31…28	27…16	15…0
Access	-	R/W	R/W
Reset	-	000 h	0000 h
	-	F2VARGAIN	F2VAROS

Table 142 – Description of the ADCUI_F2VC register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...28	-	Reserved
27...16	F2VARGAIN	Channel gain calibration
15...0	F2VAROS	Channel offset calibration

13.1.53. **ADCUI_F2VARP**

Table 143 – ADCUI_F2VARP Register

Number	31…0
Access	RO
Reset	
	F2VARHRP

Table 144 – Description of the ADCUI_F2VARP register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...0	F2VARHRP	The most significant 32 bits of the internal 57-bit accumulator of positive reactive energy

13.1.54. **ADCUI_F2VARN**

Table 145 – ADCUI_F2VARN Register

Number	31…0
Access	RO
Reset	
	F2VARHRN

Table 146 – Description of the ADCUI_F2VARN register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...0	F2VARHRN	The most significant 32 bits of the internal 57-bit accumulator of negative reactive energy

13.1.55. **ADCUI_F2AC**

Table 147 – ADCUI_F2AC Register

Number	31…28	27…16	15…12	11…0
Access	-	R/W	-	R/W
Reset	-	000 h	-	000 h
	-	F2VAGAIN	-	F2VRMSOS

Table 148 – Description of the ADCUI_F2AC register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...28	-	Reserved
27...16	F2VAGAIN	Channel gain calibration
15...12	-	Reserved
11...0	F2VRMSOS	Calibration of the voltage RMS calculator. Represented as signed values in two's-complement binary format; when applied in the calculator, the value is shifted left by 4 bits. Example: 007: VRMSOS = 7 · 16 = 112 FF9: VRMSOS = -7 · 16 = -112

13.1.56. **ADCUI_F2VR**

Table 149 – ADCUI_F2VR Register

Number	31…0
Access	R
Reset	
	F2VAHR

Table 150 – Description of the ADCUI_F2VR register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...0	F2VAHR	The most significant 32 bits of the internal 57-bit accumulator of apparent energy

13.1.57. **ADCUI_F2MD0**

Table 151 – ADCUI_F2MD0 Register

Number	31..29	28..8	7	6	5	4	3, 2	1, 0
Access	-	RO	R/W	R/W	R	R	R/W	R/W
Reset	-	-	0	0	0	0	00	00
	-	F2PER_FREQ	V2GAIN	I2GAIN	F2REACTS	F2ACTS	F2ISEL	F2VSEL

Table 152 – Description of the ADCUI_F2MD0 register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...29	-	Reserved
28...8	F2PER_FREQ	Duration of the input signal period in voltage channel V2. The value is expressed in ticks of the internal counter, averaged over the number of input-frequency periods defined by PER_LENGTH. The input signal period, based on the counter value, is calculated using the following formula: $$Period = \frac{F2PER\_FREQ \cdot 8}{ADCUI\_CLK}\ c,$$ For example, for a 50 Hz input signal, when the sigma-delta ADC block is clocked at 8.192 MHz (the recommended operating frequency), the F2PER_FREQ counter value is 20480; in this case, the value $$Period = \frac{20480 \cdot 8}{ADCUI\_CLK} = 0{,}02\,c$$
7	V2GAIN	Digital gain in voltage channel V2: 0 – no gain; 1 – +6 dB gain (result shifted left by 1).
6	I2GAIN	Digital gain in voltage channel I2: 0 – no gain; 1 – +6 dB gain (result shifted left by 1).
5	F2REACTS	Sign of reactive energy in the last period.
4	F2ACTS	Sign of active energy in the last period.
3, 2	F2ISEL	Selection of the signal source for the FIFO of current channel I2 (ADCUI_F2IDAT register): 00 – current samples (after or before the HPF, see F2ISEL_HPF); 01 – active power samples; 10 – reactive power samples; 11 – apparent power samples.
1, 0	F2VSEL	Selection of the signal source for the FIFO of voltage channel V2 (ADCUI_F2VDAT register): 00 – voltage samples before the high-pass filter; 01 – active power samples; 10 – reactive power samples; 11 – voltage samples after the high-pass filter.

13.1.58. **ADCUI_F2MD1**

Table 153 – ADCUI_F2MD1 Register

Number	31…16	15…0
Access	R/W	R/W
Reset	0000 h	0000 h
	F2VPKLVL	F2IPKLVL

Table 154 – Description of the ADCUI_F2MD1 register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...16	F2VPKLVL	Maximum permitted voltage level. The threshold value set in this register is compared with the most significant 16 bits at the output of the HPF filter of voltage channel V2. When the threshold is exceeded, the F2PEAKVF flag is set.
15...0	F2IPKLVL	Maximum permitted current level. The threshold value set in this register is compared with the most significant 16 bits at the output of the HPF filter of current channel I2. When the threshold is exceeded, the F2PEAKVF flag is set.

13.1.59. **ADCUI_F2MD2**

Table 155 – ADCUI_F2MD2 Register

Number	31…17	16…0
Access	-	RO
Reset	-	00000 h
	-	F2PHASE

Table 156 – Description of the ADCUI_F2MD2 register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...17	-	Reserved
16...0	F2PHASE	Phase shift of voltage channel V2 relative to V0. The value is expressed in ticks of the internal counter, averaged over the number of input-signal periods defined by PER_LENGTH. The phase difference, based on the counter value, is calculated using the following formula: $$Difference = \frac{F2PHASE \cdot 2^{(3-OSR\_CONF)}}{ADCUI\_CLK} c,$$ where OSR_CONF can take the values 0, 1, or 2. For example, the phase difference between V0 and V2 is 180 degrees for 50 Hz signals. When the sigma-delta ADC block is clocked at 8.192 MHz and the resulting sampling rate is 16 kHz, the counter value is 40960. Then the value $$Difference = \frac{40960 \cdot 2}{ADCUI\ CLK} = 0{,}01c$$

13.1.60. **ADCUI_F2VPEAK**

Table 157 – ADCUI_F2VPEAK Register

Number	31…24	23…0
Access		R/W
Reset		000000 h
	-	F2VPEAK

Table 158 – Description of the ADCUI_F2VPEAK register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...24	-	Reserved
23...0	F2VPEAK	Peak voltage value. Cleared by reading from the register

13.1.61. **ADCUI_F2IPEAK**

Table 159 – ADCUI_F2IPEAK Register

Number	31…24	23…0
Access		R/W
Reset		000000 h
	-	F2IPEAK

Table 160 – Description of the ADCUI_F2IPEAK register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...24	-	Reserved
23...0	F2IPEAK	Peak current value. Cleared by reading from the register

13.1.62. **ADCUI_F2VDAT**

Table 161 – ADCUI_F2VDAT Register

Number	31…24	23…0
Access	RO	R/W
Reset	00 h	000000 h
	00 h	F2VDAT

Table 162 – Description of the ADCUI_F2VDAT register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...24	-	Reserved
23...0	F2VDAT	FIFO of voltage samples (or of one of the power values). The FIFO is enabled when any of the bits V2EN, I2EN is set.

13.1.63. **ADCUI_F2IDAT**

Table 163 – ADCUI_F2IDAT Register

Number	31…24	23…0
Access	RO	R/W
Reset	00h	000000h
	00h	F2IDAT

Table 164 – Description of the ADCUI_F2IDAT register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...24	-	Reserved
23...0	F2IDAT	FIFO of current samples (or of one of the power values). The FIFO is enabled when any of the bits V2EN, I2EN is set.

13.1.64. **ADCUI_F2VRMS**

Table 165 – ADCUI_F2VRMS Register

Number	31…24	23…0
Access		R/W
Reset		000000h
	-	F2VRMS

Table 166 – Description of the ADCUI_F2VRMS register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…24	-	Reserved
23...0	F2VRMS	RMS voltage

13.1.65. **ADCUI_F2VRMS2**

Table 167 – ADCUI_F2VRMS2 Register

Number	31…0
Access	RO
Reset	
	F2VRMS2

Table 168 – Description of the ADCUI_F2VRMS2 register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...0	F2VRMS2	Squared RMS voltage

13.1.66. **ADCUI_F2IRMS**

Table 169 – ADCUI_F2IRMS Register

Number	31…24	23…0
Access		R/W
Reset		000000h
	-	F2IRMS

Table 170 – Description of the ADCUI_F2IRMS register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...24	-	Reserved
23...0	F2IRMS	RMS current value

13.1.67. **ADCUI_F2IRMS2**

Table 171 – ADCUI_F2IRMS2 Register

Number	31…0
Access	RO
Reset	
	F2IRMS2

Table 172 – Description of the ADCUI_F2IRMS2 register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...0	F2IRMS2	Squared RMS voltage

13.1.68. **ADCUI_F2STAT**

Table 173 – ADCUI_F2STAT Register

Number	6	5	4	3	2	1	0
Access	R/W	R/W	RO	RO	R/W	RO	RO
Reset	0	0	0	0	0	0	0
	F2SAGF	F2IF_OVR	F2IF_FLL	F2IF_EMP	F2VF_OVR	F2VF_FLL	F2VF_EMP

Number	15	14, 13	12	11	10	9	8	7
Access	R/W	-	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	-	0	0	0	0	0	0
	F2APSIGN	-	F2ZXTOF	F2VAOV	F2VAROVP	F2WATTOVP	F2PEAKIF	F2PEAKVF

Number	31..27	26	25	24…22	21	20	19	18	17	16
Access	-	R/W	R/W	-	R/W	RO	-	RO	R/W	RO
Reset	-		0	-	0	0	-	0	0	0
	-	F2VAROVN	F2WATTOVN	-	F2ZEROCRS	F2VANLDFL	-	F2VARNLDFL	F2VARSIGN	F2APNLDFL

Table 174 – Description of the ADCUI_F2STAT register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…27	-	Reserved
26	F2VAROVN	Flag indicating that an overrun of the negative reactive energy accumulator register has occurred. Writing “1” clears this flag.
25	F2WATTOVN	Flag indicating that an overrun of the negative active energy accumulator register has occurred. Writing “1” clears this flag.
24…22	-	Reserved
21	F2ZEROCRS	Flag indicating that a voltage 0 crossing has occurred. Writing “1” clears this flag.
20	F2VANLDFL	Apparent power below the comparison level.
19	-	Reserved
18	F2VARNLDFL	Reactive power below the comparison level.
17	F2VARSIGN	Change of sign of reactive power. Writing “1” clears this flag.
16	F2APNLDFL	Active power below the comparison level.
15	F2APSIGN	Change of sign of active power. Writing “1” clears this flag.
14, 13	-	Reserved
12	F2ZXTOF	Flag indicating that no voltage 0 crossing occurred in channel V2 during ZXTOUT. Writing “1” clears this flag.
11	F2VAOV	Flag indicating that an overrun of the apparent energy accumulator register has occurred. Writing “1” clears this flag.
10	F2VAROVP	Flag indicating that an overrun of the positive reactive energy accumulator register has occurred. Writing “1” clears this flag.
9	F2WATTOVP	Flag indicating that an overrun of the positive active energy accumulator register has occurred. Writing “1” clears this flag.
8	F2PEAKIF	Flag indicating that a current threshold exceedance has occurred. Writing “1” clears this flag.
7	F2PEAKVF	Flag indicating that a voltage threshold exceedance has occurred. Writing “1” clears this flag.
6	F2SAGF	Flag indicating that a voltage sag has occurred. Writing “1” clears this flag.
5	F2IF_OVR	Flag indicating that an overrun of the F2IDAT FIFO has occurred. Writing “1” clears this flag.
4	F2IF_FLL	Flag indicating that the F2IDAT FIFO is full.
3	F2IF_EMP	Flag indicating that the F2IDAT FIFO is empty.
2	F2VF_OVR	Flag indicating that an overrun of the F2VDAT FIFO has occurred. Writing “1” clears this flag.
1	F2VF_FLL	Flag indicating that the F2VDAT FIFO is full.
0	F2VF_EMP	Flag indicating that the F2VDAT FIFO is empty.

13.1.69. **ADCUI_F2MASK**

Table 175 – ADCUI_F2MASK Register

Number	31..27	26	25	24…22	21	20	19	18	17	16
Access	-	R/W	R/W	-	R/W	R/W	-	R/W	R/W	R/W
Reset	-		0	-	0	0	-	0	0	0
	-	F2VAROVNM	F2WATTOVNM	-	F2ZEROCRSM	F2VANLDFLM	-	F2VARNLDFLM	F2VARSIGNM	F2APNLDFLM

Number	15	14, 13	12	11	10	9	8	7
Access	R/W	-	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	-	0	0	0	0	0	0
	F2APSIGNM	-	F2ZXTOFM	F2VAOVM	F2VAROVPM	F2WATTOVPM	F2PEAKIFM	F2PEAKVFM

Number	6	5	4	3	2	1	0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0
	F2SAGFM	F2IF_OVRM	F2IF_FLLM	F2IF_EMPM	F2VF_OVRM	F2VF_FLLM	F2VF_EMPM

Table 176 – Description of the ADCUI_F2MASK register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...27	-	Reserved
26	F2VAROVNM	Mask of the F2VAROVN bit
25	F2WATTOVNM	Mask of the F2WATTOVN bit
24...22	-	Reserved
21	F2ZEROCRSM	Mask of the F2ZEROCRS bit
20	F2VANLDFLM	Mask of the F2VANLDFL bit
19	-	Reserved
18	F2VARNLDFLM	Mask of the F2VARNLDFL bit
17	F2VARSIGNM	Mask of the F2VARSIGN bit
16	F2APNLDFLM	Mask of the F2VARNLDFL bit
15	F2APSIGNM	Mask of the F2APSIGNM bit
14, 13	-	Reserved
12	F2ZXTOFM	Mask of the F2ZXTOF bit
11	F2VAOVM	Mask of the F2VAOV bit
10	F2VAROVPM	Mask of the F2VAROVP bit
9	F2WATTOVPM	Mask of the F2WATTOVP bit
8	F2PEAKIFM	Mask of the F2PEAKIF bit
7	F2PEAKVFM	Mask of the F2PEAKVF bit
6	F2SAGFM	Mask of the F2SAGF bit
5	F2IF_OVRM	Mask of the F2IF_OVR bit
4	F2IF_FLLM	Mask of the F2IF_FLL bit

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
3	F2IF_EMPM	Mask of the F2IF_EMP bit
2	F2VF_OVRM	Mask of the F2VF_OVR bit
1	F2VF_FLLM	Mask of the F2VF_FLL bit
0	F2VF_EMPM	Mask of the F2VF_EMP bit

13.1.70. **ADCUI_CCAL1**

Table 177 – ADCUI_CCAL1 Register

Number	31…16	15…0
Access	R/W	R/W
Reset	0000	0000
	I0BGAIN	V0BGAIN

Table 178 – Description of the ADCUI_CCAL1 register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...16	I0BGAIN	Calibration coefficient of channel I0
15...0	V0BGAIN	Calibration coefficient of channel V0

13.1.71. **ADCUI_CCAL2**

Table 179 – ADCUI_CCAL2 Register

Number	31…16	15…0
Access	R/W	R/W
Reset	0000	0000
	I1BGAIN	V1BGAIN

Table 180 – Description of the ADCUI_CCAL2 register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...16	I1BGAIN	Calibration coefficient of channel I1
15...0	V1BGAIN	Calibration coefficient of channel V1

13.1.72. **ADCUI_CCAL3**

Table 181 – ADCUI_CCAL3 Register

Number	31…16	15…0
Access	R/W	R/W
Reset	0000	0000
	I2BGAIN	V2BGAIN

Table 182 – Description of the ADCUI_CCAL3 register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...16	I2BGAIN	Calibration coefficient of channel I2
15...0	V2BGAIN	Calibration coefficient of channel V2

13.1.73. **ADCUI_CCAL4**

Table 183 – ADCUI_CCAL4 Register

Number	31…16	15…0
Access	U	R/W
Reset	-	0000
	-	I3BGAIN

Table 184 – Description of the ADCUI_CCAL4 register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…16	-	Reserved
15...0	I3BGAIN	Channel I3 calibration factor

13.1.74. **ADCUI_F0IRMS_INACTIVE**

Table 185 – ADCUI_F0IRMS_INACTIVE Register

Number	31…24	23…0
Access		RO
Reset		000000 h
	-	F0IRMS_INACTIVE

Table 186 – Description of the ADCUI_F0IRMS_INACTIVE register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...24	-	Reserved
23...0	F0IRMS_INACTIVE	RMS current value of the inactive channel

13.1.75. **ADCUI_F0WATTP_L**

Table 187 – ADCUI_F0WATTP_L Register

Number	24…0
Access	RO
Reset	
	F0WATTHRP_L

Table 188 – Description of the ADCUI_F0WATTP_L register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
24…0	F0WATTHRP_L	Least significant 25 bits of the internal 57-bit accumulator of positive active energy. Read only after the upper part

13.1.76. **ADCUI_F0WATTN_L**

Table 189 – ADCUI_F0WATTN_L Register

Number	24…0
Access	RO
Reset	
	F0WATTHRN_L

Table 190 – Description of the ADCUI_F0WATTN_L register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
24...0	F0WATTHRN_L	Least significant 25 bits of the internal 57-bit accumulator of negative active energy Read only after the upper part

13.1.77. **ADCUI_F0VARP_L**

Table 191 – ADCUI_F0VARP_L Register

Number	24..0
Access	RO
Reset	
	F0VARHRP_L

Table 192 – Description of the ADCUI_F0VARP_L register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
24...0	F0VARHRP_L	Least significant 25 bits of the internal 57-bit accumulator of positive reactive energy. Read only after the upper part

13.1.78. **ADCUI_F0VARN_L**

Table 193 – ADCUI_F0VARN_L Register

Number	24..0
Access	RO
Reset	
	F0VARHRN_L

Table 194 – Description of the ADCUI_F0VARN_L register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
24...0	F0VARHRN_L	Least significant 25 bits of the internal 57-bit accumulator of negative reactive energy Read only after the upper part

13.1.79. **ADCUI_F0VR_L**

Table 195 – ADCUI_F0VR_L Register

Number	24…0
Access	RO
Reset	
	F0VAHR_L

Table 196 – Description of the ADCUI_F0VR_ register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
24…0	F0VAHR_L	Least significant 25 bits of the internal 57-bit accumulator of apparent energy Read only after the upper part

13.1.80. **ADCUI_F1WATTP_L**

Table 197 – ADCUI_F1WATTP_L Register

Number	24…0
Access	RO
Reset	
	F1WATTHRP_L

Table 198 – Description of the ADCUI_F1WATTP_L register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
24…0	F1WATTHRP_L	Least significant 25 bits of the internal 57-bit accumulator of positive active energy. Read only after the upper part

13.1.81. **ADCUI_F1WATTN_L**

Table 199 – ADCUI_F1WATTN_L Register

Number	24…0
Access	RO
Reset	
	F1WATTHRN_L

Table 200 – Description of the ADCUI_F1WATTN_L register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
24...0	F1WATTHRN_L	Least significant 25 bits of the internal 57-bit accumulator of negative active energy Read only after the upper part

13.1.82. **ADCUI_F1VARP_L**

Table 201 – ADCUI_F1VARP_L Register

Number	24..0
Access	RO
Reset	
	F1VARHRP_L

Table 202 – Description of the ADCUI_F1VARP_L register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
24...0	F1VARHRP_L	Least significant 25 bits of the internal 57-bit accumulator of positive reactive energy. Read only after the upper part

13.1.83. **ADCUI_F1VARN_L**

Table 203 – ADCUI_F1VARN_L Register

Number	24..0
Access	RO
Reset	
	F1VARHRN_L

Table 204 – Description of the ADCUI_F1VARN_L register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
24...0	F1VARHRN_L	Least significant 25 bits of the internal 57-bit accumulator of negative reactive energy Read only after the upper part

13.1.84. **ADCUI_F1VR_L**

Table 205 – ADCUI_F1VR_L Register

Number	24…0
Access	RO
Reset	
	F1VAHR_L

Table 206 – Description of the ADCUI_F1VR_L register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
24…0	F1VAHR_L	Least significant 25 bits of the internal 57-bit accumulator of apparent energy Read only after the upper part

13.1.85. **ADCUI_F2WATTP_L**

Table 207 – ADCUI_F2WATTP_L Register

Number	24…0
Access	RO
Reset	
	F2WATTHRP_L

Table 208 – Description of the ADCUI_F2WATTP_L register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
24…0	F2WATTHRP_L	Least significant 25 bits of the internal 57-bit accumulator of positive active energy. Read only after the upper part

13.1.86. **ADCUI_F2WATTN_L**

Table 209 – ADCUI_F2WATTN_L Register

Number	24…0
Access	RO
Reset	
	F2WATTHRN_L

Table 210 – Description of the ADCUI_F2WATTN_L register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
24...0	F2WATTHRN_L	Least significant 25 bits of the internal 57-bit accumulator of negative active energy Read only after the upper part

13.1.87. **ADCUI_F2VARP_L**

Table 211 – ADCUI_F2VARP_L Register

Number	24..0
Access	RO
Reset	
	F2VARHRP_L

Table 212 – Description of the ADCUI_F2VARP_L register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
24...0	F2VARHRP_L	Least significant 25 bits of the internal 57-bit accumulator of positive reactive energy. Read only after the upper part

13.1.88. **ADCUI_F2VARN_L**

Table 213 – ADCUI_F2VARN_L Register

Number	24..0
Access	RO
Reset	
	F2VARHRN_L

Table 214 – Description of the ADCUI_F2VARN_L register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
24...0	F2VARHRN_L	Least significant 25 bits of the internal 57-bit accumulator of negative reactive energy Read only after the upper part

13.1.89. **ADCUI_F2VR_L**

Table 215 – ADCUI_F2VR_L Register

Number	24…0
Access	RO
Reset	
	F2VAHR_L

Table 216 – Description of the ADCUI_F2VR_L register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
24…0	F2VAHR_L	Least significant 25 bits of the internal 57-bit accumulator of apparent energy Read only after the upper part

13.1.90. **ADCUI_LOAD**

Table 217 – ADCUI_LOAD Register

Number	31…18	17…12	11…6	5…0
Access	U	R/W	R/W	R/W
Reset	-	00	00	00
		VANOLOAD	VARNOLOAD	APNOLOAD

Table 218 – Description of the ADCUI_LOAD register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…18	-	Reserved
17…12	VANOLOAD	"No-load" mode when calculating apparent energy: 000000 – calculation uses bits 18..17 of the ADCUI_CTRL1 register; 000001 – energy below 0.00156% of the full scale is ignored; 000010 – energy below 0.00312% of the full scale is ignored; … 111111 – energy below 0.1% of the full scale is ignored.
11…6	VARNOLOAD	"No-load" mode when calculating reactive energy: 000000 – calculation uses bits 16..15 of the ADCUI_CTRL1 register; 000001 – energy below 0.00078% of the full scale is ignored; 000010 – energy below 0.00156% of the full scale is ignored; … 111111 – energy below 0.05% of the full scale is ignored.
5...0	APNOLOAD	"No-load" mode when calculating active energy: 000000 – calculation uses bits 13..12 of the ADCUI_CTRL1 register; 000001 – energy below 0.00078% of the full scale is ignored; 000010 – energy below 0.00156% of the full scale is ignored; … 111111 – energy below 0.05% of the full scale is ignored.

13.1.91. **ADCUI_F0VRMS_TRUE**

Table 219 – ADCUI_F0VRMS_TRUE Register

Number	31…29	28…0
Access	U	R/W
Reset	-	0000_0000
	-	ADCUI_F0VRMS_TRUE

Table 220 – Description of the ADCUI_F0VRMS_TRUE register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...29	-	Reserved
28…0	ADCUI_F0VRMS_TRUE	RMS voltage value over one period for channel F0. Updated on voltage 0 crossing. This function operates only for a periodic signal; use of the calculation results when a 0 V signal is applied is not permitted.

13.1.92. **ADCUI_F1VRMS_TRUE**

Table 221 – ADCUI_F1VRMS_TRUE Register

Number	31…29	28…0
Access	U	R/W
Reset	-	0000_0000
	-	ADCUI_F1VRMS_TRUE

Table 222 – Description of the ADCUI_F1VRMS_TRUE register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...29	-	Reserved
28…0	ADCUI_F1VRMS_TRUE	RMS voltage value over one period for channel F1. Updated on voltage 0 crossing. This function operates only for a periodic signal; use of the calculation results when a 0 V signal is applied is not permitted.

13.1.93. **ADCUI_F2VRMS_TRUE**

Table 223 – ADCUI_F2VRMS_TRUE Register

Number	31…29	28…0
Access	U	R/W
Reset	-	0000_0000
	-	ADCUI_F2VRMS_TRUE

Table 224 – Description of the ADCUI_F2VRMS_TRUE register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...29	-	Reserved
28…0	ADCUI_F2VRMS_TRUE	RMS voltage value over one period for channel F2. Updated on voltage 0 crossing. This function operates only for a periodic signal; use of the calculation results when a 0 V signal is applied is not permitted.

13.1.94. **ADCUI_MSC**

Table 225 – ADCUI_MSC Register

Number	31	30..29	28	27	26	25…14	13…7	6…0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	000	00	00
	F90_CONF	F0VPHASE	F2ISEL_HPF	F1ISEL_HPF	F0ISEL_HPF	F0I3RMSOS	DMA_EN_RQ_S	DMA_EN_RQ_B

Table 226 – Description of the ADCUI_MSC register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31	F90_CONF	Selection of the voltage used for calculating active and reactive power: 0 – the voltage value before the HPF filter is used (legacy mode); 1 – the voltage value after the HPF filter is used (recommended).
30…29	F0VPHASE[1:0]	Least significant bits of F0VPHASE. See ADCUI_F0CTRL.
28	F2ISEL_HPF	Data written to the F2IDAT FIFO: 0 – sampled after the HPF filter; 1 – sampled before the HPF filter.
27	F1ISEL_HPF	Data written to the F1IDAT FIFO: 0 – sampled after the HPF filter; 1 – sampled before the HPF filter.
26	F0ISEL_HPF	Data written to the F0IDAT FIFO: 0 – sampled after the HPF filter; 1 – sampled before the HPF filter.
25…14	F0I3RMSOS	Offset calibration for the RMS calculator of current channel I3. Represented as signed values in two's-complement binary format; when applied in the calculator, the value is shifted left by 4 bits. Example: 007: IRMSOS = 7 · 16 = 112 FF9: IRMSOS = -7 · 16 = -112
13…7	DMA_EN_RQ_S	Enable generation of DMA requests (single transactions) from ΔΣ ADC channels 7...1.
6…0	DMA_EN_RQ_B	Enable generation of DMA requests (block transactions) from ΔΣ ADC channels 7...1

13.1.95. **ADCUI_FILTERCFG**

Table 227 – ADCUI_FILTERCFG Register

Number	31..26	25…24	23…12	11…0
Access	-	R/W	R/W	R/W
Reset	-	0	803 h	802 h
	-	CHP_ADC_CFG	CLRMS_COEFF	CHPF_COEFF

Table 228 – Description of the ADCUI_FILTERCFG register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…26	-	Reserved
25…24	CHP_ADC_CFG	Configuration of the ADC chopper period for all channels: 0 – 256 periods of the ADC synchronization signal (recommended for OSR = 256); 1 – 512 periods of the ADC synchronization signal (recommended for OSR = 512); 2 – 1024 periods of the ADC synchronization signal (recommended for OSR = 1024)
23…12	CLRMS_COEFF	Coefficient for the LPF filters in the RMS calculation blocks, which determines the cutoff frequency. The reset value is "−2045", which corresponds to a cutoff frequency of 1 Hz at the −3 dB level with a sampling rate of 4 kHz. Calculation formula $$CLRMS\_COEFF = \frac{2^{11} \times \left(\tan\left(\pi \times \left(\frac{freq\_lpf}{samplerate\_adc}\right)\right) - 1\right)}{\tan\left(\pi \times \left(\frac{freq\_lpf}{samplerate\_adc}\right)\right) + 1},$$ where freq_lpf is the required filter cutoff frequency; samplerate_adc is the ADC sampling rate. The result must be rounded to a signed integer. The result must not be less than −2047. Code for calculation in CAD tools: `CLRMS_COEFF=max(-2047, round(2^11*(tan(pi*(freq_lpf/samplerate_adc))-1)/(tan(pi*(freq_lpf/samplerate_adc))+1)));`

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
11…0	CHPF_COEFF	Coefficient for the HPF filters in all ADC channels, which determines the cutoff frequency. The reset value is "−2046", which corresponds to a cutoff frequency of 0.5 Hz at the −3 dB level with a sampling rate of 4 kHz. The recommended value is "−2016". The value "−2048" (cutoff frequency of 0 Hz) must not be used. Calculation formula $$CHPF\_COEFF = \frac{2^{11} \times \left(\tan\left(\pi \times \left(\frac{freq\_hpf}{samplerate\_adc}\right)\right) - 1\right)}{\tan\left(\pi \times \left(\frac{freq\_hpf}{samplerate\_adc}\right)\right) + 1},$$ where freq_hpf is the required filter cutoff frequency and samplerate_adc is the ADC sampling rate. The result must be rounded to a signed integer. Code for calculation in CAD tools: `CHPF_COEFF=max(-2047, round(2^11*(tan(pi*(freq_hpf/samplerate_adc))-1)/(tan(pi*(freq_hpf/samplerate_adc))+1)));`

13.1.96. **ADCUI_CLKPHASE1**

Table 229 – ADCUI_CLKPHASE1 Register

Number	31…30	29…20	19…10	9..0
Access	R/W	R/W	R/W	R/W
Reset	0	200h	200h	200h
	CLC_ADC_CFG	I2PHASE	I1PHASE	I0PHASE

Table 230 – Description of the ADCUI_CLKPHASE1 register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…30	CLC_ADC_CFG	Configuration of the ADC clock frequency for all channels: 00 – 4 MHz; 01 – 2 MHz; 10 – 1 MHz; 11 – reserved. This frequency is also output on PD14 when external delta-sigma modulators are used (when the pin is configured for the CLK_SD_EXT function).
29…20	I2PHASE[9:0]	Phase shift of current channel I2. From 0 · (ADC clock cycle) to +1023 · (ADC clock cycles). "1000000000" corresponds to an in-phase signal with voltage V2. When DCHPENI2 = 1, it is recommended to adjust the phase shift in steps of 32

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
19…10	I1PHASE[9:0]	Phase shift of current channel I1. From 0 · (ADC clock cycle) to +1023 · (ADC clock cycles). “1000000000” corresponds to an in-phase signal with voltage V1. When DCHPENI1 = 1, it is recommended to adjust the phase shift in steps of 32.
9…0	I0PHASE[9:0]	Phase shift of current channel I0. From 0 · (ADC clock cycle) to +1023 · (ADC clock cycles). “1000000000” corresponds to an in-phase signal with voltage V0. When DCHPENI0 = 1, it is recommended to adjust the phase shift in steps of 32.

13.1.97. **ADCUI_CLKPHASE2**

Table 231 – ADCUI_CLKPHASE2 Register

Number	31…26	9…0
Access	-	R/W
Reset	-	200h
	-	I3PHASE

Table 232 – Description of the ADCUI_CLKPHASE2 register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…26	-	Reserved
9...0	I3PHASE[9:0]	Phase shift of current channel I3. From 0 · (ADC clock cycle) to +1023 · (ADC clock cycles). “1000000000” corresponds to an in-phase signal with voltage V0. When DCHPENI3 = 1, it is recommended to adjust the phase shift in steps of 32.

14 Algorithms for calculating final results and their correspondence to external signals

All parameters of the calculated values depend on the IC connection type, as well as on the output data format. Figures 24 and 25 show two ADC connection types: fully differential and non-differential.

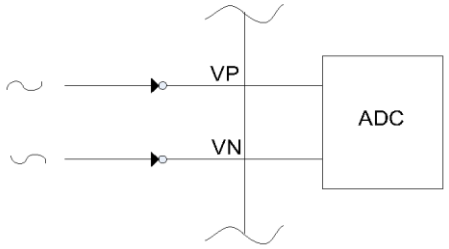


Figure 24 – Differential connection

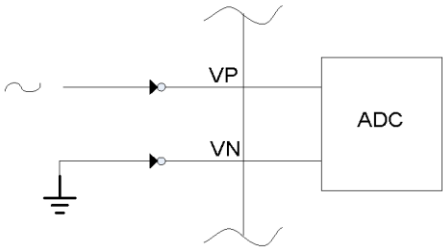


Figure 25 – Non-differential connection

To compensate for the phase shift in the current channel relative to the voltage channel, the system includes a configurable delay line, as shown in Figure 26.

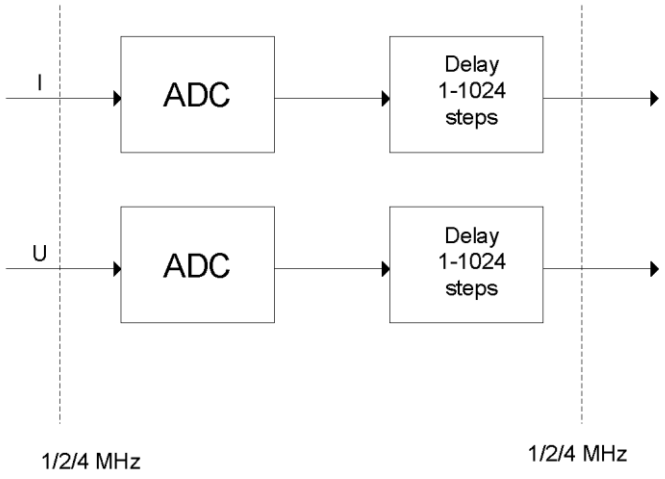


Figure 26 – Controlled phase shift in current and voltage channels

By adjusting the delay line in the voltage channel, the time offset of one channel relative to the other can be controlled. Since the sampling frequency after the ADC is 1/2/4 MHz, one step at an input signal frequency of 50 Hz corresponds to 0.018°/0.009°/0.0045°. It should be noted that in this system the shift is performed in the time domain; therefore, the phase shift expressed in degrees depends on the input signal frequency.

A *sinc*^{3}-structure filter is used as the decimation filter; its characteristics are shown in Figures 27 and 28.

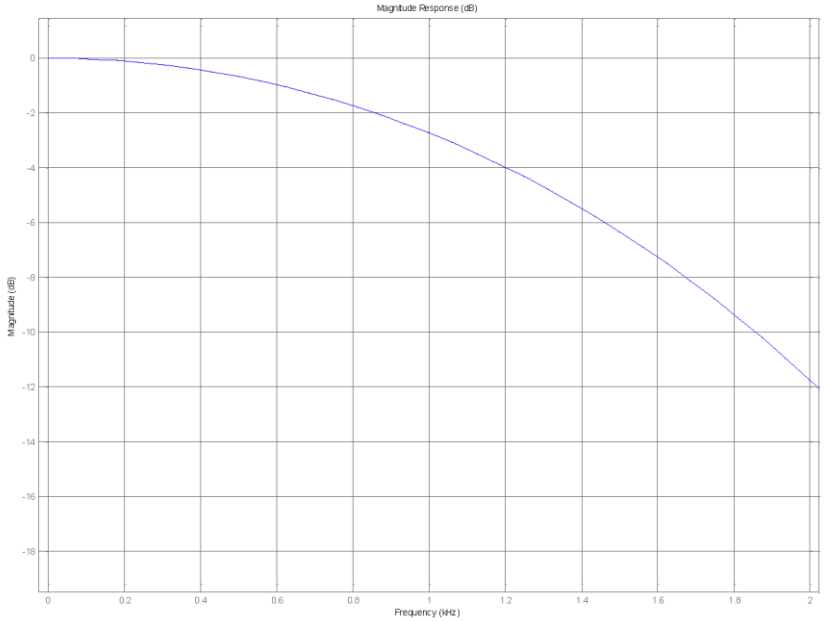


Figure 27 – Decimation filter response in the 2 kHz band

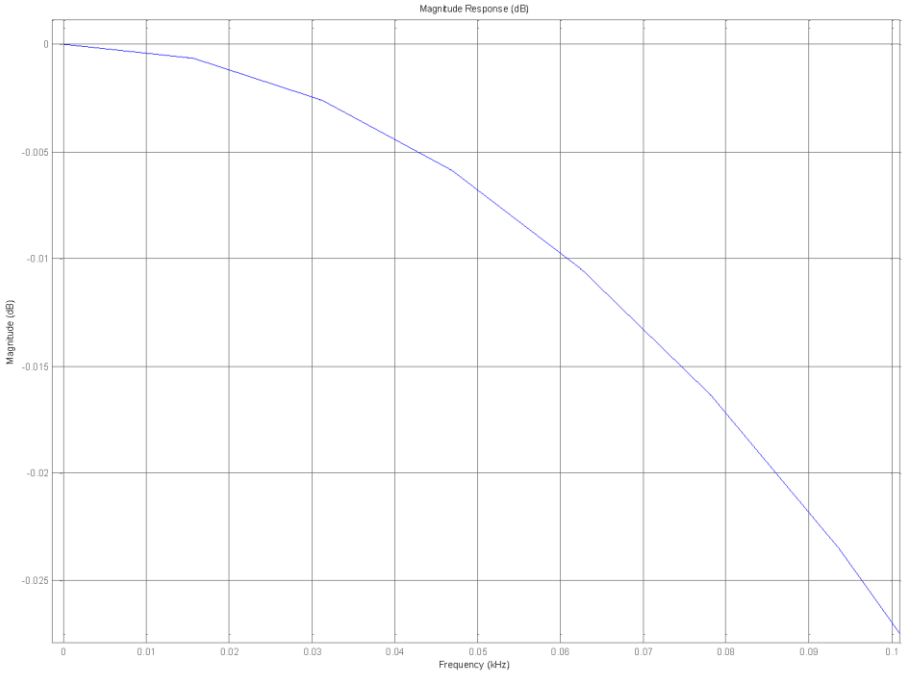


Figure 28 – Decimation filter response in the 100 Hz band

As can be seen from the characteristics above, the filter attenuates frequencies close to 2 kHz by up to 12 dB, which should be taken into account when measuring harmonics of the fundamental tone. In the band up to 100 Hz, the attenuation is insignificant (at 50 Hz, the attenuation is 0.008 dB).

To compensate for the effect of the offset in the current and voltage channels on the calculated power, a high-pass filter is placed in the current channel after the decimation filter.

This filter removes the DC component of the signal. The default cutoff frequency of the filter is 1 Hz and can be adjusted using CHPF_COEFF.

Each ADC channel has an eight-sample FIFO buffer intended for storing current or voltage samples, as well as samples of active power, reactive power, and apparent power (only for channels I0–I3). The sample source for the FIFO buffer is selected in the F0I3SEL field of the ADCUI_F0AC register for current channel I3, and in the FxISEL and FxVSEL fields of the ADCUI_FxMD0 register for the remaining channels. The selected samples are written to the FIFO buffer at the oversampling frequency specified in the OSR_CONF field of the ADCUI_CTRL1 register.

If the FIFO buffers of the channels are configured to receive current and voltage samples (FxISEL = 00, FxVSEL = 00), the FIFO sample values can be converted to input voltages using the formulas given in Table 233. Samples written to the FIFO are represented in two's-complement binary format.

Table 233 – Input voltage calculation formulas

Mode	ADC input voltage, V	
24-bit mode	$\frac{FxVDAT \cdot UREFO}{FxVGAIN \cdot 2^{24}}$,	$\frac{FxIDAT \cdot UREFO}{\text{FxIyGAIN} \cdot 2^{24}}$
16-bit mode	$\frac{FxVDAT \cdot UREFO}{FxVGAIN \cdot 2^{16}}$,	$\frac{FxIDAT \cdot UREFO}{\text{FxIyGAIN} \cdot 2^{16}}$

Each current channel can be corrected using the IxGAIN and IxBGAIN coefficients in accordance with formula (4). The IxBGAIN value is stored in two's-complement binary format.

$$I_{cor} = I_{ADC} \cdot 2^{IxGAIN} \cdot \left(1 + \frac{\text{IxBGAIN}}{2^{15}}\right). \quad (4)$$

If the signal source for the third current channel is selected before the high-pass filter (F0I3SEL = 1), the value stored in the ADCUI_F0I3DAT register does not include the IxBGAIN coefficient.

Each voltage channel can be corrected using the VxGAIN and VxBGAIN coefficients in accordance with formula (5). The VxBGAIN value is stored in two's-complement binary format.

$$V_{cor} = V_{ADC} \cdot 2^{VxGAIN} \cdot \left(1 + \frac{\text{VxBGAIN}}{2^{15}}\right). \quad (5)$$

The FxVRMS and FxIRMS registers contain the calculated RMS values of voltage and current for the corresponding phase. Table 234 lists the RMS values.

Table 234 – RMS voltage values

Mode	RMS ADC input voltage, V	
Always 24 bits, independent of the RESOL bit value	$\frac{FxVRMS \cdot UREFO}{FxVGAIN \cdot 2^{24}}$,	$\frac{FxIRMS \cdot UREFO}{\text{FxIyGAIN} \cdot 2^{24}}$

To calculate the RMS value, the algorithm described below is used (the voltage channel is shown as an example; the algorithm for the current channel is identical).

The input signal is represented as

$$V(t) = \sqrt{2} \cdot V_{rms} \cdot \sin(\omega t) . \quad (6)$$

Voltage samples are acquired at a default rate of 4 kHz, and the RMS values are updated at the same rate. Each sample is squared, which for an AC signal yields the following result

$$V^2(t) = 2 \cdot V^2{}_{rms} \cdot \sin^2(\omega t) = V^2{}_{rms} - V^2{}_{rms} \cdot \cos(2\omega t) . \quad (7)$$

Thus, the resulting signal contains a DC component equal to the RMS voltage value and a ripple at twice the frequency of the input signal. To suppress this ripple, the resulting signal is passed through a low-pass filter with a default cutoff frequency of 0.5 Hz. For a 50 Hz input signal, this filter suppresses parasitic ripple at 100 Hz (50 Hz • 2) with a factor of 46 dB. The filtered signal is then fed to the square-root extraction block. The resulting signal still contains ripple, but it is attenuated by the filter. Therefore, it is recommended to use the mode in which updating of the RMS value is synchronized with the moment when the voltage crosses 0 (ZXRMS = 1).

After square-root extraction, the RMS offset can be corrected using the 12-bit FxVRMSOS and FxIRMSOS values according to formula (8). Before correction, the value is shifted left by 4 bits.

$$\text{FxVRMS} = \text{FxVRMS} + \text{FxVRMSOS} \cdot 16 . \quad (8)$$

The FxVRMSOS and FxIRMSOS values are represented as signed two’s-complement binary numbers. As part of channel calibration, it is recommended to set FxVRMSOS to the average value of the FxVRMS register with the ADC inputs of the corresponding channel shorted, over an interval of at least 4096 samples after the HPF and LPF filters have settled. This value should be determined for all four PGA modes. The values should be written to the register depending on the PGA mode and taking into account the digital gains IxGAIN, VxGAIN, IxBGAIN, and VxBGAIN. This correction is required to eliminate the error caused by ADC noise, which is converted into a DC offset after squaring.

The FxVRMS2 and FxIRMS2 registers store the most significant 32 bits of the RMS voltage and current values prior to square-root extraction. All 48 bits are used internally in the IC for calculating FxVRMS and FxIRMS. Table 235 lists the squared RMS values.

Table 235 – Squared RMS values

Mode	Squared ADC input voltage, V²
Always 32 bits, independent of the RESOL bit value	$\frac{FxVRMS2 \cdot UREFO \cdot UREFO}{FxVGAIN \cdot FxVGAIN \cdot 2^{32}}$, $\frac{FxIRMS2 \cdot UREFO \cdot UREFO}{FxIyGAIN \cdot FxIyGAIN \cdot 2^{32}}$

To calculate reactive power, the signal in the current channel must be shifted by 90°. This is achieved using filters that maintain a 90° phase shift for both channels over a sufficiently wide frequency range. Figure 29 shows the phase response of this filter.

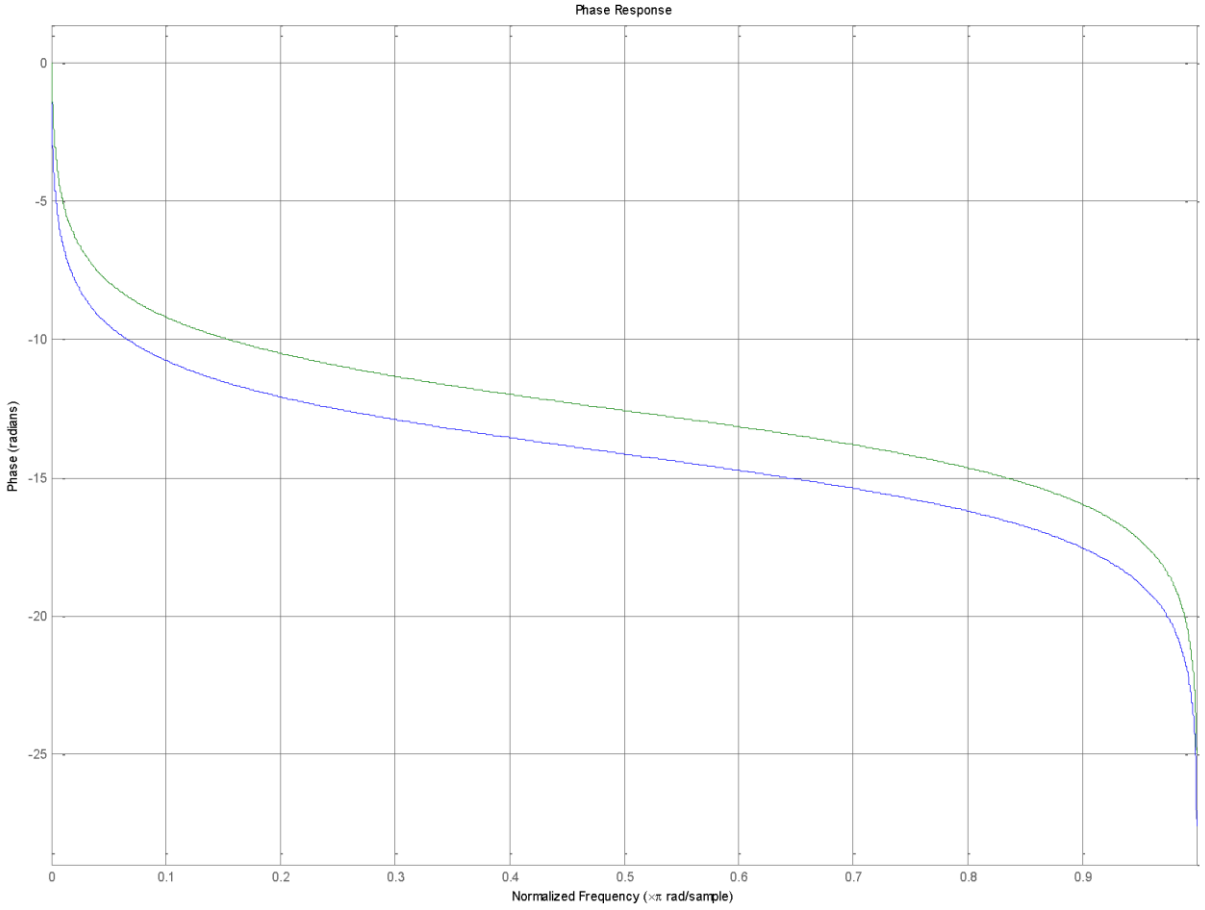


Figure 29 – Filter for shifting the signal by 90°

To calculate active and reactive energy, an approach similar to RMS calculation is used, but without square-root extraction.

$$V(t) = \sqrt{2} \cdot V_{rms} \cdot \sin(\omega t)\,, \tag{9}$$

$$I(t) = \sqrt{2} \cdot I_{rms} \cdot \sin(\omega t)\,. \tag{10}$$

In this case, the instantaneous power value is equal to the product of current and voltage.

$$P(t) = V(t) \cdot I(t) = V_{rms} \cdot I_{rms} - V_{rms} \cdot I_{rms} \cdot \cos(2\omega t)\,. \quad (11)$$

The average power value over an integer number of cycles is equal to

$$P = \frac{1}{nT}\int_0^{nT} P(t)dt = V_{rms} \cdot I_{rms}\,. \quad (12)$$

Thus, the instantaneous power value corresponds to the DC component of the product of current and voltage. To extract the DC component, a low-pass filter with a default cutoff frequency of 0.5 Hz is used. The instantaneous power value can be obtained from the *FxVDAT* and *FxIDAT* FIFOs.

If the channel FIFOs are configured to receive power samples (FxISEL = 01/10/11, FxVSEL = 01/10), the FIFO receives the results of multiplying the corresponding current and voltage values, shifted right by 23 bits. Internally, all 48 bits are used for further accumulation in the accumulators. For example, if instantaneous apparent power samples are written to the FIFO, the relation FxVDAT = FxVRMS·FxIRMS/2^{23} applies. The same applies when receiving instantaneous active and reactive power samples. Samples written to the FIFO are represented in two's-complement binary format.

Table 236 – Conversion of FIFO sample values to power values

Mode	Power, V·A
24-bit mode	$\frac{FxVDAT \cdot 2^{23} \cdot KR \cdot UREFO \cdot UREFO}{FxIGAIN \cdot FxVGAIN \cdot 2^{48}}$, $\frac{FxIDAT \cdot 2^{23} \cdot KR \cdot UREFO \cdot UREFO}{FxIGAIN \cdot FxVGAIN \cdot 2^{48}}$
KR – current-to-voltage conversion coefficient. Since the ADCs in both the "current channel" and the "voltage channel" are identical and convert voltage into a digital code, this coefficient is determined by the external components. For example, when using a 1 mΩ shunt resistor, KR = 1000.	

Each power channel has independent gain calibration (12 bits): FxWGAIN, FxVARGAIN, and FxVAGAIN. The active and reactive power channels have independent offset calibration (16 bits): FxWATTOS and FxVAROS. The correction is performed according to the formula

$$P_{cor} = (P + 2 \cdot \mathrm{xOS}) \cdot \left(1 + \frac{xGAIN}{2^{11}}\right), \quad (13)$$

where P is the corresponding instantaneous power when read from the FIFO.

Offset correction is applied first, followed by multiplication.

The calculated power values are then accumulated in accumulator registers. Each of the three power types has its own accumulator. The values in these registers are updated on the ZX event, i.e., on a zero crossing in the voltage channel, and represent the consumed energy. The accumulator register increment is equal to the sum of all samples of the corresponding instantaneous power between two ZX events, shifted right by 12 bits. For example, if the ADC sample rate is 4000 Hz and the input signal frequency is 50 Hz, the increment is equal to the sum of 80 instantaneous power samples divided by 2^{12}. This sum is the value that is checked against the minimum thresholds defined by the VANOLOAD/VARNOLOAD/APNOLOAD registers. Table 237 shows the formula for converting the accumulator register value to W·s.

Table 237 – Formula for conversion to W·s

Mode	Energy, W·s
Independent of the value of the RESOL bit.	$\frac{FxVAHR \cdot 2^{12} \cdot KR \cdot UREFO \cdot UREFO}{Fadc \cdot FxIGAIN \cdot FxVGAIN \cdot 2^{48}}$
Fadc is the ADC output data update rate after the SINC3 filter, 4000 Hz by default; FxVAHR is the apparent energy accumulator, all 57 bits. The same applies to the other energy values.	

14.1. **Typical connection diagram for three-phase electric power metering**

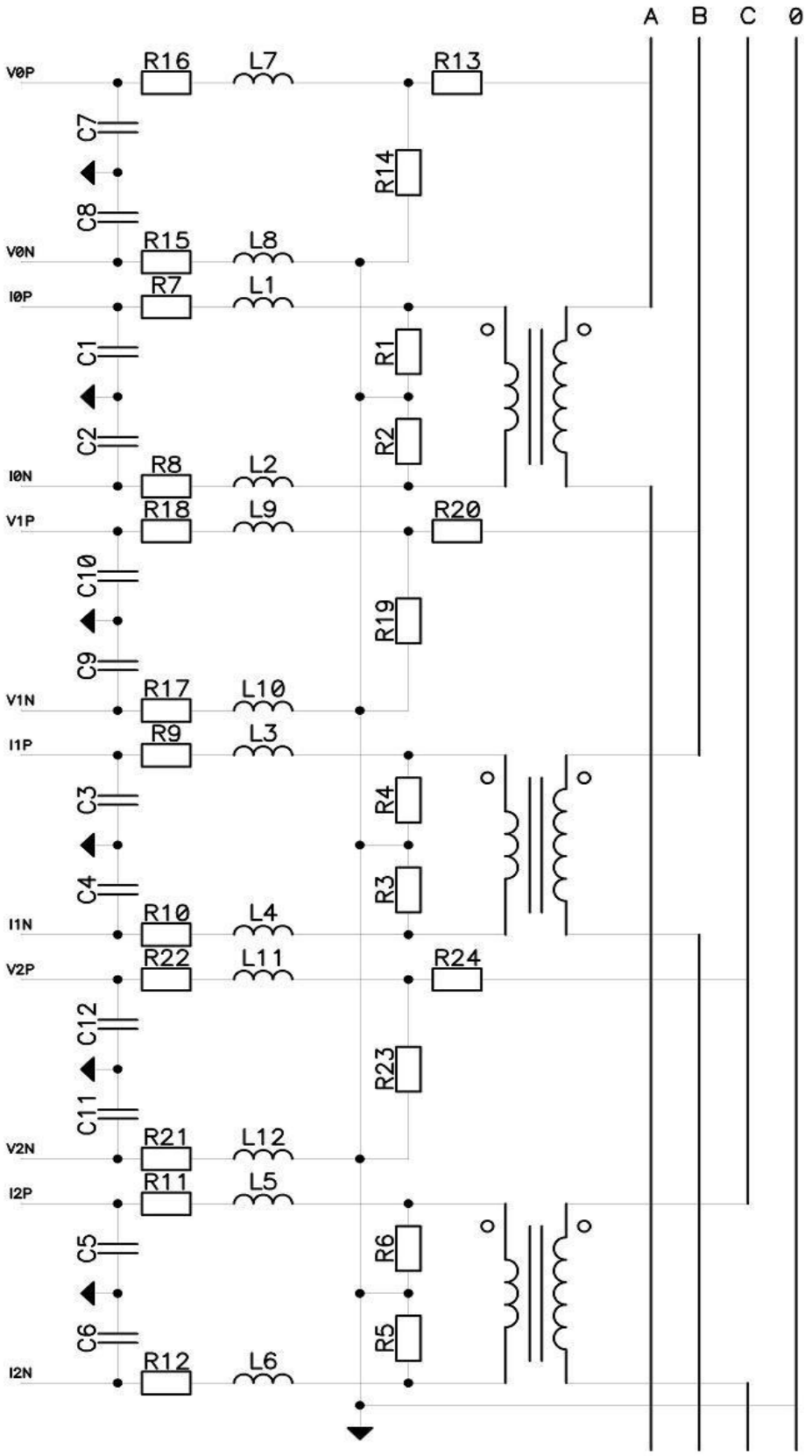


Figure 30 – Typical connection diagram for three-phase electric power metering

This connection diagram can serve as an example for designing three-phase electricity meters. A current transformer is installed on each of the three phases. The outputs of the transformers are connected to a resistive divider. The total resistance of these resistors, for example R1+R2, must match the required load at the transformer output. Both resistors have equal resistance values and form a midpoint relative to which the current is measured. In this connection diagram, the midpoint acts as the analog ground. This midpoint is connected to the neutral line of the three-phase network, if it is used. A resistive divider is used to supply the signal to the voltage channel. When selecting the transformer and calculating the resistive divider, it is important to ensure that the signal amplitude at the ADC input shall not exceed ±500 mV relative to the analog ground of the integrated circuit. A filter must be placed directly before the ADC channel inputs to eliminate aliasing effects. It should be designed for a cutoff frequency approximately one and a half times higher than the ADC sampling frequency. In this example, it is a simple first-order RC low-pass filter. Before the RC filter, inductors should also be installed to suppress radio-frequency interference. These inductors are not functionally part of the anti-aliasing filter and are selected to provide broadband suppression of radio-frequency noise. You may use alternative filter configurations depending on the requirements of the final product.

14.2. **Typical connection scheme for single-phase energy metering**

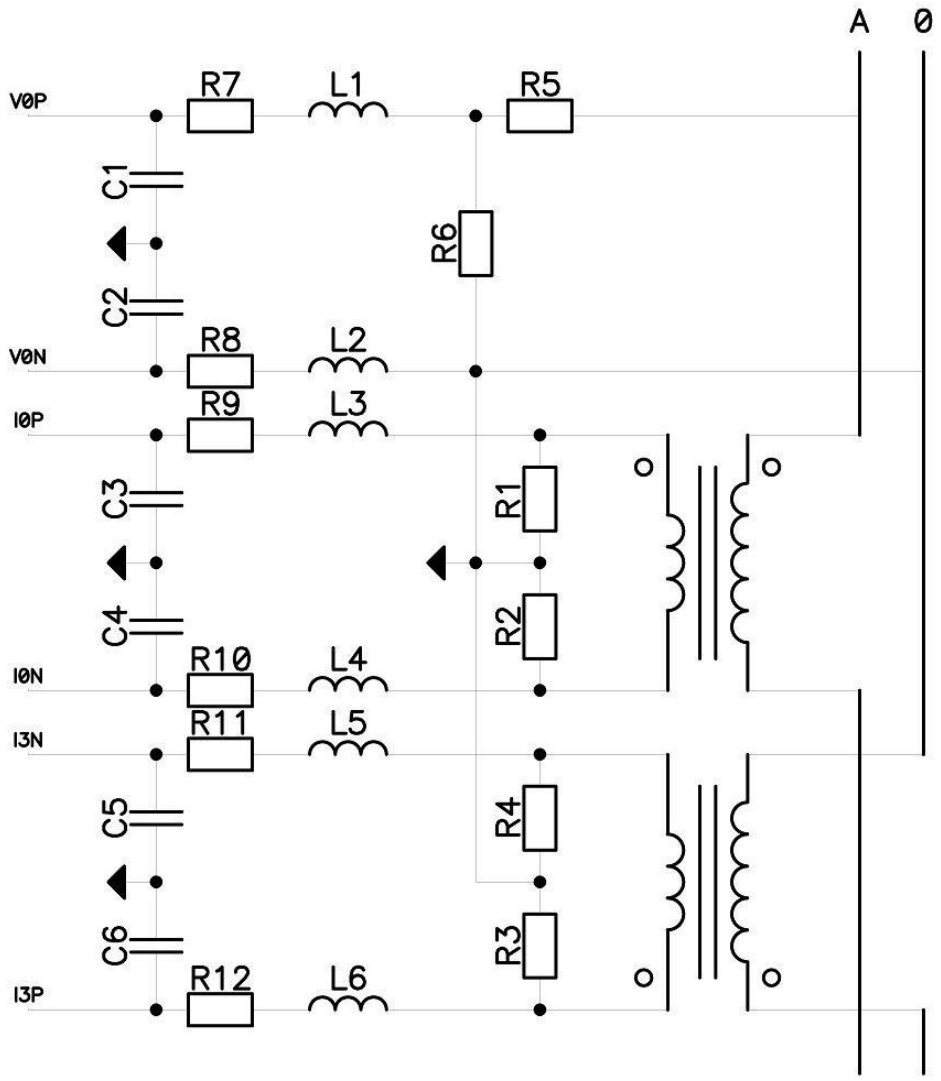


Figure 31 – Typical connection scheme for single-phase energy metering

When a single-phase electricity meter based on this IC is required, the scheme shown above can be used as a reference. In this scheme, current measurement is performed in both conductors of the single-phase network. The IC automatically accounts for the larger current value from channels I0 and I3. If current measurement in the neutral conductor is not required, the part of the circuit related to channel I3 can be omitted. In all other respects, the function of the elements in this scheme is similar to that of the three-phase energy metering scheme.

14.3. Typical connection scheme for single-phase energy metering using a shunt

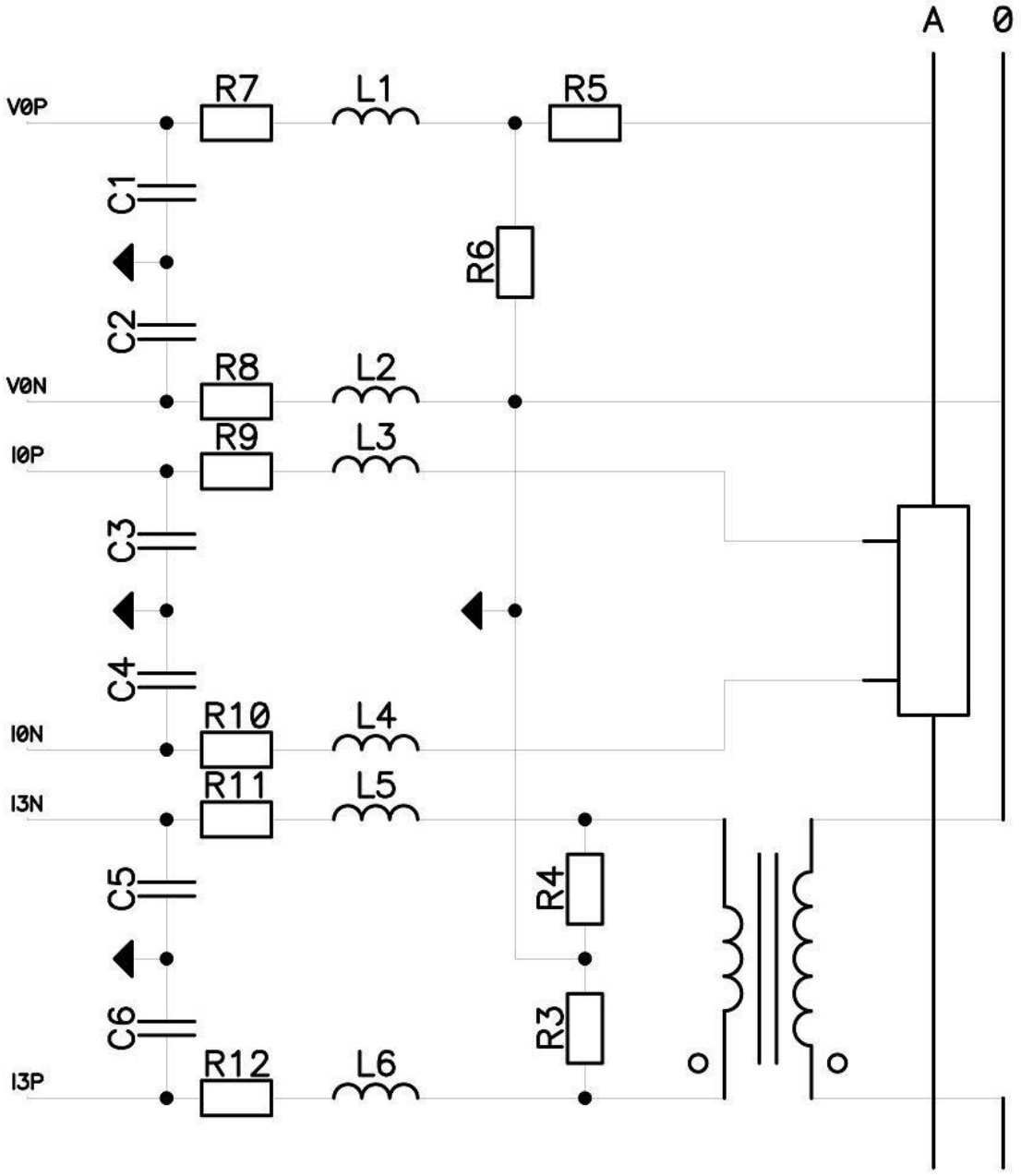


Figure 32 – Typical connection scheme for single-phase energy metering using a shunt

A shunt can also be used for current measurement. In this case, the scheme shown in the figure above should be used. Similarly to the previous transformer-based scheme, the part of the circuit related to current channel I3 can be omitted if current monitoring in the neutral conductor is not required.

15 Hardware CRC computation block

The IC includes a block for computing a 16-bit CRC with an arbitrary polynomial. The controller accepts 32-bit words and can process them either in forward order (starting from the least significant bit) or in reverse order (starting from the most significant bit). The computation speed is 2 bits per PCLK (APB bus frequency). The controller includes a FIFO with a depth of 4 samples, as well as a DMA channel for loading new words. A DMA request is generated when the FIFO is empty. The controller starts processing new words as soon as they appear in the FIFO and continues processing through the last word. After each word is processed, a flag is set. The CRC register supports both read access (to read the calculated value) and write access (to set the initial value).

15.1. Description of CRC block control registers

Table 238 – Description of CRC block control registers

Base Address	Name	Description
0x5009_8000	CRC	CRC controller
Offset		
0x00	CRC_CTRL	Global control for the CRC controller
0x04	CRC_STAT	CRC block status
0x08	CRC_DATAI	Input data FIFO register
0x0C	CRC_VAL	Calculated CRC register
0x10	CRC_POL	CRC calculation polynomial

15.1.1. CRC_CTRL

Table 239 – CRC_CTRL Register

Number	31…7	6…5	4…3	2	1	0
Access		R/W	R/W	R/W	R/W	R/W
Reset		00	00	0	0	0
	-	DCSize	DLSize	DMAEN	DATAINV	CRCEN

Table 240 – Description of the CRC_CTRL register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...7	-	Reserved
6...5	DCSize	Data size for CRC calculation: 00 – calculation for a byte (8 bits); in this case DLSize may be 00, 01, or 10; 01 – calculation for a halfword (16 bits); in this case DLSize may be 01 or 10; 10 – calculation for a word (32 bits); in this case DLSize may only be 10.

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
4…3	DLSize	Size of loaded data: 00 – byte (8 bits); the loaded byte is written to CRC_DATAI[7:0]; 01 – halfword (16 bits); the loaded halfword is written to CRC_DATAI[15:0]; 10 – word (32 bits); the loaded word is written to CRC_DATAI[31:0].
2	DMAEN	Enable DMA request generation: 0 – request not generated; 1 – request generated.
1	DATAINV	CRC computation order: 0 – starting from the least significant bit; 1 – starting from the most significant bit.
0	CRCEN	Enable block operation: 0 – block disabled; 1 – block enabled.

15.1.2. **CRC_STAT**

Table 241 – CRC_STAT Register

Number	31:4	3	2	1	0
Access		R/W	R	R	R
Reset		0	0	0	0
	-	FIFOOVER	FIFOEMPTY	FIFOFULL	CONVCOMP

Table 242 – Description of the CRC_STAT register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...4	-	Reserved
3	FIFOOVER*	FIFO overrun: 0 – normal operation; 1 – a write occurred while the FIFO was full, resulting in data loss.
2	FIFOEMPTY	FIFO empty: 0 – the FIFO contains at least one valid entry; 1 – the FIFO is empty.
1	FIFOFULL	FIFO full: 0 – the FIFO has at least one free entry; 1 – the FIFO has no free entries.
0	CONVCOMP	CRC calculation complete: 0 – calculation in progress; 1 – calculation of the data word is complete or the block is disabled.
* The bit is cleared by writing “1” to bit [3].		

15.1.3. **CRC_DATAI**

Table 243 – CRC_DATAI Register

Number	31…0
Access	W
Reset	
	DATA_IN

Table 244 – Description of the CRC_DATAI register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...0	DATA_IN	Register for writing a new sample to the FIFO

15.1.4. **CRC_VAL**

Table 245 – CRC_VAL Register

Number	31…16	15…0
Access		R/W
Reset		0000000000000000
	-	CRCOUT

Table 246 – Description of the CRC_VAL register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...16	-	Reserved
15...0	CRCOUT	Calculated value/initial value. The initial value must be written when the block is disabled or when the computation has completed

15.1.5. **CRC_POL**

Table 247 – CRC_POL Register

Number	31…17	16…0
Access		R/W
Reset		1000000000000001
	-	CRCPOL

Table 248 – Description of the CRC_POL register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...17	-	Reserved
16...0	CRC_POL	Polynomial for calculating CRC. Since this is a 16-bit CRC, the least significant and most significant bits are always set to “1” and cannot be changed.

The resulting polynomial is:

$$f(x) = x^{16} + x^{CRC\_POL[15]} + x^{CRC\_POL[14]} + \cdots + x^{CRC\_POL[2]} + x^{CRC\_POL[1]} + 1$$

16 Clock signals

The IC includes two internal oscillators, two external oscillators, and a dedicated clock generation and distribution block.

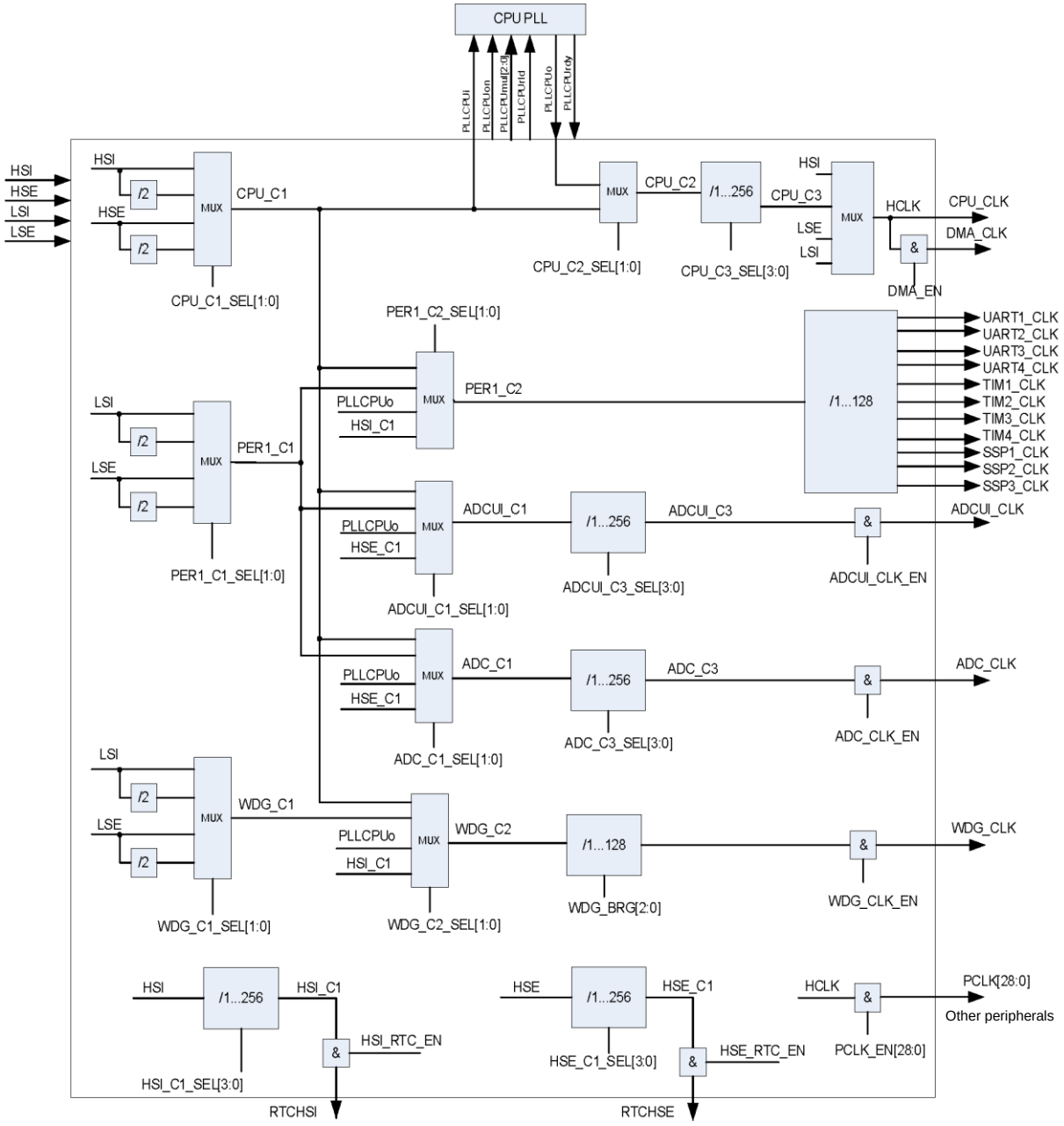


Figure 33 – Block diagram of the clock generation system

Clock frequency control is performed via the RST_CLK peripheral block. When power is applied, the IC starts operating at the HSI/16 frequency of the oscillator. Clock signals for all peripheral blocks except RST_CLK are disabled. To start operating with a required peripheral block, its clock must be enabled in the PER2_CLOCK register. Some interface controllers (UART, SSP, Timers) can operate at frequencies different from the processor core frequency; therefore, their operating rates can be configured in the corresponding registers (PER1_CLOCK, UART_CLOCK, SSP_CLOCK, TIM_CLOCK). To change the core clock frequency, it is possible to switch to another oscillator and/or use the clock frequency multiplier block. For correct clock frequency switching, the required clock

frequencies must first be generated, and then the switch to them must be performed using the appropriate multiplexers controlled by the CPU_CLOCK register. The registers of the clock controller block are not reset by the internal ndmreset reset signal generated by the debugger.

To switch from one clock source to another, both clock sources must remain enabled.

For the IRL1206FI (CHIP_ID = 217), when changing the processor core clock frequency, the number of wait cycles for Flash memory access, defined by the Delay field in the FLASH_CMD register of the FLASH_CNTRL Flash memory controller, must be configured. If the configured processor core frequency is higher than the current one, this setting must be applied before changing the frequency; otherwise, it must be applied after the frequency change.

16.1. **Internal RC HSI oscillator**

The HSI oscillator generates an 8 MHz clock signal. The oscillator starts automatically when the supply voltage U_{CC} is applied and, upon entering normal operating mode, asserts the HSI_RDY signal in the BKP_CLK register of the backup domain. Initially, the processor core starts operating at the HSI/16 clock frequency. During further operation, the HSI oscillator can be disabled using the HSI_ON signal in the BKP_CLK register. The oscillator can also be trimmed using the HSI_LSI_TRIM signal in the BKP_CLK register.

16.2. **Internal RC LSI oscillator**

The LSI oscillator generates a 32 kHz clock signal. The oscillator is enabled by the LSI_ON signal in the BKP_CLK register and, upon entering normal operating mode, asserts the LSI_RDY signal in the BKP_CLK register.

The oscillator can also be trimmed using the HSI_LSI_TRIM signal in the BKP_CLK register. The LSI oscillator frequency is equal to HSI/256.

16.3. **External HSE oscillator**

The HSE oscillator is intended to generate the clock frequency f_{C_HSE} in the range of 8…16 MHz using an external crystal resonator. The recommended total load capacitance at the integrated circuit pin (PCB parasitic capacitance + the value of the soldered capacitor) is 24 pF and 16 pF for 8 MHz and 16 MHz resonators, respectively. The oscillator starts when the supply voltage U_{CC} is applied and the HSEON enable signal is set in the HS_CONTROL register. When the differential amplitude across the resonator reaches the minimum level required for the clock signal to appear at the output of the HSE block inside the integrated circuit, the HSERDY signal is set in the CLOCK_STATUS register. The assertion of the HSERDY bit does not indicate that all transient processes associated with the resonator amplitude have fully settled. The frequency/duty cycle at the output of the HSE block may slightly differ from the nominal values. For most applications, this is sufficient. The time required for the oscillator to finally reach its nominal operating mode depends primarily on the characteristics of the crystal resonator and can be estimated using the following formula.

$$t_{start} = \mathrm{K} \cdot \frac{Q \cdot R_M}{2 \cdot \pi \cdot quartz\_freg \cdot \left(\frac{4 \cdot g_m \cdot C_L^2}{(g_m \cdot C_s')^2 + 16 \cdot \omega^2 \cdot (C_L^2 + C_s' \cdot C_L)^2} - R_M \right)}. \quad (14)$$

Code for calculation in CAD tools:

*tstart = K * ((Q * Rm) / (2 * pi * quartz_freq * ((4 * gm * (CL)^2) / ((gm*Cs)^2 + 16* (2 * pi * quartz_freq)^2 * ((CL)^2 + Cs * CL)^2) - Rm))*

A calculation for a typical 8 MHz Geyer KX-7 T crystal resonator, used on the evaluation boards, with parameters Rm = 250 Ohm (500 Ohm maximum, the average value is used for the calculation), Cs = 3.5 pF, Cl = 24 pF, for CHIP_ID = 217, gives an approximate start-up time t_{start} = 5.7 ms (the quality factor Q is 75000).

The parameters required for the calculation using formula (14), and the method for obtaining them, are given in Table 249.

Table 249 – Parameters for calculating the start-up time t_{start} using formula (14)

Formula parameter	Symbol	Method for obtaining the parameter value
Proportionality coefficient	K	12 – for CHIP_ID = 217
Crystal resonator quality factor	Q	If no data is provided in the resonator specification, assume a value of 50,000–100,000
Dynamic resistance of the crystal resonator	R_M	From the resonator specification
Tuning frequency of the crystal resonator	quartz_freq	From the resonator specification
Transconductance of the amplifying transistor	g_m	8.24 mA/V – for CHIP_ID = 217
Load capacitance of the crystal resonator	C_L	Based on the nominal values of the capacitors installed on the PCB
Shunt capacitance of the crystal resonator	C_s'	$C_s' \approx C_0$ where C_0 is taken from the crystal resonator specification
Angular frequency	ω	$\omega = 2 \cdot \pi \cdot quartz\_freg$

For the IRL1206FI (CHIP_ID = 217), the oscillator can also operate in HSEBYP mode, in which the input clock signal applied to the OSC_IN pin is routed directly to the HSE output. In this mode, the OSC_OUT pin is high impedance. The allowable frequency range f_{C_HSE} in HSEBYP mode is 0 to 48 MHz.

16.4. **External LSE oscillator**

The LSE oscillator is intended to generate a 32 kHz clock frequency using an external crystal resonator. The recommended total load capacitance at the integrated circuit pin (PCB parasitic capacitance + the value of the soldered capacitor) is 8 to 12 pF. The oscillator starts when the supply voltage U_{CCB} is applied and the LSE_ON enable signal is set in the BKP_CLK register. When the differential amplitude across the resonator reaches the minimum level required for the clock signal to appear at the output of the LSE block inside the integrated circuit, the LSE_RDY signal is set in the BKP_CLK register. The assertion of the LSE_RDY bit does not indicate that all transient processes associated with the resonator amplitude have fully settled. The frequency/duty cycle at the output of the HSE block may slightly differ from the nominal values. For most applications, this is sufficient. The time required for the oscillator to finally reach its nominal operating mode depends primarily on the characteristics of the crystal resonator and can be estimated using the following formula.

$$t_{start} = \mathrm{K} \cdot \frac{Q \cdot R_M}{2 \cdot \pi \cdot quartz\_freg \cdot \left(\frac{4 \cdot g_m \cdot C_L^2}{(g_m \cdot C_s')^2 + 16 \cdot \omega^2 \cdot (C_L^2 + C_s' \cdot C_L)^2} - R_M \right)}. \quad (15)$$

Code for calculation in CAD tools:

*tstart = K * ((Q * Rm) / (2 * pi * quartz_freq * ((4 * gm * (CL)^2) / ((gm*Cs)^2 + 16* (2 * pi * quartz_freq)^2 * ((CL)^2 + Cs * CL)^2) - Rm))*

A calculation for a typical 32.768 kHz Geyer KX-327NHT crystal resonator, used on evaluation boards, with parameters Rm = 40 kOhm (80 kOhm maximum, the average value is used for the calculation), Cs = 1.6 pF, Cl = 12 pF, for CHIP_ID = 217, gives an approximate start-up time t_{start} = 625 ms (the quality factor Q is taken as 75,000).

The parameters required for the calculation using formula (15) and the method for obtaining them are shown in Table 250.

Table 250 – Parameters for calculating the start-up time t_{start} using formula (15)

Formula parameter	Symbol	Method for obtaining the parameter value
Proportionality coefficient	K	12 – for CHIP_ID = 217
Crystal resonator quality factor	Q	If no data is provided in the resonator specification, assume a value of 50,000–100,000
Dynamic resistance of the crystal resonator	R_M	From the resonator specification
Tuning frequency of the crystal resonator	quartz_freq	From the resonator specification
Transconductance of the amplifying transistor	g_m	694 µA/V – for CHIP_ID = 217
Load capacitance of the crystal resonator	C_L	Based on the nominal values of the capacitors installed on the PCB
Shunt capacitance of the crystal resonator	C_s'	$C_s' \approx C_0$ where C_0 is taken from the crystal resonator specification
Angular frequency	ω	$\omega = 2 \cdot \pi \cdot quartz\_freg$

The oscillator can also operate in LSE_BYP mode, in which the input clock signal applied to the OSC_IN32 pin is routed directly to the LSE output. The allowable frequency range in LSEBYP mode is 0 to 1 MHz. In this mode, the OSC_OUT32 pin is high impedance. In this mode, it is recommended to set LSE_CONF[1:0] = 01 to enable the "glitch filter."

Since the LSE oscillator is powered by the supply voltage U_{CCB} and its control register BKP_CLK is located in the battery domain, the oscillator can continue operating when the main supply voltage U_{CC} is lost. The LSE oscillator is used to operate the real-time clock and calendar.

16.5. On-chip system clock frequency multiplication block

The multiplication block allows the input clock frequency to be multiplied by a factor of 2 to 8, selected by the PLLCPUMUL[2:0] field in the PLL_CONTROL register. The input frequency of the multiplier block must be in the range of 8-16 MHz; the output frequency must be up to 50 MHz for the IRL1206FI (CHIP_ID = 217). When the clock multiplication block reaches its nominal operating mode, the PLLCPURDY signal is asserted in the CLOCK_STATUS register. The block is enabled by the PLLCPUON signal in the PLL_CONTROL register. The output frequency can be used as the main clock for the processor and peripherals.

16.6. **Clock controller register description**

Table 251 – Clock controller register description

Base Address	Name	Description
0x5002_0000	RST_CLK	Clock controller
Offset		
0x00	CLOCK_STATUS	Clock controller status register
0x04	PLL_CONTROL	Frequency multiplication block control register
0x08	HS_CONTROL	High-frequency generator and oscillator control register
0x0C	CPU_CLOCK	Processor core clock control register
0x10	PER1_CLOCK	Peripheral clock control register
0x14	ADC_CLOCK	ADC and ΔΣ ADC clock control register
0x18	RTC_CLOCK	RTC block high-frequency clock generation control register
0x1C	PER2_CLOCK	Peripheral clock control register
0x20	DMA_DONE_STICK	DMA completion signal latch register
0x24	TIM_CLOCK	TIMER clock control register
0x28	UART_CLOCK	UART clock control register
0x2C	SSP_CLOCK	SSP clock control register
0x30	DIV_SYS_TIM	System core timer clock divider selection register

16.6.1. **CLOCK_STATUS**

Table 252 – CLOCK_STATUS Register

Number	31…3	2	1	0
Access	U	RO	RO	U
Reset	0	0	0	0
	-	HSE_RDY	PLL_CPU_RDY	

Table 253 – Description of the CLOCK_STATUS register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...3	-	Reserved
2	HSE_RDY	HSE oscillator ready flag: 0 – the oscillator is not running or has not stabilized; 1 – the oscillator is running and stable
1	PLL_CPU_RDY	CPU PLL ready flag: 0 – the PLL is not running or has not stabilized; 1 – the PLL is running and stable
0	-	Reserved

16.6.2. **PLL_CONTROL**

Table 254 – PLL_CONTROL Register

Number	31…11	10…8	7…4	3	2	1	0
Access	U	R/W	U	R/W	R/W	U	U
Reset	0	000	0000	0	0	0	0
	-	PLL_CPU_ MUL[2:0]	-	PLL_CPU_ SEL	PLL_CPU_ ON	-	-

Table 255 – Description of PLL_CONTROL register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...11	-	Reserved
10...8	PLL_CPU_MUL[2:0]	CPU PLL multiplication factor: PLLCPUo = PLLCPUi x (PLLCPUMUL+1)
7...4	-	Reserved
3	PLL_CPU_SEL	PLL input frequency selection restart bit: 0 – frequency from the HSE oscillator; 1 – frequency is selected by the CPU_C1_SEL[1:0] bits of the CPU_CLOCK register
2	PLL_CPU_ON	PLL enable bit: 0 – PLL disabled; 1 – PLL enabled
1	-	Reserved
0	-	Reserved

16.6.3. **HS_CONTROL**

Table 256 – HS_CONTROL Register

Number	31…2	1	0
Access	U	R/W	R/W
Reset		0	0
	-	HSEBYP	HSEON

Table 257 – Description of HS_CONTROL register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...2	-	Reserved
1	HSEBYP	HSE oscillator mode control bit: 0 – oscillator mode; 1 – external generator mode. To operate in external generator mode, the hseon bit must be set to 1.
0	HSEON	HSE oscillator mode control bit: 0 – disabled; 1 – enabled

16.6.4. **CPU_CLOCK**

Table 258 – CPU_CLOCK Register

Number	31…10	9, 8	7…4	3	2	1, 0
Access	U	R/W	R/W	U	R/W	R/W
Reset	0	01	1011	0	0	00
	-	HCLK_SEL[1:0]	CPU_C3_SEL[3:0]	-	CPU_C2_SEL	CPU_C1_SEL[1:0]

Table 259 – Description of CPU_CLOCK register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...10	-	Reserved
9, 8	HCLK_SEL[1:0]	Source select bits for HCLK: 00 – HSI; 01– CPU_C3; 10– LSE; 11– LSI
7…4	CPU_C3_SEL[3:0]	Divider select bits for CPU_C3: 0xxx – CPU_C3 = CPU_C2; 1000 – CPU_C3 = CPU_C2/2; 1001 – CPU_C3 = CPU_C2/4; 1010 – CPU_C3 = CPU_C2/8; … 1111 – CPU_C3 = CPU_C2/256
3	-	Reserved
2	CPU_C2_SEL	Source select bits for CPU_C2: 0– CPU_C1; 1– PLLCPUo
1, 0	CPU_C1_SEL[1:0]	Source select bits for CPU_C1: 00 – HSI; 01– HSI/2; 10– HSE; 11– HSE/2

16.6.5. **PER1_CLOCK**

Table 260 – PER1_CLOCK Register

Number	31..14	13	12…10	9, 8
Access	U	R/W	R/W	R/W
Reset	0	0	000	00
	-	WDG_CLK_EN	WDG_BRG_[2:0]	WDG_C2_SEL[1:0]

Number	7, 6	5	4	3, 2	1, 0
Access	R/W	R/W	R/W	R/W	R/W
Reset	00	1	0	00	00
	WDG_C1_ SEL[1:0]	DMA_EN	KeyResetProg	PER1_C2_ SEL	PER1_C1_SEL[1:0]

Table 261 – Description of PER1_CLOCK register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…14	-	Reserved
13	WDG_CLK_EN	WDG clock frequency enable: 0 – no frequency; 1 – frequency available
12…10	WDG_BRG_[2:0]	WDG clock divider: 000 – WDG_CLK == WDG_C2; 001 – WDG_CLK == WDG_C2/2; 010 – WDG_CLK == WDG_C2/4; … 111 – WDG_CLK == WDG_C2/128
9, 8	WDG_C2_SEL	Source select bits for WDG_C2: 0 00 – CPU_C1; 01– WDG_C1; 10– PLLCPUo; 11– HSI_C1
7, 6	WDG_C1_SEL[1:0]	Source select bits for WDG_C1: 00– LSI; 01– LSI/2; 10– LSE; 11– LSE/2
5	DMA_EN	DMA controller clock enable bit
4	KeyResetProg	Cryptographic key memory reset bit
3, 2	PER1_C2_SEL	PER1_C2 source selection bits: 00 – CPU_C1; 01 – PER1_C1; 10– PLLCPUo; 11– HSI_C1
1, 0	PER1_C1_SEL[1:0]	PER1_C1 source selection bits: 00 – LSI; 01 – LSI/2; 10– LSE; 11– LSE/2

16.6.6. **ADC_CLOCK**

Table 262 – ADC_CLOCK Register

Number	31…14	13	12	11…8	7…4	3, 2	1, 0
Access	U	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0000	0000	00	00
	-	ADC_CLK_EN	ADCUI_CLK_EN	ADC_C3_SEL[3:0]	ADCUI_C3_SEL[3:0]	ADCUI_C1_SEL[1:0]	ADC_C1_SEL[1: 0]

Table 263 – Description of the ADC_CLOCK register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…14	-	Reserved
13	ADC_CLK_EN	ADC CLK clock output enable bit 0 – disabled; 1 – enabled
12	ADCUI_CLK_EN	ADCUI CLK clock output enable bit: 0 – disabled; 1 – enabled
11…8	ADC_C3_SEL[3:0]	Divider select bits for ADC_C3: 0xxx – ADC_C3 = ADC_C1; 1000 – ADC_C3 = ADC_C1/2; 1001 – ADC_C3 = ADC_C1/4; 1010 – ADC_C3 = ADC_C1/8; … 1111 – ADC_C3 = ADC_C1/256
7…4	ADCUI_C3_SEL[3:0]	Divider select bits for ADCUI_C3: 0xxx – ADCUI_C3 = ADCUI_C1; 1000 – ADCUI_C3 = ADCUI_C1/2; 1001 – ADCUI_C3 = ADCUI_C1/4; 1010 – ADCUI_C3 = ADCUI_C1/8; … 1111 – ADCUI_C3 = ADCUI_C1/256
3, 2	ADCUI_C1_SEL[1:0]	ADCUI_C1 source selection bits: 00 – CPU_C1; 01– PER1_C1; 10– PLLCPUo; 11– HSE_C1 As the synchronization clock for the ADCUI block, it is recommended to select HSE = 8.192 MHz as the source without additional dividers.
1, 0	ADC_C1_SEL[1:0]	ADC_C1 source selection bits: 00 – CPU_C1; 01 – PER1_C1; 10– PLLCPUo; 11– HSE_C1

16.6.7. **RTC_CLOCK**

Table 264 – RTC_CLOCK Register

Number	31…10	9	8	7…4	3…0
Access	U	R/W	R/W	R/W	R/W
Reset	0	0	0	0000	0000
	-	HSI_RTC_EN	HSE_RTC_EN	HSI_SEL[1:0]	HSE_SEL[1:0]

Table 265 – Description of RTC_CLOCK register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…10	-	Reserved
9	HSI_RTC_EN	HSI RTC enable bit: 0 – disabled; 1 – enabled
8	HSE_RTC_EN	HSE RTC enable bit: 0 – disabled; 1 – enabled
7…4	HSI_SEL[3:0]	Divider select bits for HSI_C1: 0xxx – HSI_C1 = HSI; 1000 – HSI_C1 = HSI/2; 1001 – HSI_C1 = HSI/4; 1010 – HSI_C1 = HSI/8; … 1111 – HSI_C1 = HSI/256
3…0	HSE_SEL[3:0]	Divider select bits for HSE_C1: 0xxx – HSE_C1 = HSE; 1000 – HSE_C1 = HSE/2; 1001 – HSE_C1 = HSE/4; 1010 – HSE_C1 = HSE/8; … 1111 – HSE_C1 = HSE/256

16.6.8. **PER2_CLOCK**

Table 266 – PER2_CLOCK Register

Number	30…0
Access	R/W
Reset	0x00000010
	PCLK_EN[30:0]

Table 267 – Description of PER2_CLOCK register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
30…0	PCLK_EN[29:0]	Peripheral blocks clock enable bit: 0 – disabled; 1 – enabled PCLK[0] – SSP1 PCLK[1] – UART1 PCLK[2] – UART2 PCLK[3] – FLASH PCLK[4] – RST_CLK PCLK[5] – DMA PCLK[6] – I2C PCLK[7] – UART3 PCLK[8] – ADC PCLK[9] – WWDG PCLK[10] – IWDG PCLK[11] – POWER PCLK[12] – BKP PCLK[13] – ADCUI PCLK[14] – TIMER1 PCLK[15] – TIMER2 PCLK[16] – PORTA PCLK[17] – PORTB PCLK[18] – PORTC PCLK[19] – CRC PCLK[20] – reserved PCLK[21] – reserved PCLK[22] – reserved PCLK[23] – RANDOM PCLK[24] – ISO7816 PCLK[25] – SSP2 PCLK[26] – SSP3 PCLK[27] – TIMER3 PCLK[28] – TIMER4 PCLK[29] – UART4 PCLK[30] – PORTD

16.6.9. **DMA_DONE_STICK**

Table 268 – DMA_DONE_STICK register

Number	31..0
Access	R/C
Reset	0
	DMA_DONE_STICK

Table 269 – Description of the DMA_DONE_STICK register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...0	DMA_DONE_STICK	The register bits are set to 1 upon completion of the corresponding DMA channels 0-31 (dma_done[C]). The register bits are cleared only by writing a zero

16.6.10.**TIM_CLOCK**

Table 270 – TIM_CLOCK Register

Number	27	26	25	24	23…19	18…16	15…8	7…0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	00000	000	00000000	00000000
	TIM4_CLK_EN	TIM3_CLK_EN	TIM2_CLK_EN	TIM1_CLK_EN	TIM4_BRG[4:0]	TIM3_BRG[2:0]	TIM2_BRG[7:0]	TIM1_BRG[7:0]

Table 271 – Description of TIM_CLOCK register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…28	-	Reserved
27	TIM4_CLK_EN	TIM4 clock frequency enable: 0 – no frequency; 1 – frequency available
26	TIM3_CLK_EN	TIM3 clock frequency enable: 0 – no frequency; 1 – frequency available
25	TIM2_CLK_EN	TIM2 clock frequency enable: 0 – no frequency; 1 – frequency available
24	TIM1_CLK_EN	TIM1 clock frequency enable: 0 – no frequency; 1 – frequency available
23…19	TIM_BRG[4:0]*	TIM4 clock divider: xx000 – TIM4_CLK == PER1_C2; xx001 – TIM4_CLK == PER1_C2/2; xx010 – TIM4_CLK == PER1_C2/4; … xx111 – TIM4_CLK == PER1_C2/128

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
18…16	TIM3_BRG[2:0]*	TIM3 clock divider: 000 – TIM3_CLK == PER1_C2; 001 – TIM3_CLK == PER1_C2/2; 010 – TIM3_CLK == PER1_C2/4; … 111 – TIM3_CLK == PER1_C2/128
15…8	TIM2_BRG[7:0]*	TIM2 clock divider: xxxxx000 – TIM2_CLK == PER1_C2; xxxxx001 – TIM2_CLK == PER1_C2/2; xxxxx010 – TIM2_CLK == PER1_C2/4; … xxxxx111 – TIM2_CLK == PER1_C2/128
7…0	TIM1_BRG[7:0]*	TIM1 clock divider: xxxxx000 – TIM1_CLK == PER1_C2; xxxxx001 – TIM1_CLK == PER1_C2/2; xxxxx010 – TIM1_CLK == PER1_C2/4; … xxxxx111 – TIM1_CLK == PER1_C2/128
* The TIMx_CLK timer clock frequency must be less than or equal to the CPU_CLK core clock frequency.		

16.6.11.**UART_CLOCK**

Table 272 – UART_CLOCK Register

Number	30	29..27	26	25	24	23…16	15…8	7…0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	000	0	0	0	00000000	00000000	00000000
	UART4_CLK_EN	UART4_BRG[2:0]	UART3_CLK_EN	UART2_CLK_EN	UART1_CLK_EN	UART3_BRG[7:0]	UART2_BRG[7:0]	UART1_BRG[7:0]

Table 273 – Description of UART_CLOCK register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31	-	Reserved
30	UART4_CLK_EN	UART4 clock frequency enable: 0 – no frequency; 1 – frequency available
29…27	UART4_BRG[2:0]	UART4 clock divider: 000 – UART4_CLK == PER1_C2; 001 – UART4_CLK == PER1_C2/2; 010 – UART4_CLK == PER1_C2/4; … 111 – UART4_CLK == PER1_C2/128

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
26	UART3_CLK_EN	UART3 clock frequency enable: 0 – no clock; 1 – clock enabled
25	UART2_CLK_EN	UART2 clock frequency enable: 0 – no clock; 1 – clock enabled
24	UART1_CLK_EN	UART1 clock frequency enable: 0 – no clock; 1 – clock enabled
23…16	UART3_BRG[7:0]	UART3 clock divider: xxxxx000 – UART3_CLK == PER1_C2; xxxxx001 – UART3_CLK == PER1_C2/2; xxxxx010 – UART3_CLK == PER1_C2/4; … xxxxx111 – UART3_CLK == PER1_C2/128
15…8	UART2_BRG[7:0]	UART2 clock divider: xxxxx000 – UART2_CLK == PER1_C2; xxxxx001 – UART2_CLK == PER1_C2/2; xxxxx010 – UART2_CLK == PER1_C2/4; … xxxxx111 – UART2_CLK == PER1_C2/128
7…0	UART1_BRG[7:0]	UART clock divider: xxxxx000 – UART1_CLK == PER1_C2; xxxxx001 – UART1_CLK == PER1_C2/2; xxxxx010 – UART1_CLK == PER1_C2/4; … xxxxx111 – UART1_CLK == PER1_C2/128

16.6.12. **SSP_CLOCK**

Table 274 – SSP_CLOCK Register

Number	26	25	24	23…16	15…8	7…0
Access	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	00000000	00000000	00000000
	SSP3_CLK_EN	SSP2_CLK_EN	SSP1_CLK_EN	SSP3_BRG [7:0]	SSP2_BRG [7:0]	SSP1_BRG [7:0]

Table 275 – Description of the SSP_CLOCK register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…27	-	Reserved
26	SSP3_CLK_EN	SSP3 clock frequency enable: 0 – no frequency; 1 – frequency available
25	SSP2_CLK_EN	SSP2 clock frequency enable: 0 – no frequency; 1 – frequency available
24	SSP1_CLK_EN	SSP1 clock frequency enable: 0 – no frequency; 1 – frequency available
23…16	SSP3_BRG[7:0]	SSP3 Clock Divider: xxxxx000 – SSP3_CLK == PER1_C2; xxxxx001 – SSP3_CLK == PER1_C2/2; xxxxx010 – SSP3_CLK == PER1_C2/4; … xxxxx111 – SSP3_CLK == PER1_C2/128
15…8	SSP2_BRG[7:0]	SSP2 clock divider: xxxxx000 – SSP2_CLK == PER1_C2; xxxxx001 – SSP2_CLK == PER1_C2/2; xxxxx010 – SSP2_CLK == PER1_C2/4; … xxxxx111 – SSP2_CLK == PER1_C2/128
7…0	SSP1_BRG[7:0]	SSP1 clock divider: xxxxx000 – SSP1_CLK == PER1_C2; xxxxx001 – SSP1_CLK == PER1_C2/2; xxxxx010 – SSP1_CLK == PER1_C2/4; … xxxxx111 – SSP1_CLK == PER1_C2/128

16.6.13. **DIV_SYS_TIM**

Table 276 – DIV_SYS_TIM Register

Number	7..0
Access	R/W
Reset	0
	DIV_SYS_TIM

Table 277 – Description of the DIV_SYS_TIM register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31..8	-	Reserved
7...0	DIV_SYS_TIM	System core timer clock divider: CLK_TIMER_PULSE = HCLK/(DIV_SYS_TIM + 2)

17 Battery domain and real-time clock

The battery domain block is intended to provide calendar and real-time clock functions, to retain a certain set of user data when the main power supply disconnects. The WAKEUP input monitoring function is also implemented in the battery domain. This allows the state of the main power supply to be determined even when it is absent. When the supply voltage U_{CC} drops, the battery domain voltage regulator automatically switches the battery domain supply from U_{CC} to U_{CCB}. If a separate power source (battery) is present on U_{CCB}, the battery domain remains powered and continues to perform its functions.

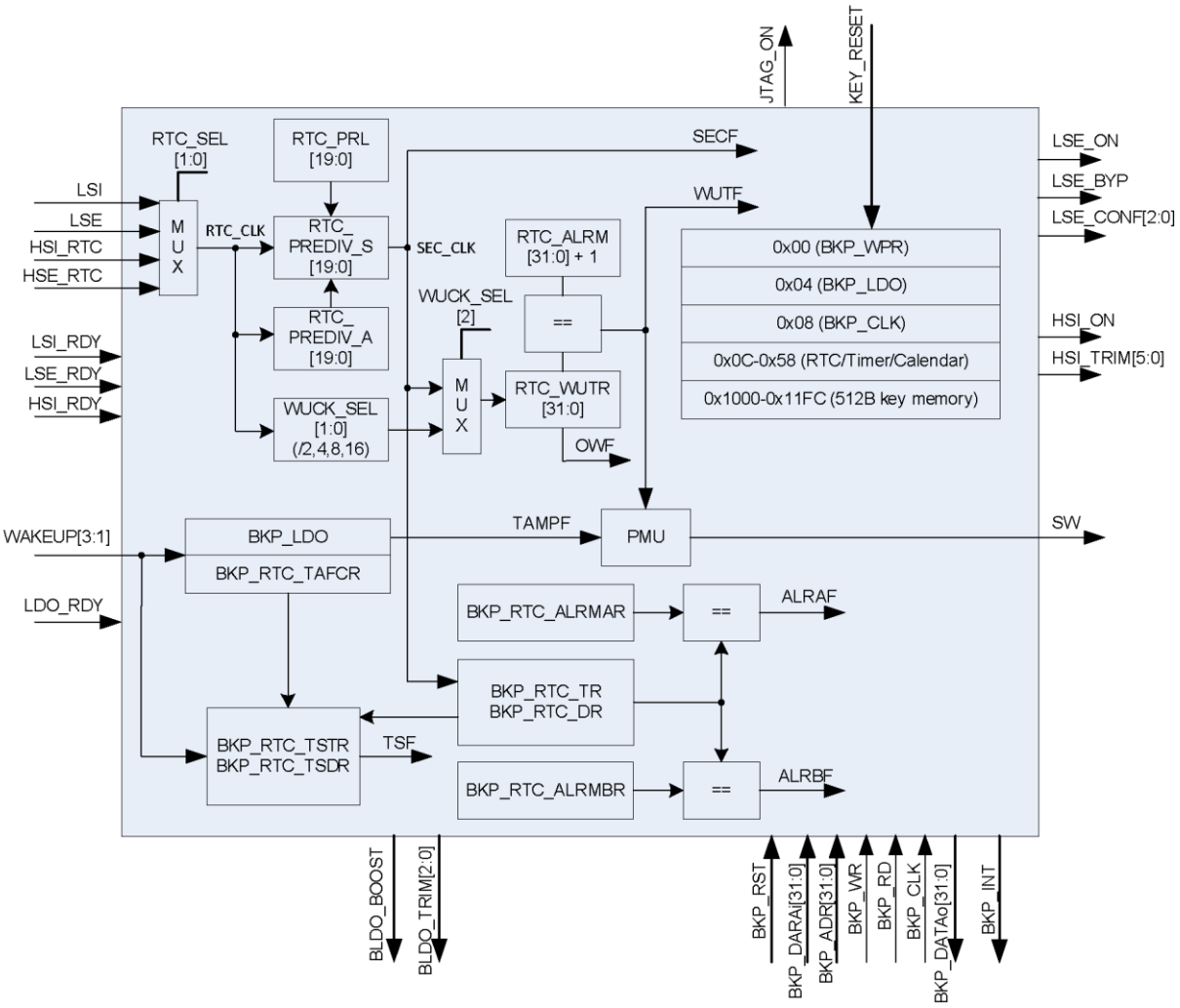


Figure 34 – Block diagram of the battery domain and the real-time clock

17.1. Real time clock

The real-time clock provides a timekeeping mechanism on the chip, including operation when the main power supply is disconnected. The real-time clock is enabled by the RTC_EN bit. The RTC clock source may be the LSI or LSE oscillator, as well as the HSI oscillator (HSI_RTC frequency) or the HSE oscillator (HSE_RTC frequency) with an additional divider of up to 256. The HSI_RTC and HSE_RTC frequencies are generated in the RST_CLK clock control block and can be selected only when there is the digital core supply, generated by the internal voltage regulator. The LSI frequency can be selected when the U_{CC} supply is present, and the LSE frequency when U_{CCB} is present. Selection between the clock sources is performed by the RTC_SEL bits. If the main power supply U_{CC} may be turned off,

the LSE oscillator must be used as the RTC_CLK source, since it is also powered from U_{CCB}. The LSE oscillator control bits are located in the battery domain and therefore are not reset when the main power supply is disconnected.

To count seconds, the real-time clock uses a 20-bit prescaler to divide the RTC_CLK input frequency, producing the SEC_CLK signal at its output. The BKP_RTC_PREDIV_S register serves as the prescaler counter and is clocked by RTC_CLK. The BKP_RTC_PRL register defines the prescaler division factor, and the BKP_RTC_PREDIV_S counter increments from 0 to BKP_RTC_PRL. The division factor in the BKP_RTC_PRL register must be set so that the SEC_CLK frequency equals 1 Hz. After the BKP_RTC_PREDIV_S counter reaches the value BKP_RTC_PRL, it is reset to 0; in this case, the SECF flag in the BKP_RTC_CS register is set, and the BKP_RTC_TR, BKP_RTC_DR, and BKP_RTC_WUTR registers (when the BKP_RTC_WUT timer is clocked from SEC_CLK) are updated 1 RTC_CLK cycle later.

The SEC_CLK frequency, when the real-time clock is enabled, is generated at the output of the BKP block as the RTCCLK_1Hz clock signal; to output the RTCCLK_1Hz signal, the BKP_RTC_PRL register, which defines the division factor, must be set to a nonzero value. The high-level pulse duration of the RTCCLK_1Hz signal is equal to 1 period of the RTC_CLK frequency.

The RTC_CLK frequency is calibrated using the RTC_CAL[7:0] bits (see the description of error 0013 in IRL1206FI Errata Notice document). The value of RTC_CAL[7:0] specifies how many RTC_CLK cycles out of 2^{20} are masked using the additional counter BKP_RTC_PREDIV_A. This calibration method operates by extending one BKP_RTC_PREDIV_S period, which occurs at the beginning of the 2^{20} interval, by RTC_CAL[7:0] RTC_CLK cycles, or by RTC_CAL[7:0]/RTC_CLK seconds. Thus, the RTC_CAL[7:0] bits are used to slow down the clock, with one unit of slowdown equal to $1/2^{20}$, or ~0.95 ppm. To speed up the clock, the BKP_RTC_PRL register must be programmed with a value smaller than required, after which slowing is applied using the RTC_CAL[7:0] bits. The value of the RTC_CAL[7:0] bits can be changed while the real-time clock is operating.

The real-time clock module is based on a binary-coded decimal timer/counter. The counting result is stored in two 32-bit registers. The first register, BKP_RTC_TR, contains time information, while the second register, BKP_RTC_DR, holds the calendar data, including the year, month, day of the week, and date. All data is represented in BCD format, allowing them to be used directly for display on various indicators. The calendar module supports automatic detection of leap years as well as the number of days in the current month. Writing to the BKP_RTC_TR and BKP_RTC_DR registers is allowed only when the real-time clock is disabled (RTC_EN = 0).

Among the functions of the RTC module are two dedicated event registers (BKP_RTC_ALRMAR and BKP_RTC_ALRMBR) that can be used to implement alarms. The format of the event registers is similar to that of the counting registers, which simplifies their programming.

The BKP_RTC_WUT watchdog timer can operate either at the SEC_CLK frequency used by the counter/calendar, or at an independent frequency selected using the WUCK_SEL[2:0] bits. The BKP_RTC_ALRM register is used to set the watchdog timer interval, upon triggering of which the WUTF interrupt flag is set.

The process of writing to the BKP_RTC_WUTR, BKP_RTC_PREDIV_S, BKP_RTC_DR, and BKP_RTC_TR registers is indicated by the WEC flag in the BKP_RTC_CS register. During the write operation, the WEC flag is set to “1”; upon completion of the write, the WEC flag is cleared to “0”. Writing to any of these registers can be performed only when the WEC flag is equal to “0”. While the WEC flag is set to “1”, a new write to these registers will not be performed.

Writing to the BKP_RTC_PRL and BKP_RTC_ALRM registers, as well as to the RTC_CAL[7:0] field of the BKP_RTC_CR register, takes 2 RTC_CLK cycles. If, after writing these registers, a write is performed to the BKP_RTC_WUTR, BKP_RTC_PREDIV_S, BKP_RTC_DR, or BKP_RTC_TR register, completion of all register writes can be monitored using the WEC flag in the BKP_RTC_CS register.

Another useful feature is the TIMESTAMP function, which is intended to determine the exact time at which an external event detected on the WAKEUP1–WAKEUP3 inputs occurs. The active edge of the signal on the WAKEUP1–WAKEUP3 inputs, by which the external event is captured, is configured in the BKP_RTC_TAFCR register. When the specified edge is detected on the WAKEUP1–WAKEUP3 inputs, the values of the counting registers are stored in the corresponding BKP_RTC_TSTR and BKP_RTC_TSDR registers.

The battery domain also provides monitoring of the WAKEUP1–WAKEUP3 input signals using the TAMPER function. In the BKP_RTC_TAFCR register, the monitored active level or edge of the event is programmed, and if the signal at the input matches the programmed value, the event is recorded in the TAMP_IN_SEL[2:0] and TAMPF control bits.

To provide emergency battery power to the integrated circuit, the battery domain includes an SW pin that allows control of an external switch, which connects the battery supply to the main power supply of the integrated circuit. The SW pin operates as an open-drain output. The SW_CNTRL bit in the BKP_LDO register enables control of the SW pin by the internal SW_PWR signal. If SW_CNTRL = 0, the SW pin is in a high-impedance state (the external switch is open). If SW_CNTRL = 1, the output state of the SW pin is determined by the SW_PWR signal:

– SW_PWR = 0 (external switch open): the SW pin is in a high-impedance state;

– SW_PWR = 1 (external switch closed): a low logic level is driven on the SW pin.

The state of the SW pin can be inverted using the SWCTR_INV bit of the RTC_TMPCAL3 register.

The SW_PWR signal is controlled by the TAMPER function, which monitors the WAKEUP1–WAKEUP3 inputs, and by the BKP_RTC_WUT watchdog timer. Setting the SW_PWR signal to “1” (closing the external switch) occurs upon the following events:

– Detection of a TAMPER event on any of the WAKEUP1–WAKEUP3 inputs when there is no main power supply to the integrated circuit;

– Triggering of the BKP_RTC_WUT watchdog timer (WUTF = 1) when WAKE_SLEEP = 1 in the BKP_RTC_TAFCR register and there is no main power supply to the integrated circuit.

The SW_PWR signal to “0” (opening the external switch) is cleared upon the following events:

– Writing “1” to the SW_OFF bit of the BKP_LDO register;

– Triggering of the BKP_RTC_WUT watchdog timer (WUTF = 1) when WAKE_SLEEP = 0 in the BKP_RTC_TAFCR register.

17.1.1. **Temperature compensation of the real-time clock SEC_CLK frequency**

For the real-time clock SEC_CLK frequency, a temperature compensation block for correcting frequency deviation from the nominal value is implemented in the battery domain of the integrated circuit. Use of this block is appropriate when the LSE oscillator with an external 32 kHz crystal resonator is used as the RTC_CLK source.

To enable the temperature compensation block, the TEMP_EN bit must be set. After enabling, an automatic polling process of the auxiliary temperature sensor is started, and the results are written to the TEMP_RESULT bits of the BKP_RTC_TMPCAL3 register when TEMP_RESULT_RDY = 1.

When temperature compensation is enabled, the BKP_RTC_PREDIV_S counter, on every 4th cycle (period of the SEC_CLK frequency), increments not from 0 to BKP_RTC_PRL, but from 0 to (BKP_RTC_PRL − CONSTx), where x corresponds to the code value obtained from the temperature sensor (the value of the TEMP_RESULT field when TEMP_RESULT_RDY = 1).

When the temperature compensation block is enabled, during the 1st cycle the TEMP_RESULT value is 0, and a compensation value of 0 is applied. Subsequently, on every 4th cycle, when the TEMP_RESULT bits are updated, the corresponding CONSTx constant is selected for compensation. Table 252 shows the relationship between the temperature sensor conversion result and temperature. The temperature range is shown for reference only, as the parameters of the temperature sensor are not specified.

Table 278 – Relationship between the temperature sensor conversion result and temperature

Temperature range, °C	TEMP_RESULT field value	Calibration constant used	Calibration constant reset value	Note
≤ –53	0	CONST0	26	Operation of the integrated circuits at below −50 °C is not guaranteed.
(–53; –48]	1	CONST1	22	Operation of the integrated circuits at below −50 °C is not guaranteed
(–48; –43]	2	CONST2	20	
(–43; –38]	3	CONST3	16	
(–38; –28]	4	CONST4	12	
(–28; –18]	5	CONST5	8	
(–18; –8]	6	CONST6	4	
(–8; 22]	7	CONST7	0	
(22; 52]	8	CONST8	0	
(52; 62]	9	CONST9	4	
(62; 72]	10	CONST10	8	
(72; 77]	11	CONST11	12	
(77; 82]	12	CONST12	14	
(82; 87]	13	CONST13	16	Operation of the integrated circuits at over 85°C is not guaranteed
(87; 92]	14	CONST14	20	Operation of the integrated circuits at over 85°C is not guaranteed.
> 92	15	CONST15	22	Operation of the integrated circuits at over 85°C is not guaranteed

Thus, the CONSTx constant values represent an acceleration of the SEC_CLK frequency, with the acceleration equal to CONSTx/(4 · (BKP_RTC_PRL + 1) − CONSTx). At BKP_RTC_PRL = 32767, one unit of correction (acceleration) is ~7.63 ppm.

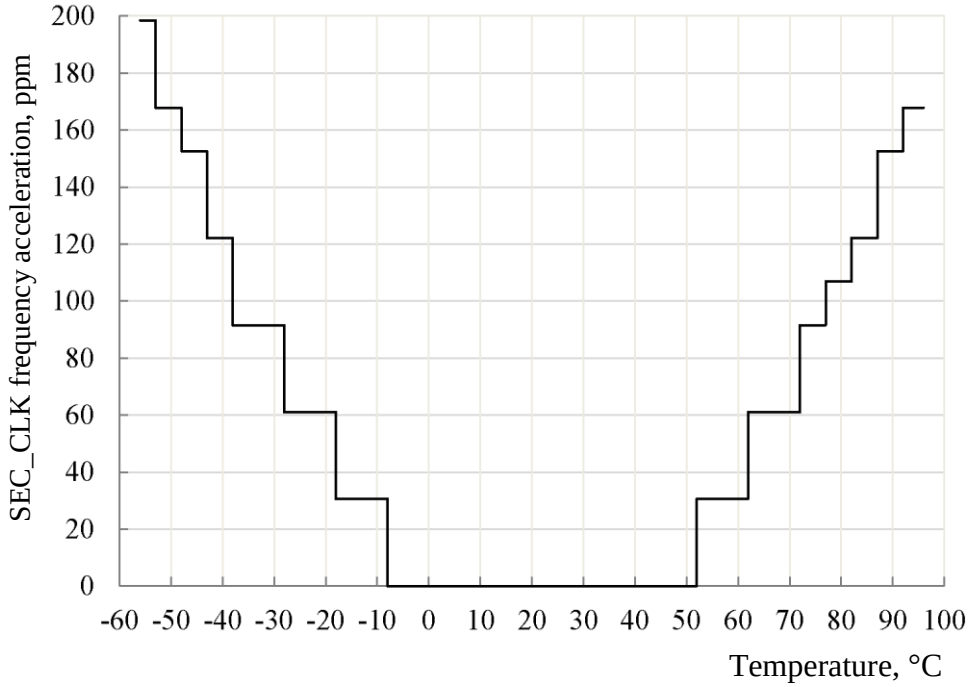


Figure 35 – Typical dependence of SEC_CLK frequency correction (acceleration) on temperature at BKP_RTC_PRL = 32767

For each crystal resonator, the CONSTx values must be selected according to the resonator's characteristics. The reset values are selected for a 32 kHz crystal resonator whose frequency deviation versus temperature characteristic is described by the formula

$$-0.035 \cdot 10^{-6} \cdot (\text{temperature} - 25)^2.$$

The graph of the frequency deviation of such a crystal resonator, in ppm versus temperature, is shown in Figure 36.

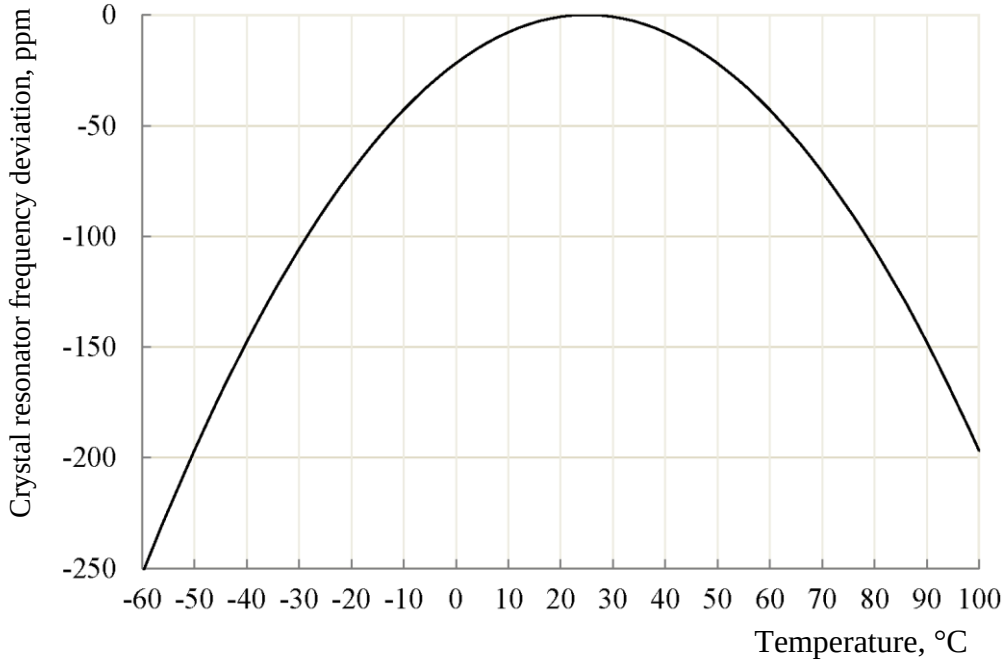


Figure 36 – Typical dependence of the frequency deviation of a 32 kHz crystal resonator on temperature

17.2. **Register memory and cryptographic keys**

The battery domain contains 16 on-chip 32-bit registers for storing control bits of the battery domain and the RTC. A 512-byte cryptographic key RAM is provided for software developers.

17.3. **Description of the battery domain block registers**

Table 279 – Description of the battery domain block registers

Base Address	Name	Description
0x5006_0000	BKP	Battery domain and real-time clock controller
Offset		
0x00	BKP_WPR	Key register for enabling access to the block registers
0x04	BKP_LDO	LDO block control register
0x08	BKP_CLK	HSI, LSI, and LSE block control register
0x0C	BKP_RTC_CR	RTC block control register
0x10	BKP_RTC_WUTR	Watchdog timer counter register
0x14	BKP_RTC_PREDIV_S	Counter/calendar prescaler counter register
0x18	BKP_RTC_PRL	Counter/calendar prescaler division factor register
0x1C	BKP_RTC_ALRM	Register holding the comparison value for the BKP_RTC_WUT watchdog timer counter and generation of the WUTF signal
0x20	BKP_RTC_CS	Real-time clock control and status flag register
0x24	BKP_RTC_TR	Counter/calendar time register in BCD format
0x28	BKP_RTC_DR	Counter/calendar date register in BCD format
0x2C	BKP_RTC_ALRMAR	Event A register – calendar-time alarm
0x30	BKP_RTC_ALRMBR	Event B register – calendar-time alarm
0x34	BKP_RTC_TSTR1	External event time register for input 1
0x38	BKP_RTC_TSTR2	External event time register for input 2
0x3C	BKP_RTC_TSTR3	External event time register for input 3
0x40	BKP_RTC_TSDR1	External event date register for input 1
0x44	BKP_RTC_TSDR2	External event date register for input 2
0x48	BKP_RTC_TSDR3	External event date register for input 3
0x4C	BKP_RTC_TAFCR	Event detection control register
0x50	BKP_RTC_TMPCAL1	Calibration and control register for the LSE frequency temperature compensation block
0x54	BKP_RTC_TMPCAL2	Calibration and control register for the LSE frequency temperature compensation block
0x58	BKP_RTC_TMPCAL3	Calibration and control register for the LSE frequency temperature compensation block
0x1000-0x11FC	BKP_MEM	512-byte RAM memory

17.3.1. **BKP_MEM (128x32)**

Table 280 – BKP_MEM Memory

Number	31…0
Access	R/W
Reset	X
	BKP_MEM[31:0]

Table 281 – Description of the BKP_MEM register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…0	BKP_MEM[31:0]	RAM memory cell

17.3.2. **BKP_WPR**

Table 282 – BKP_WPR Memory

Number	31…0
Access	R/W
Reset	0
	BKP_WPR[31:0]

Table 283 – Description of the BKP_WPR register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…0	BKP_WPR[31:0]	Writing the value 0x8555AAA1 to the register enables write access to the other registers of the battery domain block.

17.3.3. **BKP_LDO**

Table 284 – BKP_LDO Register

Number	31, 30	29	28, 27	26	25	15…10	9...7	6
Access	U	R/C1	R/W	R/C1	R/W	U	R	R/W
Reset	0	0	00	0	0	0	000	1
	-	SW_OFF	MODE[1:0]	TAMPF	FPOR	-	WAKEUP 3-1	JTAG_ON

Number	5	4, 3	2	1, 0
Access	R/W	R/W	R/W	R/W
Reset	0	00	0	00
	SW_CNTRL	LDO_BOOST[1:0]	BLDO_BOOST	BLDO_TRIM[1:0]

Table 285 – Description of the BKP_LDO register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...30	-	Reserved
29	SW_OFF	Disabling emergency battery power supply to the integrated circuit. To disable, write 1. When read, this field reflects the state of the SW_PWR signal.
28, 27	MODE[1:0]	When using the bootloader described in section 9: the integrated circuit startup mode obtained from the MODE[1:0] inputs (PC[0], PC[1]). When using a bootloader other than the one described in section 9, this field may be used at the user's discretion.
26	TAMPF	Indication of detection by the TAMPER circuitry of unauthorized device opening via the WAKEUP3–WAKEUP1 inputs (logical OR of the TAMP_IN_SEL[2:0] bits): 0 – no tamper event detected; 1 – a tamper event was detected. To clear the tamper indication, write TAMPF = 1. Clearing the TAMPF flag also clears the TAMP_IN_SEL[2:0] flags. Verify clearing of the tamper indication by reading TAMPF = 0 or by reading TAMP_IN_SEL[2:0] = 0.
25	FPOR	When using the bootloader described in section 9: POR trigger flag. Upon a power-on reset, this flag is set to 0. It may be used by the bootloader located in the BOOT memory area, described in section 9, as a software flag to determine the reset type. Example algorithm for evaluating the FPOR bit: – FPOR = 0: program execution after power-on reset; set FPOR to 1; – FPOR = 1: program execution after system reset; do not change the FPOR state. When using a bootloader other than the one described in section 9, this field may be used at the user's discretion.
24...10	-	Reserved
9…7	WAKEUP3-1	State of the WAKEUP3–WAKEUP1 inputs.
6	JTAG_ON	JTAG port enable: 0 – disabled; 1 – enabled
5	SW_CNTRL	SW pin control enable bit for providing emergency battery power to the integrated circuit: 1 – enabled; 0 – disabled
4, 3	LDO_BOOST[1:0]	Main regulator boost mode activation bits: 00 – typical regulator output value; 01 – increased regulator voltage; 10 – reduced regulator voltage; 11 – reduced regulator voltage. Only the values specified in TRIM_LDO are recommended for use.

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
2	BLDO_BOOST	Boost mode activation input of the battery domain regulator: 0 – typical regulator output value; 1 – increased regulator output voltage using BLDO_TRIM[1:0].
1, 0	BLDO_TRIM[1:0]	Battery domain power regulator trimming: 00 – typical value; 01…11 – increased reference voltage.

17.3.4. **BKP_CLK**

Table 286 – BKP_CLK Register

Number	31…21	20…15	14	13	12
Access	U	R/W	RO	R/W	RO
Reset	0	000000	1	1	0
	-	HSI_LSI_TRIM[5:0]	HSI_RDY	HSI_ON	LSI_RDY

Number	11…6	5	4, 3	2	1	0
Access	U	R/W	R/W	RO	R/W	R/W
Reset	0	0	00	0	0	0
	-	LSI_ON	LSE_CONF[1:0]	LSE_RDY	LSE_BYP	LSE_ON

Table 287 – Description of the BKP_CLK register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31..21	-	Reserved
20...15	HSI_LSI_TRIM[5:0]	HSI and LSI oscillator trimming coefficient. It is recommended to use the TRIM_HSI bit values programmed into Flash memory during manufacturing.
14	HSI_RDY	HSI oscillator ready flag: 0 – the oscillator is not running or has not stabilized; 1 – the oscillator is operating in normal mode
13	HSI_ON	HSI oscillator control bit: 1 – oscillator enabled; 0 – oscillator disabled. This bit is set to 1 by hardware only upon a reset caused by application of battery power.
12	LSI_RDY	LSI oscillator ready flag: 0 – the oscillator is not running or has not stabilized; 1 – the oscillator is operating in normal mode
11…6	-	Reserved
5	LSI_ON	LSI oscillator control bit: 0 – oscillator disabled; 1 – oscillator enabled

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
4, 3	LSE_CONF[1:0]	LSE oscillator configuration bits: 00 – default mode. 01 – glitch filter enabled. 10 – amplifier current increased. 11 – amplifier current increased and glitch filter enabled
2	LSE_RDY	LSE oscillator ready flag: 0 – the oscillator is not running or has not stabilized; 1 – the oscillator is operating in normal mode
1	LSE_BYP	LSE oscillator mode control bit: 0 – oscillator mode; 1 – bypass (external oscillator) mode
0	LSE_ON	LSE oscillator enable bit: 0 – oscillator disabled; 1 – oscillator enabled

17.3.5. **BKP_RTC_CR**

Table 288 – BKP_RTC_CR Register

Number	31…26	25	24…15	14…12	11
Access	U	R/W	U	R/W	R/W
Reset	0	0	0	000	0
	-	FMT	-	WUCK_SEL[2:0]	RTC_RESET

Number	10…3	2	1, 0
Access	R/W	R/W	R/W
Reset	0000000	0	00
	RTC_CAL[7:0]	RTC_EN	RTC_SEL[1:0]

Table 289 – Description of the BKP_RTC_CR register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…26	-	Reserved
25	FMT	Time format: 0– 24 hours a day; 1 – AM/PM time format
24…15	-	Reserved
14	WUCK_SEL[2]	Independent clock selection bits for the watchdog timer: WUCK_SEL[2] = 1 – common clocking with the counter/calendar; WUCK_SEL[2] = 0 – independent clocking from the selected divider

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
13...12	WUCK_SEL[1:0]	Clock divider selection bits for the watchdog timer: WUCK_SEL[1:0] = 3 – divide by 2; WUCK_SEL[1:0] = 2 – divide by 4; WUCK_SEL[1:0] = 1 – divide by 8; WUCK_SEL[1:0] = 0 – divide by 16.
11	RTC_RESET	Real-time clock reset: 0 – RTC is not reset; 1 – RTC is reset. Resets the RTC, WUT, TAMPER, and TIMESTAMP blocks, as well as the BKP_RTC_WUTR, BKP_RTC_PREDIV_S, BKP_RTC_DR, and BKP_RTC_TR registers.
10...3	RTC_CAL[7:0]	RTC_CLK frequency trimming coefficient. Out of every 2^{20} RTC_CLK cycles, RTC_CAL cycles are masked: 00000000 – 0 cycles; 00000001 – 1 cycle; …. 11111111 – 255 cycles. Thus, if the initial RTC_CLK frequency is 32768.00000 Hz, the average clock frequency after trimming is: RTC_CLK_C = 32768.00000 Hz at RTC_CAL = 0; RTC_CLK_C = 32767.96875 Hz at RTC_CAL = 1; … RTC_CLK_C = 32760.03125 Hz at RTC_CAL = 255. In the general case, the average RTC_CLK frequency after trimming is RTC_CLK_C = RTC_CLK • (1 – RTC_CAL[7:0]/2^{20}). One trimming unit (slowdown) equals $1/2^{20}$, or ~0.95 ppm. The trimmed RTC_CLK_C frequency is used only to generate the SEC_CLK frequency.
2	RTC_EN	Real-time clock enable bit: 0 – operation disabled; 1 – operation enabled. Enables operation of the RTC, WUT, and TIMESTAMP blocks.

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
1, 0	RTC_SEL[1:0]	Real-time clock source selection bits: 00 – LSI; 01– LSE; 10– HSI_RTC; 11– HSE_RTC To switch the RTC_CLK frequency from one source to another, both clock sources must be enabled and in operating mode. An exception is the first switch from LSI to another source after a power-on reset; in this case, enabling the LSI source is not required. A valid RTC_CLK frequency is generated after 3 periods of the source frequency being switched from, plus 3 periods of the source frequency being switched to. Use of the RTC_CLK frequency is allowed only after a valid frequency has been generated. Until a valid RTC_CLK frequency is formed, RTC operation must be disabled or the RTC must be held in reset.

17.3.6. **BKP_RTC_WUTR**

Table 290 – BKP_RTC_WUTR Register

Number	31
Access	R/W
Reset	0
	RTC_WUTR[31:0]

Table 291 – Description of the BKP_RTC_WUTR register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...0	RTC_WUTR[31:0]	Watchdog timer counter value

17.3.7. **BKP_RTC_PREDIV_S**

Table 292 – BKP_RTC_PREDIV_S Register

Number	31...20	19…0
Access	U	R/W
Reset	0	0
	-	RTC_DIV[19:0]

Table 293 – Description of the BKP_RTC_PREDIV_S register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…20	-	-
19…0	RTC_DIV[19:0]	Counter/calendar prescaler counter value

17.3.8. **BKP_RTC_PRL**

Table 294 – BKP_RTC_PRL Register

Number	31…20	19…0
Access	U	R/W
Reset	0	0x7FFB
	-	RTC_PRL[19:0]

Table 295 – Description of the BKP_RTC_PRL register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…20	-	-
19…0	RTC_PRL[19:0]	Division factor for generating the SEC_CLK frequency: SEC_CLK = RTC_CLK/(RTC_PRL[19:0] + 1), where RTC_CLK is the clock frequency taking into account trimming by RTC_CAL[7:0].

17.3.9. **BKP_RTC_ALRM**

Table 296 – BKP_RTC_ALRM Register

Number	31…0
Access	R/W
Reset	0
	RTC_ALRM[31:0]

Table 297 – Description of the BKP_RTC_ALRM register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...0	RTC_ALRM[31:0]	Comparison value for the BKP_RTC_WUT watchdog timer counter and generation of the WUTF signal. The WUTF signal is generated when the BKP_RTC_WUTR counter exceeds the value in the BKP_RTC_ALRM register: RTC_WUTR[31:0] == RTC_ALRM[31:0] + 1

17.3.10. **BKP_RTC_CS**

Table 298 – BKP_RTC_CS Register

Number	31…14	13	12	11	10	9	8
Access	U	R/C1	R/W	R/W	R/C1	R/C1	R/W
Reset	0	0	0	0	0	0	0
	-	TSF	ALRBF_IE	ALRAF_IE	ALRBF	ALRAF	ALRB_EN

Number	7	6	5	4	3	2	1	0
Access	R/W	RO	R/W	R/W	R/W	R/C1	R/C1	R/C1
Reset	0	0	0	0	0	0	0	0
	ALRA_EN	WEC	WUTF_IE	SECF_IE	OWF_IE	WUTF	SECF	OWF

Table 299 – Description of the BKP_RTC_CS register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…14	-	Reserved
13	TSF	External event flag detected by the TIMESTAMP logic on the WAKEUP1–WAKEUP3 inputs (logical OR of the TS_IN_SEL[2:0] bits). The flag is cleared by writing 1. Clearing the TSF flag also clears the TS_IN_SEL[2:0] flags.
12	ALRBF_IE	Interrupt enable for the ALRBF event: 0 – disabled; 1 – allowed
11	ALRAF_IE	Interrupt enable for the ALRAF event: 0 – disabled; 1 – allowed
10	ALRBF	Event B time set flag. The flag is cleared by writing 1.
9	ALRAF	Event A time set flag. The flag is cleared by writing 1.
8	ALRB_EN	Event B enable. Match between BKP_RTC_ALRMBR and the current calendar time: 0 – disabled; 1 – allowed
7	ALRA_EN	Event A enable. Match between BKP_RTC_ALRMAR and the current calendar time: 0 – disabled; 1 – allowed

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
6	WEC	RTC register write-in-progress flag: 0 – writing to any of the BKP_RTC_WUTR, BKP_RTC_PREDIV_S, BKP_RTC_DR, or BKP_RTC_TR registers is allowed; 1 – a write operation to one of the BKP_RTC_WUTR, BKP_RTC_PREDIV_S, BKP_RTC_DR, or BKP_RTC_TR registers is in progress, and writing to these registers is not allowed. After writing each of the BKP_RTC_WUTR, BKP_RTC_PREDIV_S, BKP_RTC_DR, and BKP_RTC_TR registers, wait for the write operation to complete.
5	WUTF_IE	Interrupt enable for the WUTF event: 0 – disabled; 1 – allowed
4	SECF_IE	Interrupt enable for the SECF event: 0 – disabled; 1 – allowed
3	OWF_IE	Interrupt enable for the OWF event: 0 – overrun interrupt disabled; 1 – overrun interrupt enabled.
2	WUTF	Flag indicating that the BKP_RTC_WUTR watchdog timer counter has exceeded the BKP_RTC_ALRM value: 0 – no exceed event occurred; 1 – an exceed event occurred. The flag is set when RTC_WUTR[31:0] == RTC_ALRM[31:0] + 1. The flag is cleared by writing 1.
1	SECF	Flag indicating a match between the BKP_RTC_PREDIV_S and BKP_RTC_PRL register values: 0 – no match occurred; 1 – a match occurred. The SEC_CLK frequency is generated on match events between BKP_RTC_PREDIV_S and BKP_RTC_PRL. Thus, the SECF flag is set on every SEC_CLK cycle. The flag is cleared by writing 1.
0	OWF	Watchdog timer overrun flag for BKP_RTC_WUT: 0 – no overrun; 1 – an overrun occurred. The flag is cleared by writing 1.

17.3.11. **BKP_RTC_TR**

Table 300 – BKP_RTC_TR Register

Number	31…23	22	21, 20	19..16	15
Access	U	R/W	R/W	R/W	U
Reset	0	0	00	0000	0
	-	PM	HT[1:0]	HU[3:0]	-

Number	14…12	11…8	7	6...4	3...0
Access	R/W	R/W	U	R/W	R/W
Reset	000	0000	0	000	0000
	MNT[2:0]	MNU[3:0]	-	ST[2:0]	SU[3:0]

Table 301 – Description of the BKP_RTC_TR register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…23	-	Reserved
22	PM	AM/PM value: 0 – AM or 24-hour format; 1 – PM
21, 20	HT[1:0]	Tens of hours in BCD format
19..16	HU[3:0]	Hours units in BCD format
15	-	Reserved
14..12	MNT[2:0]	Tens of minutes in BCD format
11..8	MNU[3:0]	Minutes in BCD format
7	-	Reserved
6..4	ST[2:0]	Tens of seconds in BCD format
3..0	SU[3:0]	Seconds units in BCD format

17.3.12. **BKP_RTC_DR**

Table 302 – BKP_RTC_DR Register

Number	31…24	23…20	19…16	15…13
Access	U	R/W	R/W	R/W
Reset	0	0001	1001	010
	-	YT[3:0]	YU[3:0]	WDU[2:0]

Number	12	11..8	7, 6	5, 4	3...0
Access	R/W	R/W	U	R/W	R/W
Reset	0	0001	0	00	0001
	MT	MU[3:0]	-	DT[1:0]	DU[3:0]

Table 303 – Description of the BKP_RTC_DR register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…24	-	Reserved
23...20	YT[3:0]	Tens of years in BCD format
19...16	YU[3:0]	Units of years in BCD format
15...13	WDU[2:0]	Day of the week: 000 – not used; 001 – Monday; … 111 – Sunday
12	MT	Tens of months in BCD format
11…8	MU[3:0]	Units of months in BCD format
7, 6	-	Reserved
5, 4	DT[1:0]	Tens of dates in BCD format
3...0	DU[3:0]	Units of dates in BCD format

17.3.13. **BKP_RTC_ALRMAR, BKP_RTC_ALRMBR**

Table 304 – BKP_RTC_ALRMAR, BKP_RTC_ALRMBR registers

Number	31	30	29, 28	27…24	23	22	21, 20
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	00	0000	0	0	00
	MSK4	WDSEL	DT[1:0]	DU[3:0]	MSK3	PM	HT[1:0]

Number	19…16	15	14…12	11…8	7	6…4	3…0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0000	0	000	0000	0	000	0000
	HU[3:0]	MSK2	MNT[2:0]	MNU[3:0]	MSK1	ST[2:0]	SU[3:0]

Table 305 – Description of the bits of the BKP_RTC_ALRMAR, BKP_RTC_ALRMBR registers

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31	MSK4	Date mask: 0 – date is included; 1 – date is ignored.
30	WDSEL	Day-of-week selection: 0 – the date fields DU[3:0] and DT[1:0] are used; 1 – DU[3:0] defines the day of the week, and DT[1:0] is not used.
29, 28	DT[1:0]	Tens of dates in BCD format
27...24	DU[3:0]	Units of dates in BCD format
23	MSK3	Hour mask: 0 – hours are included; 1 – hours are ignored.
22	PM	AM/PM value: 0 – AM or 24-hour format; 1– PM
21, 20	HT[1:0]	Tens of hours in BCD format
19…16	HU[3:0]	Hours units in BCD format
15	MSK2	Minute mask: 0 – minutes are included; 1 – minutes are ignored.
14…12	MNT[2:0]	Tens of minutes in BCD format
11…8	MNU[3:0]	Minutes in BCD format
7	MSK1	Second mask: 0 – seconds are included; 1 – seconds are ignored.
6…4	ST[2:0]	Tens of seconds in BCD format
3…0	SU[3:0]	Seconds units in BCD format

17.3.14. BKP_RTC_TSTR1-BKP_RTC_TSTR3, BKP_RTC_TSDR1-BKP_RTC_TSDR3

A pair of read-only registers in which the date of external events is stored. The format is fully identical to the corresponding counting registers BKP_RTC_TR and BKP_RTC_DR.

17.3.15. BKP_RTC_TAFCR

Table 306 – BKP_RTC_TAFCR Register

Number	31…29	28...26	25…23
Access	R/W	R/W	R/W
Reset	000	000	000
	TAMP_LEVEL_W[2:0]	TAMP_EDGE_W[2:0]	TAMP_EN_W[2:0]

Number	22	21...19	18...16	15…10
Access	R/W	R/C1	R/C1	U
Reset	0	000	000	0
	WAKE_SLEEP	TS_IN_SEL[2:0]	TAMP_IN_SEL[2:0]	-

Number	9	8...6	5	4	3...1	0
Access	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0
	TS_IE	TS_EDGE[2:0]	TS_EN	TAMP_IE	TAMP_TRG[2:0]	TAMP_EN

Table 307 – Description of the BKP_RTC_TAFCR register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...29	TAMP_LEVEL_W[2:0]	Disabling level capture on the WAKEUP1–WAKEUP3 inputs: 0 – level capture for pulses longer than two real-time clock synchronization cycles is enabled; 1 – level capture is disabled. The pulse edge is captured according to the TAMP_EDGE_W[2:0] bits. Each bit of TAMP_LEVEL_W[2:0] configures level capture for the corresponding WAKEUP3–WAKEUP1 input.
28…26	TAMP_EDGE_W[2:0]	Disabling edge capture on the WAKEUP1–WAKEUP3 inputs: 0 – edge capture is enabled; 1 – edge capture is disabled; pulses shorter than two real-time clock synchronization cycles are not captured. Pulses longer than two cycles are captured according to the TAMP_LEVEL_W bits. Each bit of TAMP_EDGE_W[2:0] configures edge capture for the corresponding WAKEUP3–WAKEUP1 input.

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
25...23	TAMP_EN_W[2:0]	Individual enable of TAMPER event detection on the WAKEUP1–WAKEUP3 inputs when TAMP_EN = 0: 0 – disabled; 1 – enabled. Each bit of TAMP_EN_W[2:0] enables TAMPER event detection on the corresponding WAKEUP3–WAKEUP1 input.
22	WAKE_SLEEP	SW switch control selection on a BKP_RTC_WUT watchdog timer event (WUTF = 1): 0 – the SW switch is opened (SW_PWR = 0) when the BKP_RTC_WUT watchdog timer triggers (WUTF = 1); 1 – the SW switch is closed (SW_PWR = 1) when the BKP_RTC_WUT watchdog timer triggers (WUTF = 1).
21..19	TS_IN_SEL[2:0]	Inputs detected by the TIMESTAMP logic: 0 – the input was not triggered; 1 – the input was triggered. Each bit of TS_IN_SEL[2:0] corresponds to triggering of the respective WAKEUP3–WAKEUP1 input. Multiple bits may be set simultaneously if multiple inputs are triggered. Cleared by writing 1 to the corresponding bit.
18..16	TAMP_IN_SEL[2:0]	Inputs detected by the TAMPER logic: 0 – the input was not triggered; 1 – the input was triggered. Each bit of TAMP_IN_SEL[2:0] corresponds to triggering of the respective WAKEUP3–WAKEUP1 input. Multiple bits may be set simultaneously if multiple inputs are triggered. Cleared by writing 1 to the corresponding bit.
15…10	-	Reserved
9	TS_IE	Interrupt enable for an external TIMESTAMP event (TSF flag): 0 – disabled; 1 – allowed
8...6	TS_EDGE[2:0]	Trigger edge selection for an external TIMESTAMP event: 0 – rising edge; 1 – falling edge. Each bit of TS_EDGE[2:0] configures the trigger edge for the corresponding WAKEUP3–WAKEUP1 input.
5	TS_EN	Enable of time capture for external TIMESTAMP events detected on the WAKEUP1–WAKEUP3 inputs: 0 – disabled; 1 – allowed

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
4	TAMP_IE	Interrupt enable for a TAMPER event (TAMPF flag): 0 – disabled; 1 – allowed
3...1	TAMP_TRG[2:0]	Active level of the TAMPER event: 0 – rising edge or high level; 1 – falling edge or low level. Each bit of TAMP_TRG[2:0] configures signal detection for the corresponding WAKEUP3–WAKEUP1 input.
0	TAMP_EN	Enable of TAMPER event detection on the WAKEUP1–WAKEUP3 inputs: 0 – disabled; 1 – allowed After changing the bit state, the TAMPER enable/disable delay is two real-time clock synchronization cycles.

17.3.16. **BKP_RTC_TMPCAL1**

Table 308 – BKP_RTC_TMPCAL1 Register

Number	31…30	29…25	24…20
Access	R/W	R/W	R/W
Reset	00	01000	01100
	CONST6[1:0]	CONST5[4:0]	CONST4[4:0]

Number	19…15	14…10	9…5	4...0
Access	R/W	R/W	R/W	R/W
Reset	10000	10100	10110	11010
	CONST3[4:0]	CONST2[4:0]	CONST1[4:0]	CONST0[4:0]

Table 309 – Description of the BKP_RTC_TMPCAL1 register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…30	CONST6[1:0]	Two least significant bits of the calibration constant for temperature sensor code 6
29…25	CONST5[4:0]	Calibration constant for temperature sensor code 5
24…20	CONST4[4:0]	Calibration constant for temperature sensor code 4
19...15	CONST3[4:0]	Calibration constant for temperature sensor code 3
14...10	CONST2[4:0]	Calibration constant for temperature sensor code 2
9...5	CONST1[4:0]	Calibration constant for temperature sensor code 1
4..0	CONST0[4:0]	Calibration constant for temperature sensor code 0

17.3.17. **BKP_RTC_TMPCAL2**

Table 310 – BKP_RTC_TMPCAL2 Register

Number	31…28	27…23	22…18
Access	R/W	R/W	R/W
Reset	1110	01100	01000
	CONST12[3:0]	CONST11[4:0]	CONST10[4:0]

Number	17…13	12…8	7…3	2...0
Access	R/W	R/W	R/W	R/W
Reset	00100	00000	00000	001
	CONST9[4:0]	CONST8[4:0]	CONST7[4:0]	CONST6[4:2]

Table 311 – Description of the BKP_RTC_TMPCAL2 register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…28	CONST12[3:0]	Four least significant bits of the calibration constant for temperature sensor code 12
27…23	CONST11[4:0]	Calibration constant for temperature sensor code 11
22…18	CONST10[4:0]	Calibration constant for temperature sensor code 10
17...13	CONST9[4:0]	Calibration constant for temperature sensor code 9
12...8	CONST8[4:0]	Calibration constant for temperature sensor code 8
7...3	CONST7[4:0]	Calibration constant for temperature sensor code 7
2...0	CONST6[4:2]	Three most significant bits of the calibration constant for temperature sensor code 6

17.3.18. **BKP_RTC_TMPCAL3**

Table 312 – BKP_RTC_TMPCAL3 Register

Number	31…29	28	27…24	23…19	18…14	13…9
Access	U	R	R	R/W	R/W	R/W
Reset	0	0	0000	10110	10100	10000
	-	TEMP_RESULT_RDY	TEMP_RESULT[3:0]	CONST15[4:0]	CONST14[4:0]	CONST13[4:0]

Number	8	7…5	4	3	2	1	0
Access	R/W	U	R/W	R/W	R/W	U	R/W
Reset	0	0	0	0	0	-	0
	CONST12[4]	-	TEMP_EN	LSE_CONF_B2	BLDO_TRIM_B2	-	SWCTR_INV

Table 313 – Description of the BKP_RTC_TMPCAL3 register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…29	-	Reserved
28	TEMP_RESULT_RDY	Validity flag for the data in the TEMP_RESULT[3:0] bits: 1 – data is valid; 0 – data is not valid
27…24	TEMP_RESULT[3:0]	Temperature sensor code value
23…19	CONST15[4:0]	Calibration constant for temperature sensor code 15
18…14	CONST14[4:0]	Calibration constant for temperature sensor code 14
13...9	CONST13[4:0]	Calibration constant for temperature sensor code 13
8	CONST12[4]	The most significant bit of the calibration constant for temperature sensor code 12
7...5	-	Reserved
4	TEMP_EN	Enable of the temperature sensor circuitry: 1 – enabled; 0 – disabled
3	LSE_CONF_B2	LSE_CONF[2]. Always write zero
2	BLDO_TRIM_B2	BLDO_TRIM[2]. Always write zero
1	-	Reserved
0	SWCTR_INV	Inversion of SW output control logic (open-drain): 0 – when SW_PWR = 1, the output switches from high-impedance (HZ) to logic 0; 1 – when SW_PWR = 1, the output switches from logic 0 to high-impedance (HZ).

18 Input/output ports

The integrated circuit has four input/output ports. Port A is 16 bits wide, Port B is 15 bits wide, Port C is 8 bits wide, and Port D is 16 bits wide; their pins are multiplexed among various functional blocks, and each pin is controlled individually. To transfer control of port pins to a specific peripheral block, the required function and configuration must be set for the corresponding pins.

Table 314 – Input/Output Ports

Pin	Analog function MODE=00, ANALOG_EN=0 OE=0		Digital function						
			I/O port MODE=00, ANALOG_EN=0/1 OE=0/1	Primary MODE=01, ANALOG_EN=0/1 OE=X		Alternate MODE=10, ANALOG_EN=0/1 OE=X		Override MODE=11, ANALOG_EN=0/1 OE=X	
Port A									
PA0	-		PA0	TMR1_CH1	1	SSP2_FSS	9	-	
PA1	-		PA1	TMR1_CH1N		SSP2_CLK		-	
PA2	-		PA2	TMR1_CH2		SSP2_RXD		-	
PA3	-		PA3	TMR1_CH2N		SSP2_TXD		-	
PA4	-		PA4	TMR1_CH3		-		-	
PA5	-		PA5	TMR1_CH3N		-		-	
PA6*	-		PA6/TCK	TMR1_CH4		-		-	
PA7*	-		PA7/TDO	TMR1_CH4N		-		-	
PA8*	-		PA8/TMS	TMR1_ETR		-		-	
PA9*	-		PA9/TDI	TMR1_BRK		-		-	
PA10	-		PA10	EXT_INT1		UART3_RXD	10	TMR4_CH1	16
PA11	-		PA11	TMR2_CH4N	2	UART3_TXD		TMR4_CH1N	
PA12	-		PA12	SSP1_FSS	3	-		TMR4_CH2	
PA13	-		PA13	SSP1_CLK		-		TMR4_CH2N	
PA14	-		PA14	SSP1_RXD		-		TMR4_CH3	
PA15	-		PA15	SSP1_TXD		-		TMR4_CH3N	
Port B									
PB0	-		PB0	UART1_TXD	4	UART7816_TXD	11	TMR3_CH1	17
PB1	-		PB1	UART1_RXD		UART7816_RXD		TMR3_CH1N	
PB2	-		PB2	nSIROUT1		UART7816_CLK		TMR3_CH2	
PB3	-		PB3	nSIRIN1		UART7816_CTS		TMR3_CH2N	
PB4	-	1	PB4	nUART1DTR		-		TMR3_CH3	
PB5	-		PB5	nUART1RTS		UART7816_RTS	12	TMR3_CH3N	
PB6	-	2	PB6	nUART1RI		EXT_INT2		TMR3_CH4	
PB7	-		PB7	nUART1DCD		EXT_INT3		TMR3_CH4N	
PB8	-		PB8	nUART1DSR		TMR2_ETR	13	TMR3_ETR	
PB9	-		PB9	nUART1CTS		TMR2_BRK		TMR3_BRK	
PB10	-		PB10	TMR2_CH2	5	UART4_RXD	20	-	
PB11	-		PB11	TMR2_CH2N		UART4_TXD		-	
PB12	-		PB12	TMR2_CH3		-		-	
PB13	-		PB13	TMR2_CH3N		UART2_TXD	6	-	
PB14	-		PB14	TMR2_CH4		UART2_RXD		-	

Pin	Analog function MODE=00, ANALOG_EN=0 OE=0		Digital function						
			I/O port MODE=00, ANALOG_EN=0/1 OE=0/1	Primary MODE=01, ANALOG_EN=0/1 OE=X		Alternate MODE=10, ANALOG_EN=0/1 OE=X		Override MODE=11, ANALOG_EN=0/1 OE=X	
Port C									
PC0	-		PC0/MODE0	TMR4_ETR	19	-		TMR4_CH4	18
PC1	-	3	PC1/MODE1	TMR4_BRK		-		TMR4_CH4N	
PC2	-		PC2	TMR2_CH1	7	SSP3_FSS	14	I2C1_SCL	15
PC3	-		PC3	TMR2_CH1N		SSP3_CLK		I2C1_SDA	
PC4	-		PC4	EXT_INT2		SSP3_RXD		-	
PC5	-		PC5	EXT_INT3		SSP3_TXD		-	
PC6	-		PC6	TMR2_ETR	8	I2C1_SCL	15	-	
PC7	-		PC7	TMR2_BRK		I2C1_SDA		-	
Port D									
PD0	-		PD0	SSP2_FSS	9	SD_V0_EXT			
PD1	-		PD1	SSP2_CLK		SD_V1_EXT			
PD2	-		PD2	SSP2_RXD		SD_V2_EXT		-	
PD3	-		PD3	SSP2_TXD		-		-	
PD4	-		PD4	UART2_TXD	6	-		-	
PD5	-		PD5	UART2_RXD		-		-	
PD6	-		PD6	UART3_RXD	10	-		-	
PD7	-		PD7	UART3_TXD		-		-	
PD8	-		PD8	UART4_RXD	20	-		-	
PD9	-		PD9	UART4_TXD		-		-	
PD10	-		PD10	UART7816_TXD	11	SD_I0_EXT		-	
PD11	-		PD11	UART7816_RXD		SD_I1_EXT		-	
PD12	-		PD12	UART7816_CLK		SD_I2_EXT		-	
PD13	-		PD13	UART7816_CTS		SD_I3_EXT		-	
PD14	-		PD14	UART7816_RTS	12	CLK_SD_EXT		-	
PD15	-		PD15	EXT_INT1		RTCCLK_1Hz		-	

* When the JTAG port is enabled, the Timer 1 function on pins PA6–PA9 must not be used. When the JTAG port is enabled, bit 6 of the PORT_RXTX register of Port A is always “0”.

Notes

1 – Timer 1;
2, 5, 7, 8, 13 – Timer 2;
3 – SSP1 serial interface;
4 – UART1 serial interface;
6 – UART2 serial interface;
9 – SSP2 serial interface;
10 – UART3 serial interface;
11, 12 – UART ISO7816 serial interface;
14 – SSP3 serial interface;
15 – I2C serial interface;
16, 18, 19 – Timer 4;
17 – Timer 3;
20 – UART4 serial interface

To use a pin in analog mode, the corresponding register fields must be set as follows: MODEx[1:0] = 00, ANALOG_EN[x] = 0, OE[x] = 0. To use a pin in the digital I/O port function, the receiver mode must be configured in the PORT_ANALOG register, and the transmitter mode must be configured in the PORT_OE register. To use a pin in a digital function other than the I/O port, the receiver mode must be configured in the

PORT_ANALOG register, while the transmitter mode is set automatically according to the selected function.

18.1. **Description of I/O port registers**

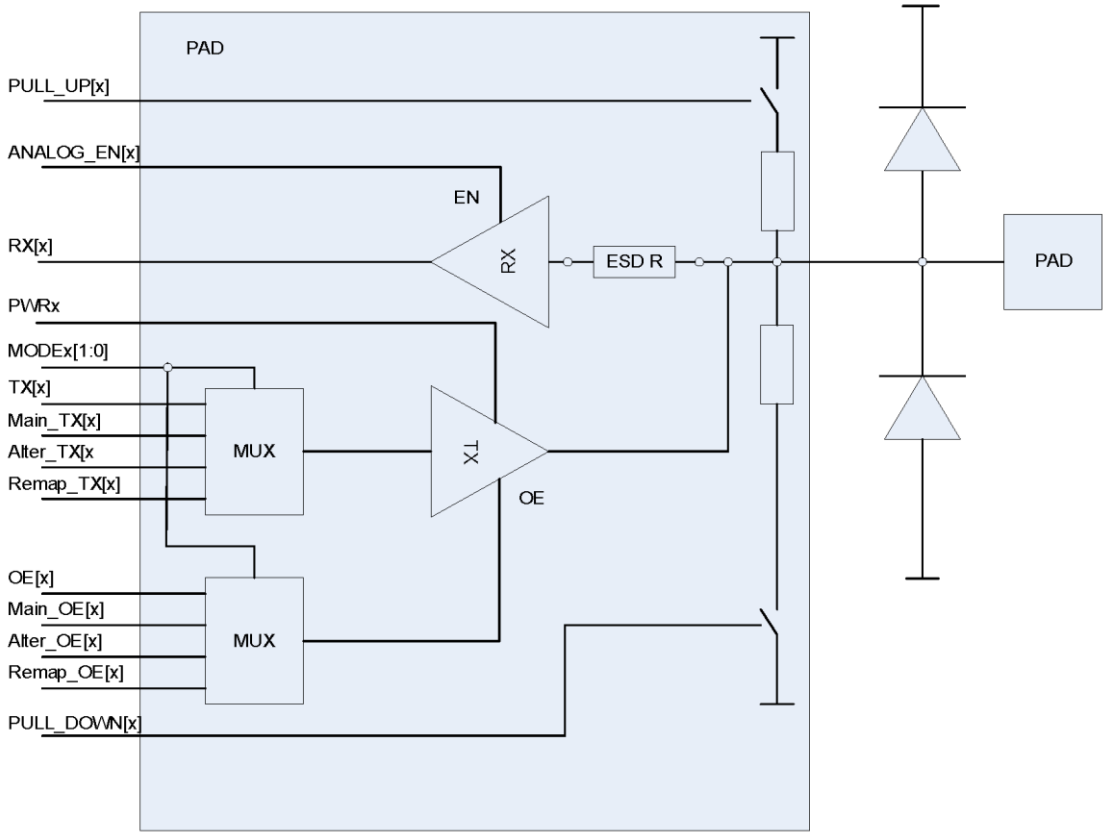


Figure 37 – Block diagram of an input/output port bit

Table 315 – Description of I/O port registers

Base Address	Name	Description
0x5008_0000	PORTA	Port A controller
0x5008_8000	PORTB	Port B controller
0x5009_0000	PORTC	Port C controller
0x500F_0000	PORTD	Port D controller
Offset		
0x00	PORT_RXTX[15:0]	Port data register
0x04	PORT_OE[15:0]	Port transmitter mode register
0x08	PORT_FUNC[31:0]	Port digital function register
0x0C	PORT_ANALOG[15:0]	Port receiver mode register
0x10	PORT_PULL[31:0]	Port pull-up/pull-down configuration register
0x14	-	Reserved
0x18	PORT_PWR[31:0]	Port transmitter drive strength register
0x1C	-	Reserved
0x20	PORT_SETTX[15:0]	Port transmitter output data set register
0x24	PORT_CLRTX[15:0]	Port transmitter output data clear register
0x28	PORT_RDTX[15:0]	Port transmitter output data read register

18.1.1. **PORT_RXTX**

Table 316 – PORT_RXTX Register

Number	31..16	15…0
Access	U	R/W
Reset	0	0
	-	RXTX[15:0]

Table 317 – Description of PORT_RXTX register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...16	-	Reserved
15…0	RXTX[15:0]	Port direct data access register. When read, it shows the port receiver data: – ANALOG_EN=0: 0 – ANALOG_EN=1: the state at the port pin. When written, the value is stored in the port transmitter output data register.

Note: When JTAG operation is enabled, bit 6 of the PORT_RXTX register on port A is always “0.”

18.1.2. **PORT_OE**

Table 318 – PORT_OE Register

Number	31…16	15…0
Access	U	R/W
Reset	0	0
	-	OE[15:0]

Table 319 – Description of PORT_OE register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…16	-	Reserved
15…0	OE[15:0]	Port transmitter operating mode: 0 – transmitter disabled (analog mode); 1 – transmitter enabled for output (digital mode).

18.1.3. **PORT_FUNC**

Table 320 – PORT_FUNC Register

Number	31, 30	…	3, 2	1, 0
Access	R/W	…	R/W	R/W
Reset	0	…	0	0
	MODE15[1:0]	…	MODE1[1:0]	MODE0[1:0]

Table 321 – Description of PORT_FUNC register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...2	MODEx[1:0]	Same as MODE0[1:0] for the other port bits.
1…0	MODE0[1:0]	Function assigned to the port pin: 00 – IO port; 01 – primary function; 10 – alternate function; 11 – Overridden function. When MODEx[1:0] = 00, the transmitter's operating mode and output value are configured automatically according to the selected function.

18.1.4. **PORT_ANALOG**

Table 322 – PORT_ANALOG Register

Number	31…16	15…0
Access	U	R/W
Reset	0	0
	-	ANALOG_EN[15:0]

Table 323 – Description of the PORT_ANALOG register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…16		
15…0	ANALOG_EN[15:0]	Port receiver operating mode: 0 – receiver disabled (analog mode); the receiver output is always "0." In this mode, any voltage (0 to U_{CC}) may be applied to the port pin without increasing the IC power consumption. Transmitter control remains active in this mode. 1 – receiver enabled (digital mode); the receiver outputs the pin state according to the input threshold levels U_{IL} and U_{IH} (Table 594). Additional power consumption of the IC, depending on the voltage applied to the port pin, is shown in Figure 145.

18.1.5. **PORT_PULL**

Table 324 – PORT_PULL Register

Number	31…16	15…0
Access	R/W	R/W
Reset	0	0
	PULL_UP[15:0]	PULL_DOWN[15:0]

Table 325 – Description of PORT_PULL register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...16	PULL_UP[15:0]	Pull-up enable for the port pin: 0 – pull-up to U_{CC} disabled; 1 – pull-up to U_{CC} enabled.
15…0	PULL_DOWN[15:0]	Pull-down enable for the port pin: 0 – pull-down disabled; 1 – pull-down enabled.

18.1.6. **PORT_PWR**

Table 326 – PORT_PWR Register

Number	31	30	…	3	2	1	0
Access	U	R/W	…	U	R/W	U	R/W
Reset	-	0	…	-	0	-	0
		PWR15			PWR1		PWR0

Table 327 – Description of PORT_PWR register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...2	PWRx	Same as PWR0 for the remaining port bits.
1	-	Reserved
0	PWR0	Port transmitter drive strength. For IRL1206FI (CHIP_ID = 217): 0 – nominal pad drive strength (load up to 2 mA). 1 – increased pad drive strength (load up to 4 mA).

18.1.7. **PORT_SETTX**

Table 328 – PORT_SETTX Register

Number	31…16	15…0
Access	U	R/W
Reset	0	0
	-	SETTX[15:0]

Table 329 – Description of PORT_SETTX register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31..16	-	Reserved
15…0	SETTX[15:0]	Port transmitter output data set register Writing “1” sets the corresponding bit in the port output data register to “1”; writing “0” does not change the port output data register. When read, the value stored in the port output data register is returned. **Attention!** The value read from the register may differ from the actual level on the corresponding port pin, for example if the transmitter is disabled.

18.1.8. **PORT_CLRTX**

Table 330 – PORT_CLRTX Register

Number	31…16	15…0
Access	U	R/W
Reset	0	0
	-	CLRTX[15:0]

Table 331 – Description of the PORT_CLRTX register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31..16	-	Reserved
15…0	CLRTX[15:0]	Port transmitter output data clear register Writing “1” sets the corresponding bit in the port output data is cleared to “0”; writing “0” does not change the port output data register. When read, the value stored in the port output data register is returned. **Attention!** The value read from the register may differ from the actual level on the corresponding port pin, for example if the transmitter is disabled.

18.1.9. **PORT_RDTX**

Table 332 – PORT_RDTX Register

Number	31…16	15…0
Access	U	RO
Reset	0	0
	-	RDTX[15:0]

Table 333 – Description of PORT_RDTX register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31..16	-	Reserved
15…0	RDTX[15:0]	Port transmitter output data. **Attention!** The value read from the register may differ from the actual level on the corresponding port pin, for example if the transmitter is disabled.

19 Power supply voltage detector

The power supply voltage detector block (the PVD block) is intended to monitor two supply voltages during operation of the integrated circuit: the main supply voltage U_{CC} and the battery supply voltage U_{CCB}. The PVD block compares external voltage levels with internal reference levels and, when the voltage exceeds or falls below the reference level, generates a signal or interrupt for subsequent software processing.

The reference voltage level for comparison with the main supply voltage is set by the PLS[2:0] bits in the PVDCS register, while the reference level for comparison with the battery supply voltage is set by the PLSB[2:0] bits in the PVDCS register. Based on the voltage levels, the PVD and PVDB flags are generated. These flags are set when the corresponding event occurs and are cleared by software.

Table 334 – Parameters for generation of the PVD and PVDB flags

Parameter name, unit of measurement	Parameter specification		
	minimum	typical	maximum
Supply voltage, V	2.3	-	3.6
Battery domain supply voltage, V	1.8	-	3.6
PVD trigger level, V at PLS = “000”		2.3	
PVDB trigger level, V at PLSB = “000”		1.8	
PVD trigger level, V at PLS = “001”		2.4	
PVDB trigger level, V at PLSB = “001”		2.0	
PVD trigger level, V at PLS = “010”		2.5	
PVDB trigger level, V at PLSB = “010”		2.2	
PVD trigger level, V at PLS = “011”		2.6	
PVDB trigger level, V at PLSB = “011”		2.4	
PVD trigger level, V at PLS = “100”		2.7	
PVDB trigger level, V at PLSB = “100”		2.6	
PVD trigger level, V at PLS = “101”		2.8	
PVDB trigger level, V at PLSB = “101”		2.8	
PVD trigger level, V at PLS = “110”		2.9	
PVDB trigger level, V at PLSB = “110”		3.0	
PVD trigger level, V at PLS = “111”		3.0	
PVDB trigger level, V at PLSB = “111”		3.2	

19.1. **Description of the PVD block registers**

Table 335 – Description of PVD Block Registers

Base Address	Name	Description
0x5005_8000	POWER	Power supply subsystem sensor
Offset		
0x00	PVDCS[26:0]	Power sensor control and status register

19.1.1. **PVDCS**

Table 336 – PVDCS Register

Number	26	25	24	23	22	21	20	19
Access	R/W	U	R/W	U	R/W	R/W	R/W	U
Reset	0	0	0	0	0	0	1	0
	INVB	-	INV	-	IEPVD	IEPVDB	PVD	-

Number	18	15…13	12...6	5…3	2	1	0
Access	R/W	R/W	U	R/W	R/W	U	R/W
Reset	1	000	00000	000	0	0	0
	PVDB	PLS	-	PLSB	PVDEN	-	PVDBEN

Table 337 – Description of PVDCS Register Bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...27	-	Reserved
26	INVB	Flag for inversion of the PVDB sensor output: 0 – no inversion; 1 – inversion enabled. When inversion is disabled, the flag is set when the value exceeds the specified threshold. When inversion is enabled, the flag is set when the value falls below the specified threshold.
25	-	Reserved
24	INV	Flag for inversion of the PVD sensor output: 0 – no inversion; 1 – inversion enabled. When inversion is disabled, the flag is set when the value exceeds the specified threshold. When inversion is enabled, the flag is set when the value falls below the specified threshold.
23	-	Always write “0”
22	IEPVD	Interrupt enable flag for the PVD sensor: 0 – interrupt disabled; 1 – interrupt enabled
21	IEPVDB	Interrupt enable flag for the PVDB sensor: 0 – interrupt disabled; 1 – interrupt enabled

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
20	PVD	Main supply voltage comparison result: 0 – supply voltage is lower than the level specified by PLS; 1 – supply voltage is higher than the level specified by PLS. Cleared by writing 0 twice. If the sensor continues to assert the signal during clearing, the flag will not be cleared.
19	-	Always write “0”
18	PVDB	Battery supply voltage comparison result: 0 – supply voltage is lower than the level specified by PLSB; 1 – supply voltage is higher than the level specified by PLSB. Cleared by writing 0 twice. If the sensor continues to assert the signal during clearing, the flag will not be cleared.
17, 16	-	Reserved
15…13	PLS[2:0]	Voltage level for comparison with the main supply voltage: 000 – greater than 2.3 V; 001 – greater than 2.4 V; … 110 – greater than 2.9 V; 111 – greater than 3.0 V. Voltage step: 100 mV
12…6	-	Reserved
5…3	PLSB[2:0]	Voltage level for comparison with the battery supply voltage: 000 – greater than 1.8 V; 001 – greater than 2.0 V; … 110 – greater than 3.0 V; 111 – greater than 3.2 V. Voltage step: 200 mV
2	PVDEN	Enable bit for the main supply voltage sensor block and the OverVoltage detector block: 0 – sensor disabled; 1 – sensor enabled. Cleared only by the POR circuit. Writable only to 1. *It is recommended to enable the block with PLS = 000, then set the required level, then configure inversion, then enable interrupts.*
1	-	Reserved
0	PVDBEN	Enable bit for the battery supply voltage sensor block: 0 – sensor disabled; 1 – sensor enabled. *It is recommended to enable the block with PLSB = 000, then set the required level, then configure inversion, then enable interrupts.*

20 General-purpose timers

The integrated circuit has four general-purpose timer blocks, and each one can be used for many different applications, including:

- Counting TIM_CLK frequency cycles or counting outside events;
- Making interrupts and DMA requests when set events happen;
- Capturing input signals, including measuring how long the input pulses last;
- Generating different kinds of output signals.

The timers are built on a 32-bit reloadable counter. Counting modes include up-counting, down-counting, and bidirectional counting. The clock source may be the internal TIM_CLK clock, external signals, or other timers.

Each timer block contains four channels, each including an input capture circuit and a PWM block with dead-time insertion and hardware lockout functions.

Each timer supports generation of interrupts and DMA requests.

20.1. Main characteristics

Main characteristics of the timer block:

- 32-bit counter with up-counting, down-counting, or bidirectional counting modes;
- 32-bit TIM_CLK prescaler;
- Clock source selection circuitry for the main counter, allowing selection of external signals or other timers;
- Four independent channels, each of which can operate in one of the following modes:
- Capture mode: captures (stores) the current counter value upon a change of a specified input signal;
- PWM mode: continuously compares programmed values with the counter value to generate output signals;
- PWM output signal generation options:
- Reset to LOW level on match;
- Set to HIGH level on match;
- Toggle (invert) on match;
- Toggle under a specified condition;
- Generation of interrupts and DMA requests on the following events:
- Counter update;
- Capture;
- Compare;
- External events on the ETR and BRK inputs.

20.1.1. **Block diagram**

The block diagram of the Timer block is shown in Figure 38.

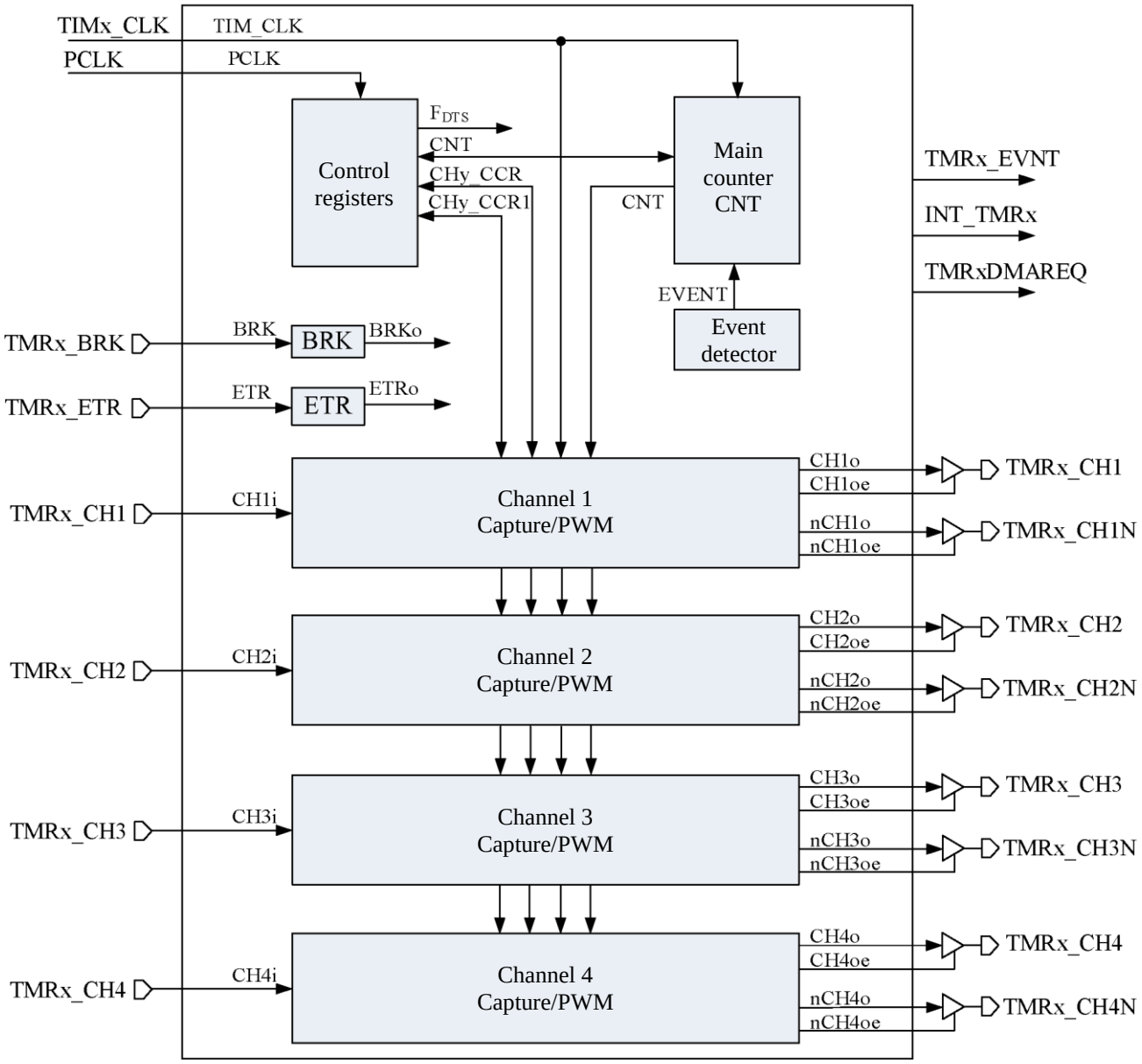


Figure 38 – Block diagram of the Timer block

The timer includes a main 32-bit CNT counter, a control register block, and four capture/PWM channels.

The timer can operate in the following modes:

– Timer;

– Extended timer with several timers combined;

– Capture circuit;

– PWM circuit.

20.2. **Basic timer block**

The timer is built around a 32-bit counter. The basic timer block includes:

- Main timer counter (CNT);
- Counting base (maximum value) of the main counter (ARR);
- TIM_CLK prescaler (PSG) used to clock the main counter;
- Main counter control register (CNTRL).

The signal that updates the main CNT counter may be the internal TIM_CLK clock, events from other counters, or external input signals (see subsection “Event sources for counting”).

20.2.1. **Timer clock initialization**

Before using the timer, the clock signals must first be enabled. These parameters are configured in the Clock signals block.

The general-purpose timer uses PCLK and TIM_CLK as clock sources. PCLK is used to read and write the block registers over the APB bus. The timer blocks operate at the TIM_CLK frequency.

To enable PCLK, the corresponding block clock bit must be set in the PER2_CLOCK register: bit 14 for timer 1, bit 15 for timer 2, bit 27 for timer 3, and bit 28 for timer 4. The TIM_CLK frequency is configured in the TIM_CLOCK register: the TIMxBRG field sets the division factor applied to the PER1_C2 clock to generate TIM_CLK, and the TIMxCLKEN bit controls enabling TIM_CLK to the timer block.

The TIMx_CLK timer clock frequency must be less than or equal to the CPU_CLK core clock frequency.

After the clock signals are supplied, the timer can be used.

20.2.2. **Initialization of the main timer counter**

To start the main counter, you must configure the following:

- The initial value of the main timer counter in the CNT register;
- The counting base for the main counter in the ARR register;
- The counter operating mode in the CNTRL register:
- Select the counter event source EVNT_SEL[3:0];
- The main counter mode CNT_MODE[1:0]:
  - Values 00 and 01 when clocked from the internal source;
  - Value 10 when clocked from external signals;
- The main counter counting direction DIR;
- When clocked from the internal source, set the prescaler value in the PSG register; in this case the main counter will count at the frequency TIM_CLKd = TIM_CLK/(PSG + 1);
- Enable the counter by setting CNT_EN.

The CNT, PSG, and ARR register values can be changed even while the counter is running. The CNT and PSG register values take effect immediately after they are written. The value of the counting base register (ARR) can take effect immediately after writing if bit ARRB_EN = 0 in the CNTRL register.

If bit ARRB_EN = 1 is set, the written ARR value is applied when CNT == ARR. Note that if up-counting is selected for the timer, the new ARR value will be used in the next counting period. If down-counting is selected for the timer, the new ARR value will be used after one counting period.

The CNT_MODE[1:0] field in the CNTRL register defines the operating mode of the main counter:

– CNT_MODE[1:0] = 00 or 10 – the counting direction is defined by the DIR bit:
– DIR = 0 – up-counting;
– DIR = 1 – down-counting;
– CNT_MODE[1:0] = 01 – bidirectional counting with automatic DIR updating.

20.2.3. **Counting modes**

20.2.3.1. **Up-counting: CNT_MODE[1:0] = 00, DIR = 0**

```
TIMERx->CNTRL = 0x00000000;//Timer initialization mode
//Configure main counter operation
TIMERx->CNT = 0x00000004;//Initial counter value
TIMERx->PSG = 0x00000000;//TIM_CLK prescaler
TIMERx->ARR = 0x00000013;//Counting base

//Enable timer operation
TIMERx->CNTRL = 0x00000001;//Up-counting on TIM_CLK
```

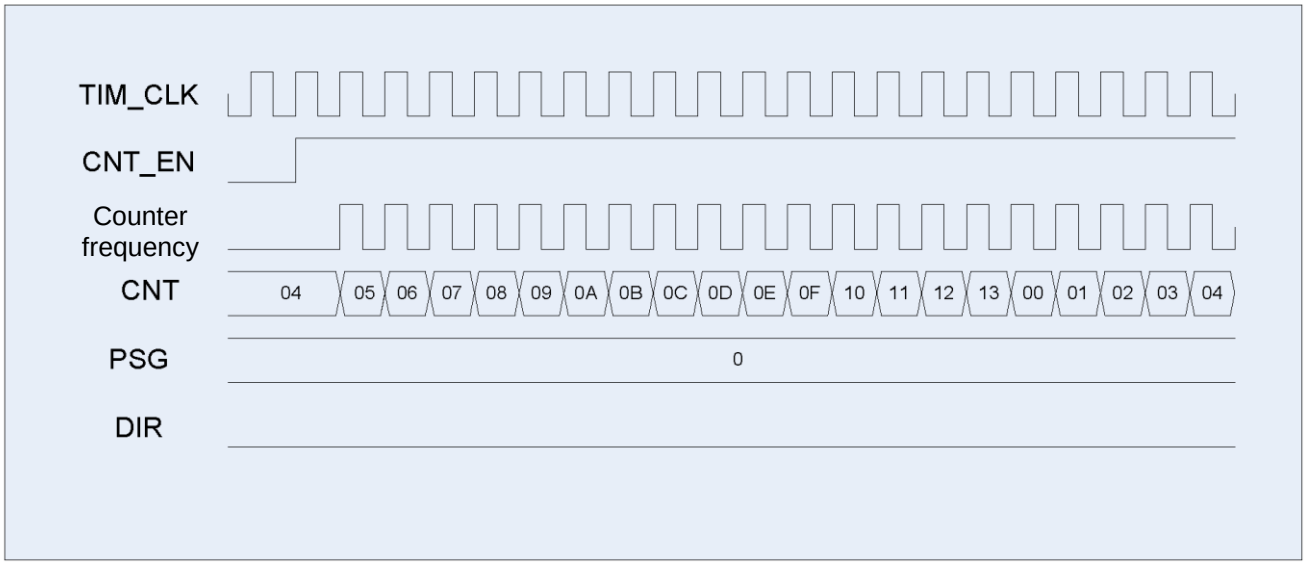


Figure 39 – Timer operation diagrams, up-counting from 0 to 0x13, start value 0x04

20.2.3.2. **Down-counting: CNT_MODE[1:0] = 00, DIR = 1**

TIMERx->CNTRL = 0x00000000;//Timer initialization mode
//Configure main counter operation
TIMERx->CNT = 0x00000004;//Initial counter value
TIMERx->PSG = 0x00000000;//TIM_CLK prescaler
TIMERx->ARR = 0x00000013;//Counting base
//Enable timer operation.
TIMERx->CNTRL = 0x00000009;//Down-counting on TIM_CLK

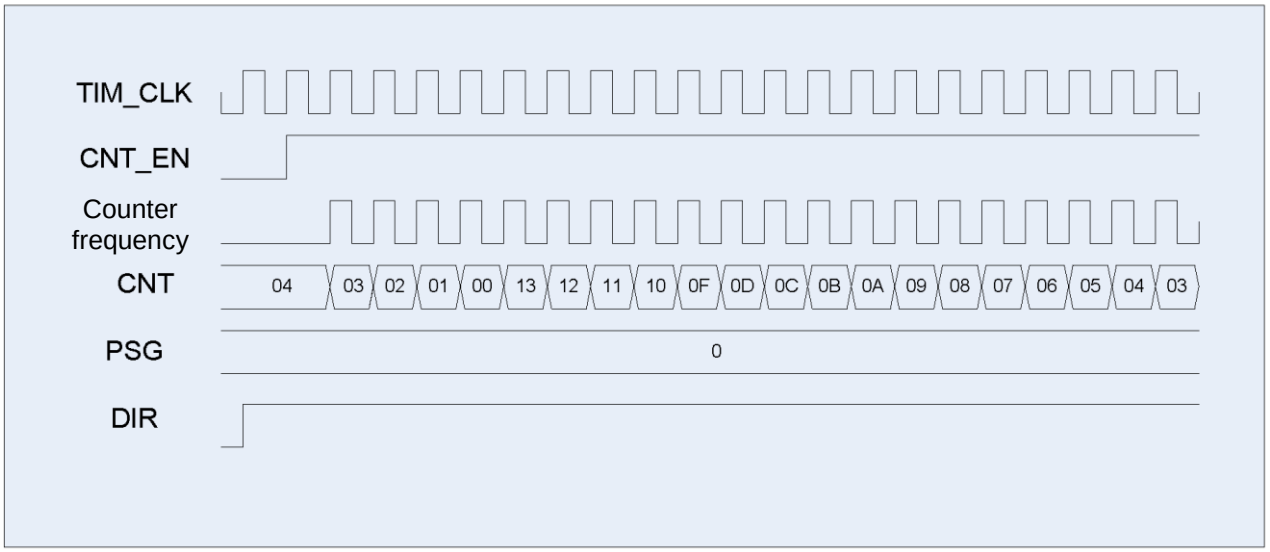


Figure 40 – Timer operation diagrams, down-counting from 0x13 to 0, start value 0x04

20.2.3.3. **Bidirectional counting: CNT_MODE = 01, DIR = 0**

TIMERx->CNTRL = 0x00000000;//Timer initialization mode
//Configure main counter operation
TIMERx->CNT = 0x00000004;//Initial counter value
TIMERx->PSG = 0x00000000;//TIM_CLK prescaler
TIMERx->ARR = 0x00000013;//Counting base
//Enable timer operation.
TIMERx->CNTRL = 0x00000041;//Bidirectional counting on TIM_CLK

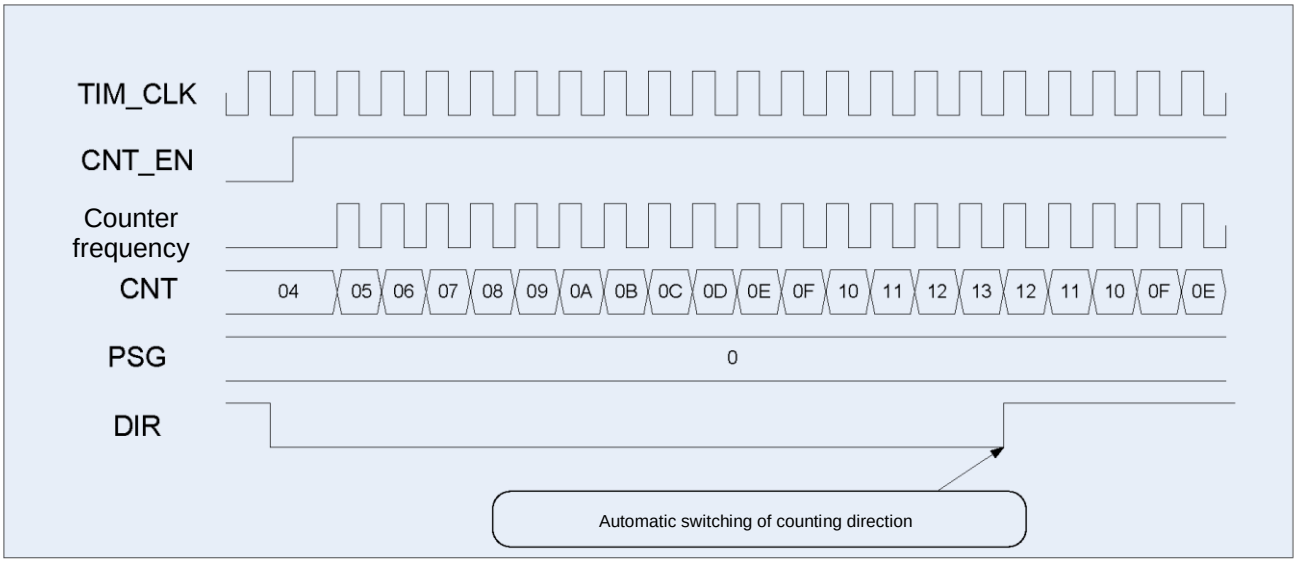


Figure 41 – Timer operation diagrams, bidirectional counting, starting with up-counting

20.2.3.4. **Bidirectional counting: CNT_MODE = 01, DIR = 1**

```
TIMERx->CNTRL = 0x00000000;//Timer initialization mode
//Configure main counter operation
TIMERx->CNT = 0x00000004;//Initial counter value
TIMERx->PSG = 0x00000000;//TIM_CLK prescaler
TIMERx->ARR = 0x00000013;//Counting base
//Enable timer operation.
TIMERx->CNTRL = 0x00000049;//Bidirectional counting on TIM_CLK
```

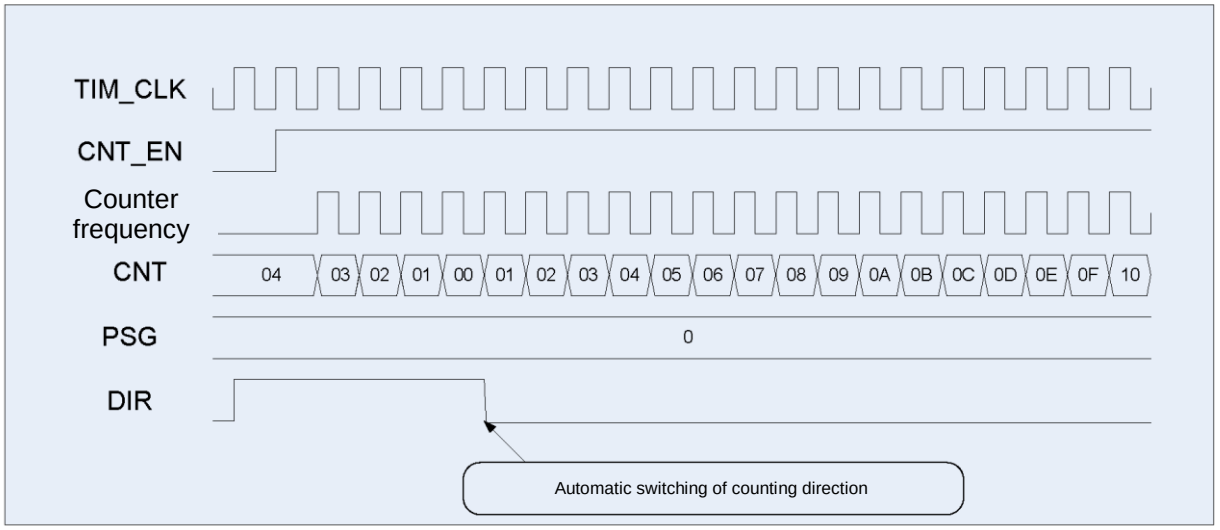


Figure 42 – Timer operation diagrams, bidirectional counting, starting with down-counting

20.2.4. **FDTS clock frequency**

The timer block can generate an additional clock FDTS, which may be used by the dead-time generator and by the digital filters on the ETR and CHyi inputs.

The FDTS clock is derived from TIM_CLK by dividing it by a programmed factor (1, 2, 3, or 4). The FDTS frequency is configured in the CNTRL register, field FDTS[1:0].

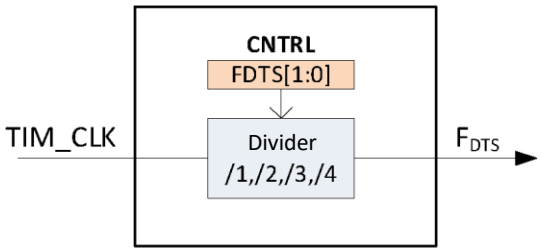


Figure 43 – FDTS clock generation scheme

The diagram of possible FDTS frequencies is shown in Figure 44.

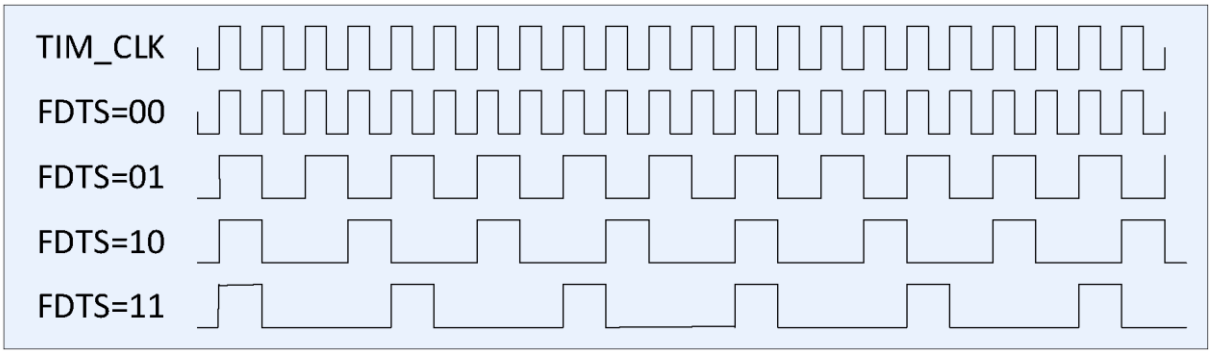


Figure 44 – F_{DTS} clock frequency as a function of the FDTS[1:0] value in the CNTRL register

20.3. **Event sources for counting**

The main timer counter can be clocked from the following sources:

- Internal clock signal (TIM_CLKd);
- Event in another timer (CNT == ARR);
- External clock signal, “Mode 1”: rising-edge event on the CHyi channel input;
- External clock signal, “Mode 2”: rising- or falling-edge event on the ETR input.

The clock source for the main counter is selected in the CNTRL register, field EVNT_SEL[3:0]. When any source other than the internal clock signal is selected (EVNT_SEL[3:0] = 0000), CNT_MODE[1:0] must also be set to 10 in the CNTRL register.

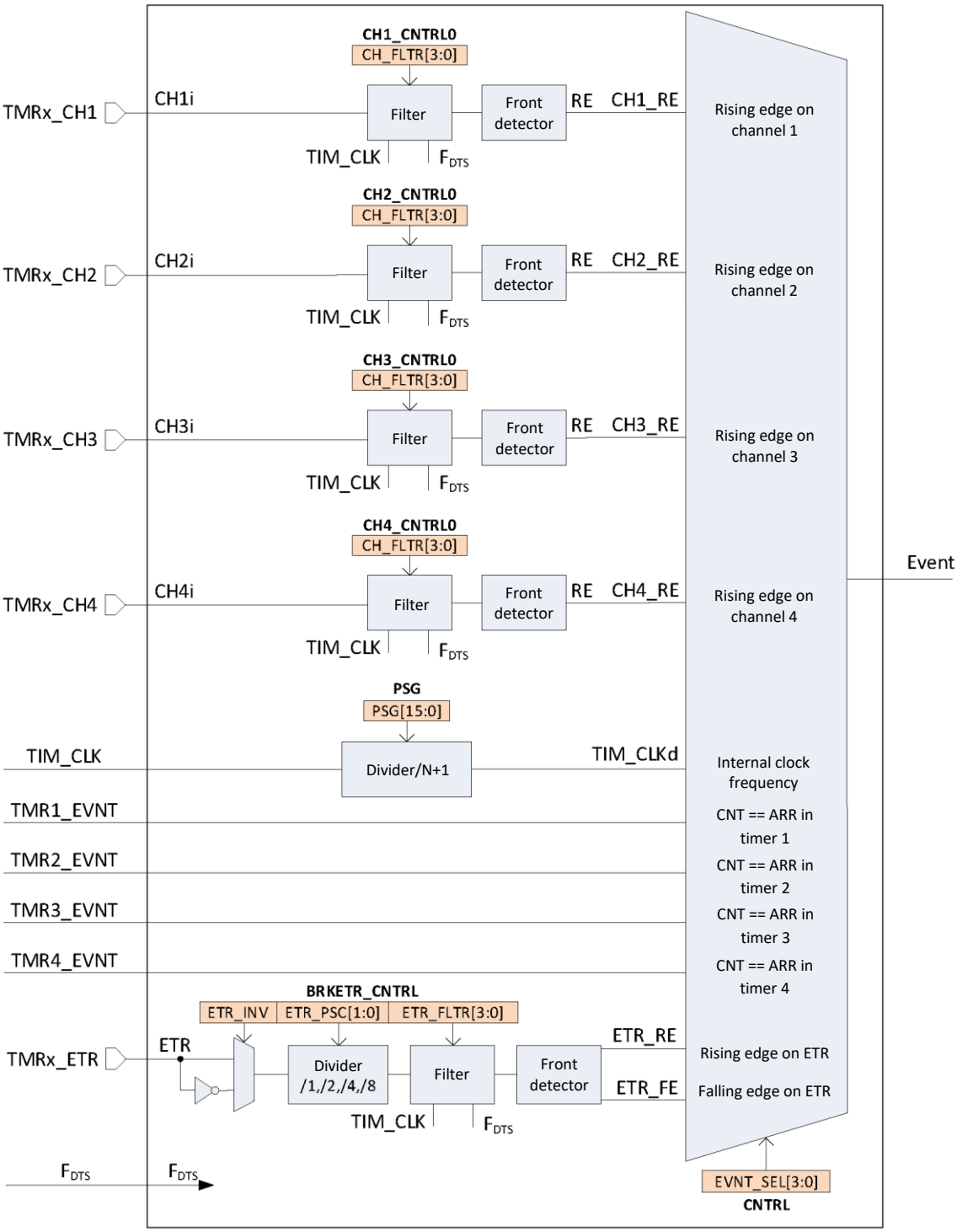


Figure 45 – Block diagram of event generation for counting

20.3.1. **Internal clock signal (TIM_CLKd)**

This mode is selected when EVNT_SEL[3:0] = 0000 and CNT_MODE[1:0] = 0x in the CNTRL register. The main timer counter is clocked from the internal frequency TIM_CLKd, which is obtained by dividing TIM_CLK according to the division factor written to the PSG register.

If the main counter prescaler value (PSG) is not zero, the divider counter is incremented on each TIM_CLK pulse until it reaches the value stored in the prescaler register. Then the divider counter is reset to zero, the main timer counter is changed by 1, and counting starts again. Thus, the output frequency of the prescaler is

$$\text{TIM\_CLKd} = \frac{\text{TIM\_CLK}}{\text{PSG} + 1} \tag{16}$$

The PSG register value can be changed even while the counter is running; the new prescaler value takes effect immediately after it is written. The counter operation diagrams when updating the PSG value are shown in Figures 47 and 48.

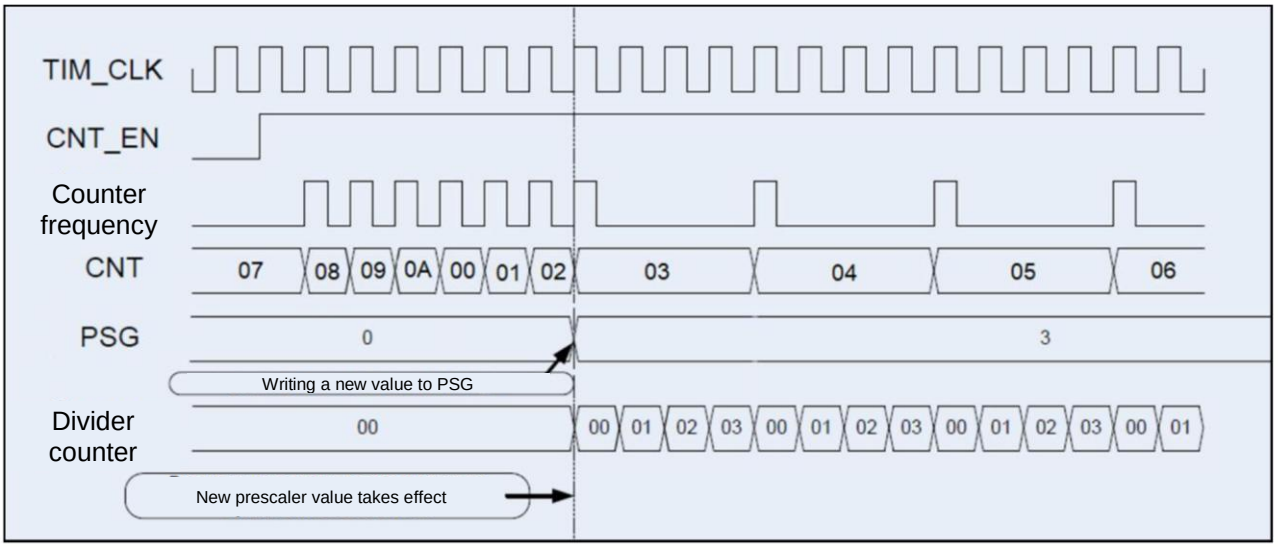


Figure 46 – Counter operation diagram: up-counting
(CNT_MODE[1:0] = 00, EVNT_SEL[3:0] = 0000, DIR = 0)

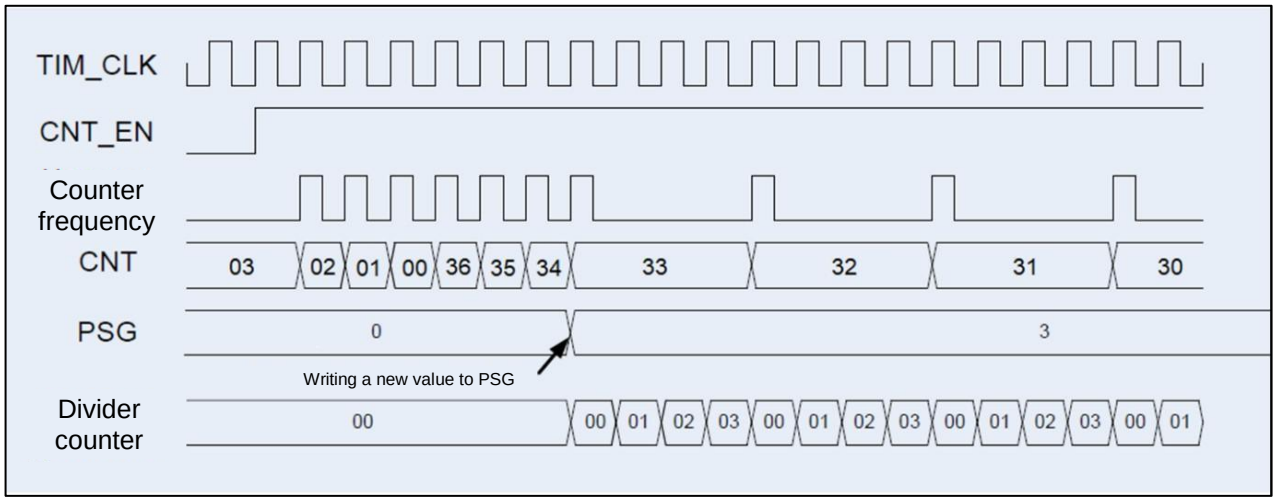


Figure 47 – Counter operation diagram: down-counting
(CNT_MODE[1:0] = 00, EVNT_SEL[3:0] = 0000, DIR = 1)

20.3.2. **Event in another timer (CNT == ARR)**

All timers are fully independent, but they also support synchronized operation. This makes it possible to build more complex timer arrays that operate fully autonomously and do not require any program code to implement complex timing functions.

Each timer has a trigger output TMRx_EVNT, which is connected to the inputs of the other timers. Clocking from another timer is selected when EVNT_SEL[3:0] = 0001–0011 or 1010 and CNT_MODE[1:0] = 10 in the CNTRL register. The main timer counter is clocked from another timer by the TMRx_EVNT signal, which is asserted when CNT == ARR. Resynchronization of the TMRx_EVNT signal (CNT == ARR) from one timer to another occurs with a delay of one TIM_CLK cycle.

Timers can be synchronized in various modes. An example of a cascaded timer configuration is shown in Figure 49, and the timing diagrams for these timers are given in Figure 50.

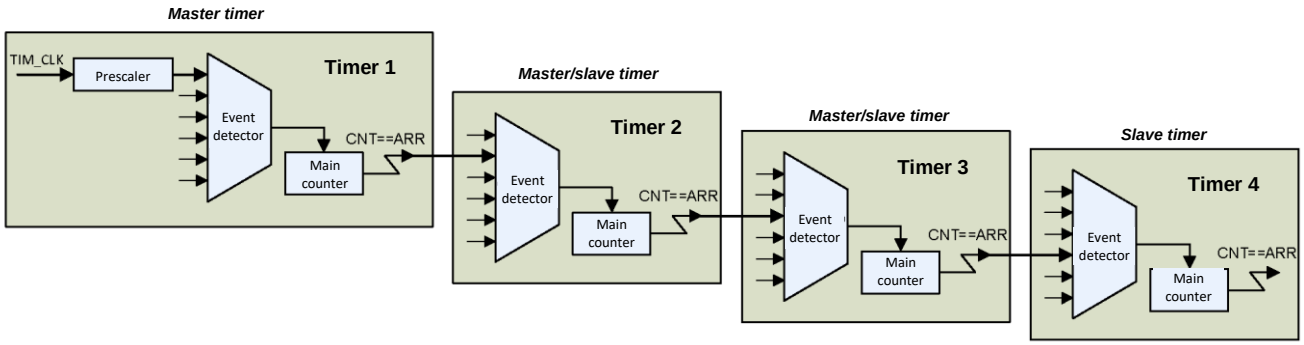


Figure 48 – Example of cascaded connection of timers

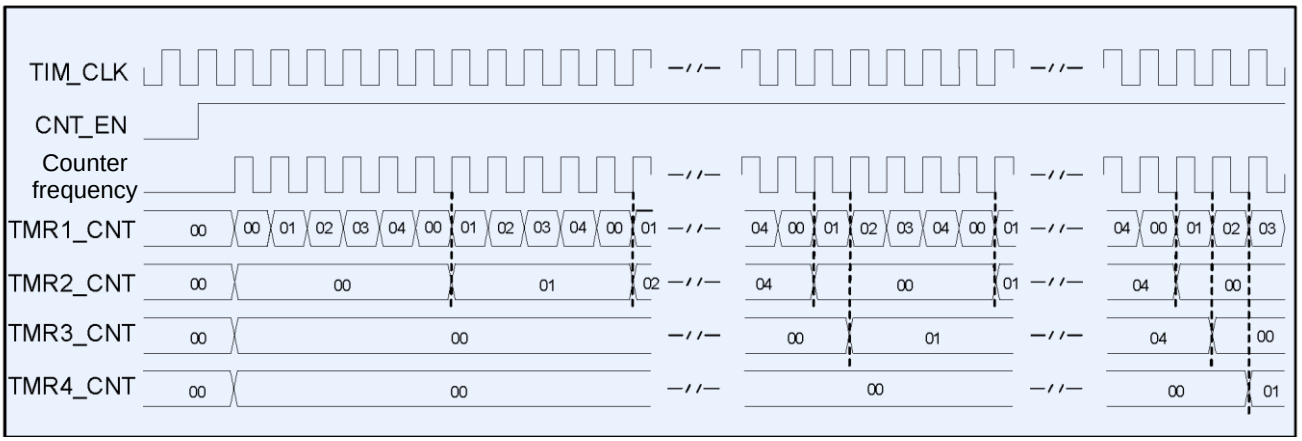


Figure 49 – Timing diagrams for four timers in cascade

TIMER1: DIR = 0, EVNT_SEL[3:0] = 0000, CNT_MODE[1:0] = 00;
TIMER2: DIR = 0, EVNT_SEL[3:0] = 0001, CNT_MODE[1:0] = 10;
TIMER3: DIR = 0, EVNT_SEL[3:0] = 0010, CNT_MODE[1:0] = 10;
TIMER4: DIR = 0, EVNT_SEL[3:0] = 0011, CNT_MODE[1:0] = 10.

20.3.3. External clock signal, “Mode 1”: rising-edge event on the CHyi channel input

This mode is selected when EVNT_SEL[3:0] = 01xx and CNT_MODE[1:0] = 10. The main timer counter counts on the rising edge of the external signal applied to the CHyi channel input. The CH_SEL[1:0] bits of the CHy_CNTRL0 register have no effect, since they are used for timer channel operation only in capture mode.

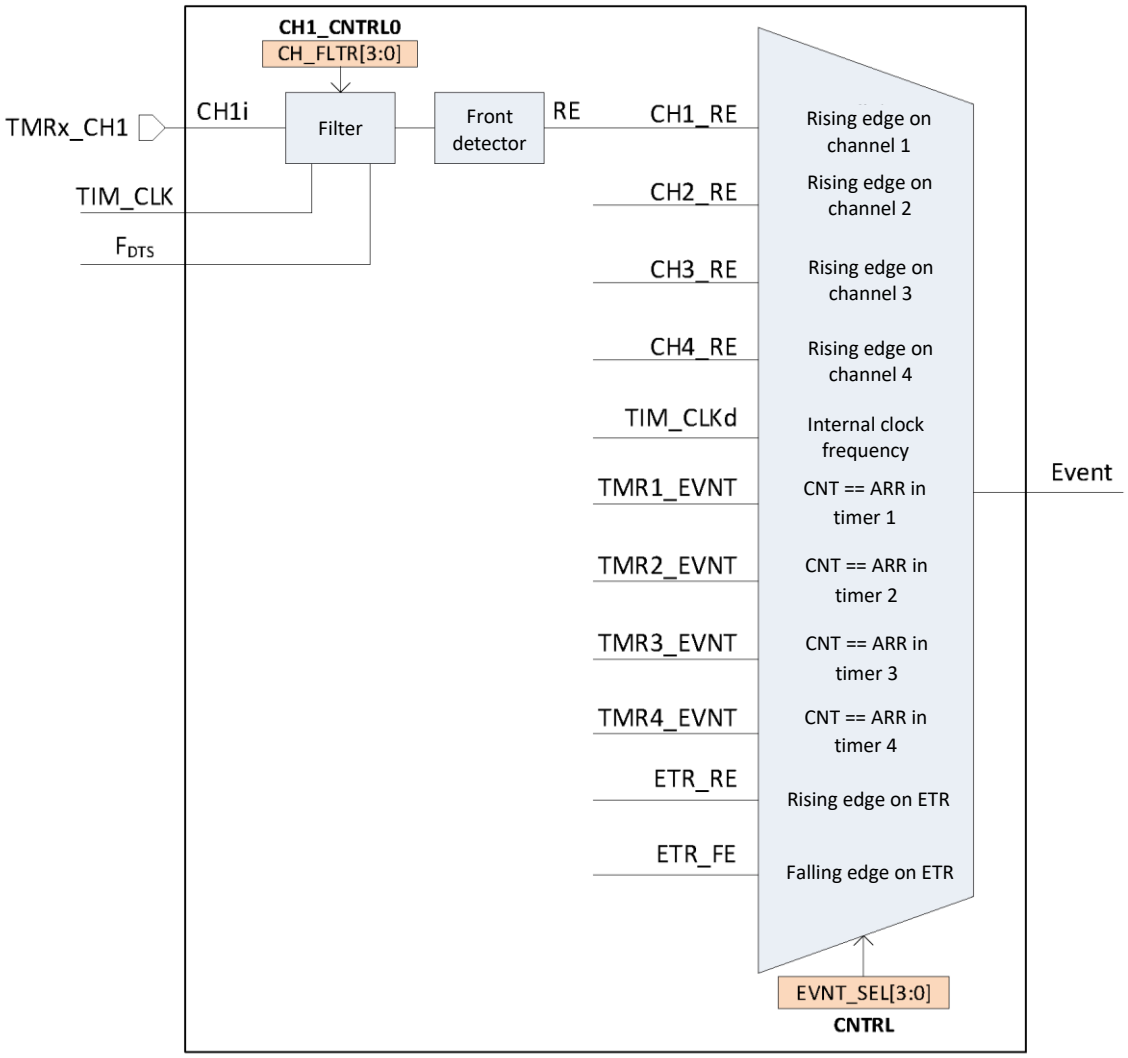


Figure 50 – Clocking scheme from the first channel input

From the CHyi input, the external clock signal is fed to the digital filter block. This block filters the input signal to suppress pulses whose duration is shorter than a specified threshold (see “Digital filter block”). The filter settings are configured in the CH_FLTR[3:0] field of the CHy_CNTRL0 register.

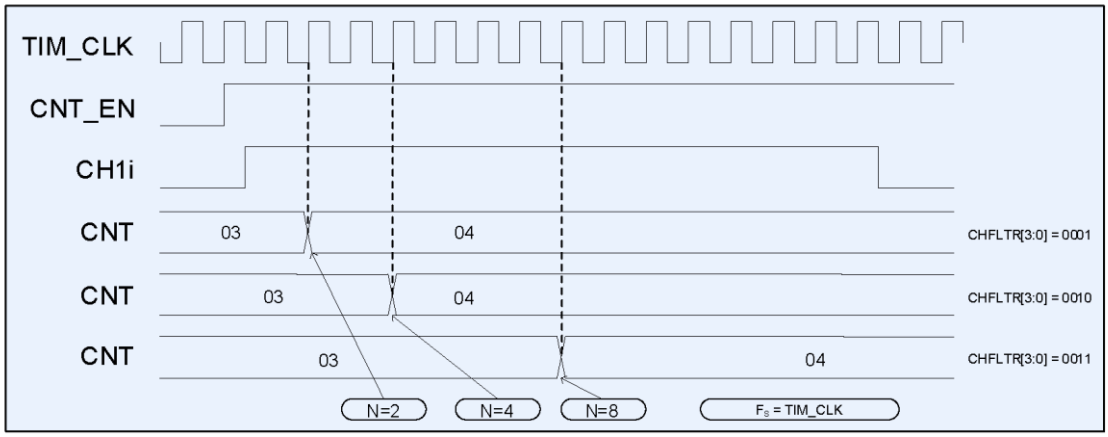


Figure 51 – External clocking timing diagram for different filter options

20.3.4. External clock signal, “Mode 2”: rising- or falling-edge event on the ETR input.

This mode is selected when EVNT_SEL[3:0] = 100x and CNT_MODE[1:0] = 10 in the CNTRL register. The main timer counter can be clocked on either the rising or the falling edge of the external signal applied to the ETR input, depending on the value in the EVNT_SEL[3:0] field.

The clock configuration for the ETR input is set in the BRKETR_CNTRL register. The ETR_INV bit enables inversion of the input signal. The ETR_PSC[1:0] field sets the division factor of the asynchronous prescaler for the external clock (1, 2, 4, or 8). After the prescaler, the clock signal is applied to the digital filter block, where it can be further filtered to suppress pulses whose duration is shorter than a specified threshold (see “Digital filter block”). Filter settings are configured in the ETR_FLTR[3:0] field.

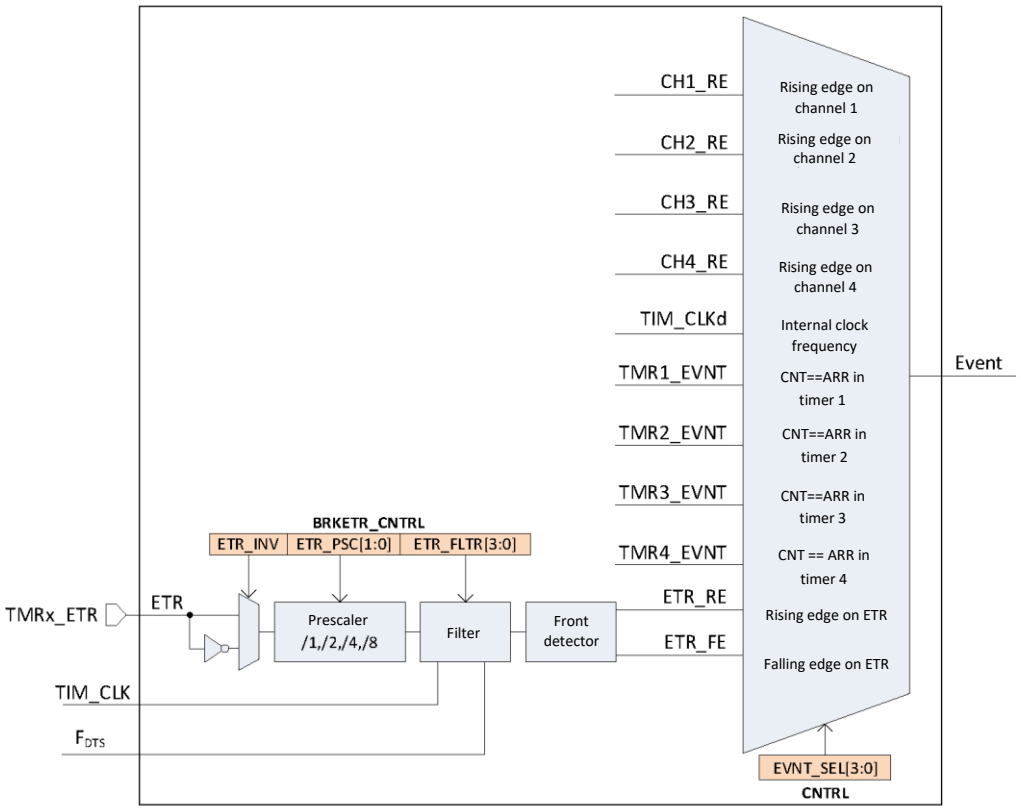


Figure 52 – Clocking scheme from the ETR input

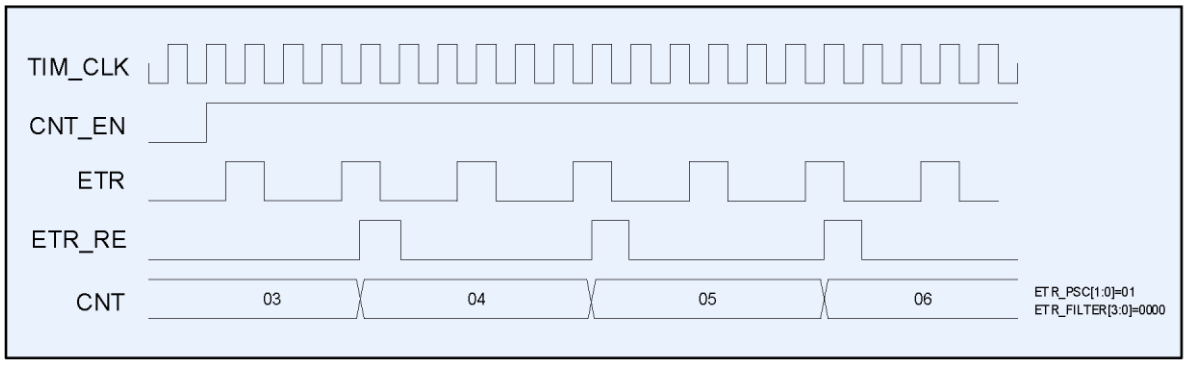


Figure 53 – Clocking diagram from the ETR input, EVNT_SEL[3:0] = 1000

20.4. **Capture mode**

Each timer channel can be independently switched to capture mode. In capture mode, on an event from an external input signal, the current value of the main counter CNT is latched into the CHy_CCR and CHy_CCR1 registers. Events are registered only on the CHyi inputs (pins configured for the TMRx_CHy function).

The block diagram of the capture block is shown in Figure 54.

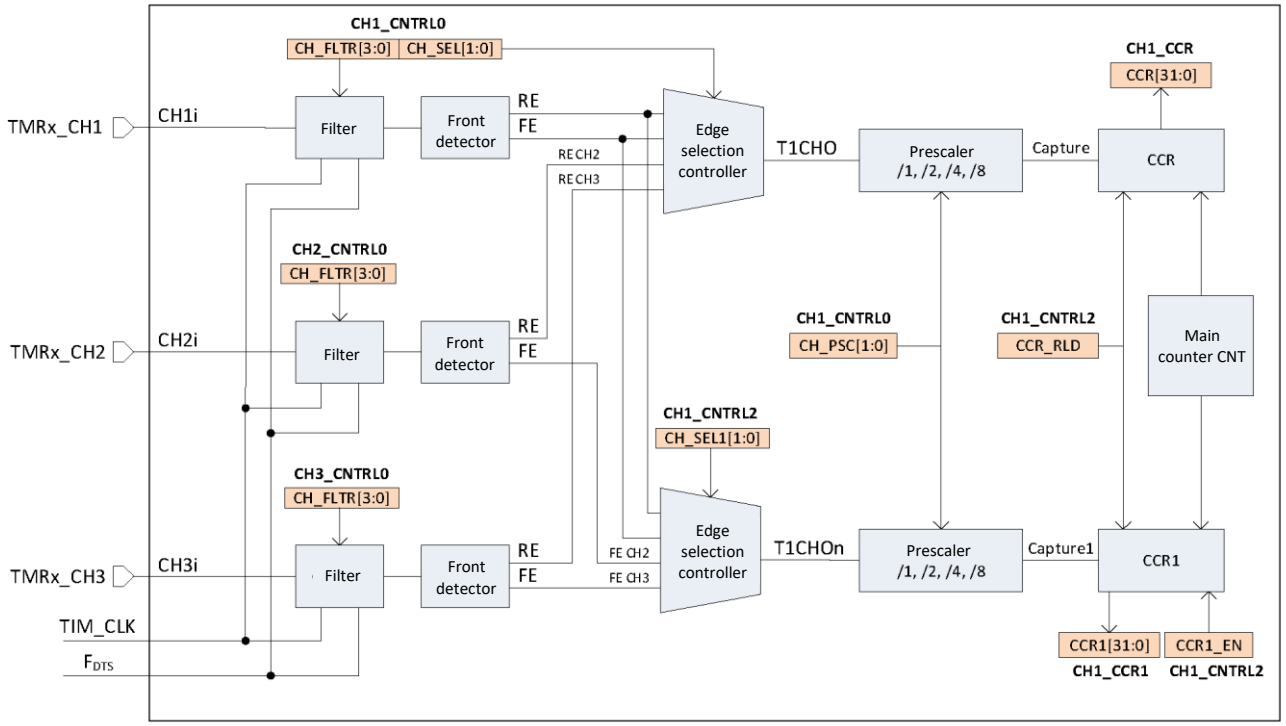


Figure 54 – Block diagram of the capture block (example for channel 1)

To enable capture mode for a specific channel, a “1” must be written to the CAP_NPWM bit of the CHy_CNTRL0 channel control register. To use the CHy_CCR1 register, a “1” must be written to the CCR1_EN bit of the CHy_CNTRL2 register.

The external signal from the CHyi input first passes through the filter block. This block filters the input signal to suppress pulses whose duration is shorter than a specified threshold (see “Digital filter block”). The filter settings are configured in the CH_FLTR[3:0] field of the CHy_CNTRL0 register.

The signal from the filter block is then applied to the Edge detector block. When a positive edge of the input signal is detected, this block generates the RE signal; when a negative edge of the input signal is detected, it generates the FE signal.

In the Edge selection controller block, the capture source is selected from the positive edge of the channel, the negative edge of the channel, or the positive and negative edges of signals from other channels. Configuration of the Edge selection controller block for the CHy_CCR register is performed in the CH_SEL[1:0] field of the CHy_CNTRL0 register, and for the CHy_CCR1 register in the CH_SEL1[1:0] field of the CHy_CNTRL2 register. The selected capture signal is then applied to the prescaler, which, depending on the value in the CH_PSC[1:0] field of the CHy_CNTRL0 register, allows capturing every event, or every second, every fourth, or every eighth event.

The prescaler for the CHy_CCR register generates the Capture signal, and the prescaler for the CHy_CCR1 register generates the Capture1 signal. On the Capture and Capture1 signals, the current value of the main counter CNT is written to the CHy_CCR and CHy_CCR1 registers.

Figure 55 shows an example where the value of the main counter CNT is captured into the CHy_CCR register on the positive edge at the channel input, and into the CHy_CCR1 register on the negative edge at the channel input.

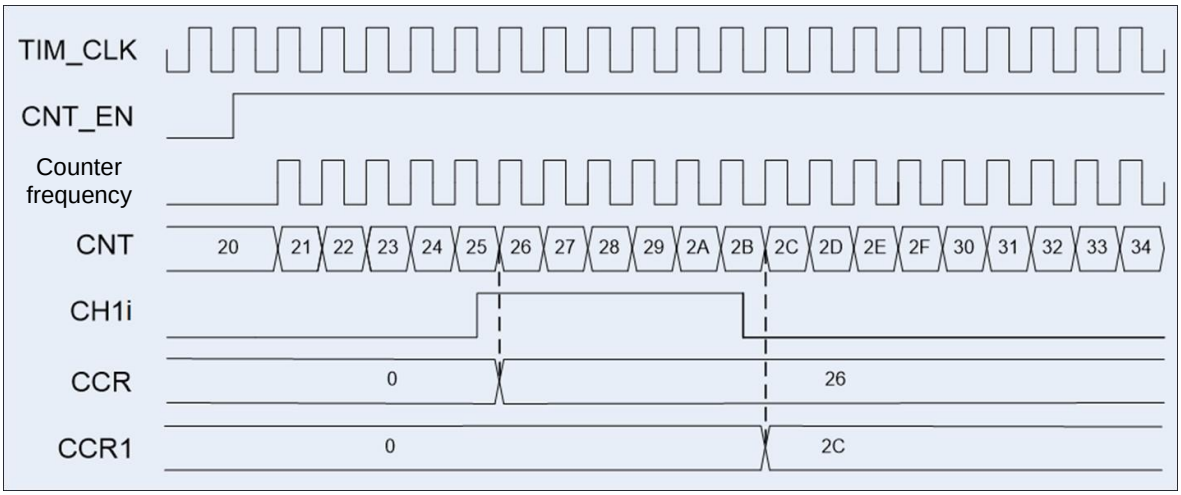


Figure 55 – Capture timing diagram for an event on channel 1 input

On a capture event on a specific channel, interrupt generation can be enabled in the IE register, and DMA request generation can be enabled in the DMA_RE register.

A delay between the capture event and writing the current value of the main counter CNT to the CHy_CCR and CHy_CCR1 registers can be configured using the EVNT_DLY bit in the CHy_CNTRL2 register. If EVNT_DLY is “0”, the capture event is generated first, and then, one TIM_CLK cycle later, CNT is written to the CHy_CCR and CHy_CCR1 registers. If EVNT_DLY is “1”, the contents of the CHy_CCR and CHy_CCR1 registers are updated synchronously with the capture event. When implementing readout of the CHy_CCR and CHy_CCR1 registers on a capture event, it is recommended to set EVNT_DLY to “1”.

20.5. **PWM mode**

Each timer channel can be independently switched to PWM mode to generate output signals with configurable dead time. The block diagram of the PWM generation block is shown in Figure 56.

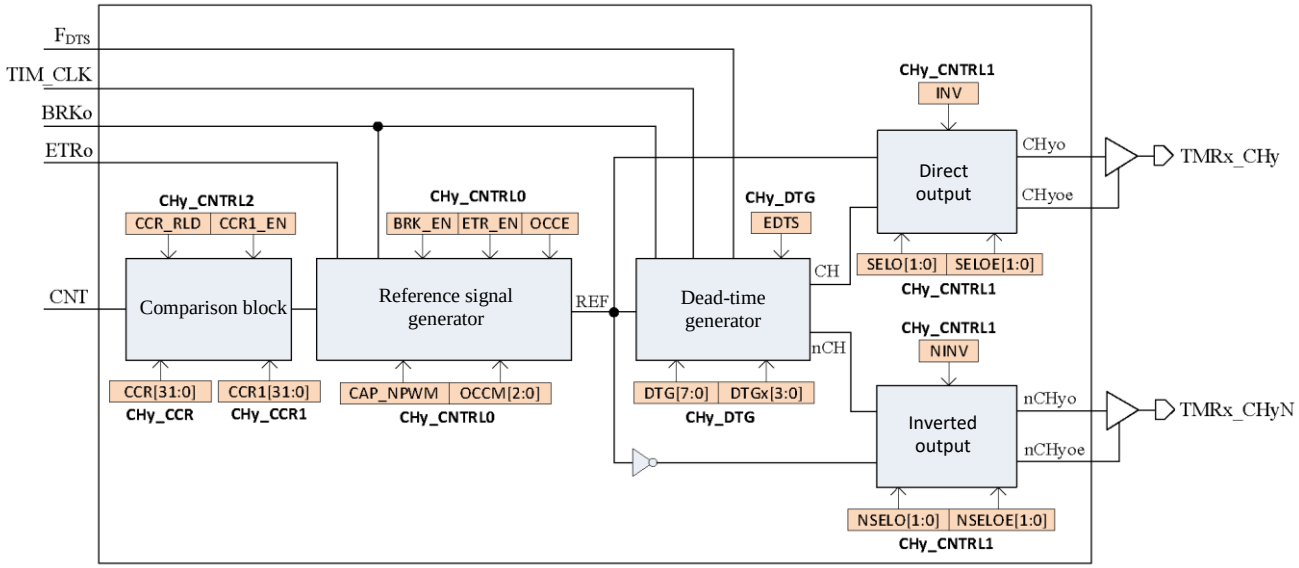


Figure 56 – Block diagram of the PWM generation block

To enable PWM mode for a specific channel, a “0” must be written to the CAP_NPWM bit in the CHy_CNTRL0 channel control register.

20.5.1. **Reference signal generator REF**

In PWM mode, the reference signal generator block produces the REF signal. This signal is generated by comparing the values in the CHy_CCR and CHy_CCR1 registers with the main counter CNT. The REF signal generation format is configured in the timer channel control register CHy_CNTRL0, field OCCM[2:0].

If bit CCR1_EN = 0 in the CHy_CNTRL2 register, only the comparison result between the CHy_CCR register and the main counter CNT is used to generate the REF signal.

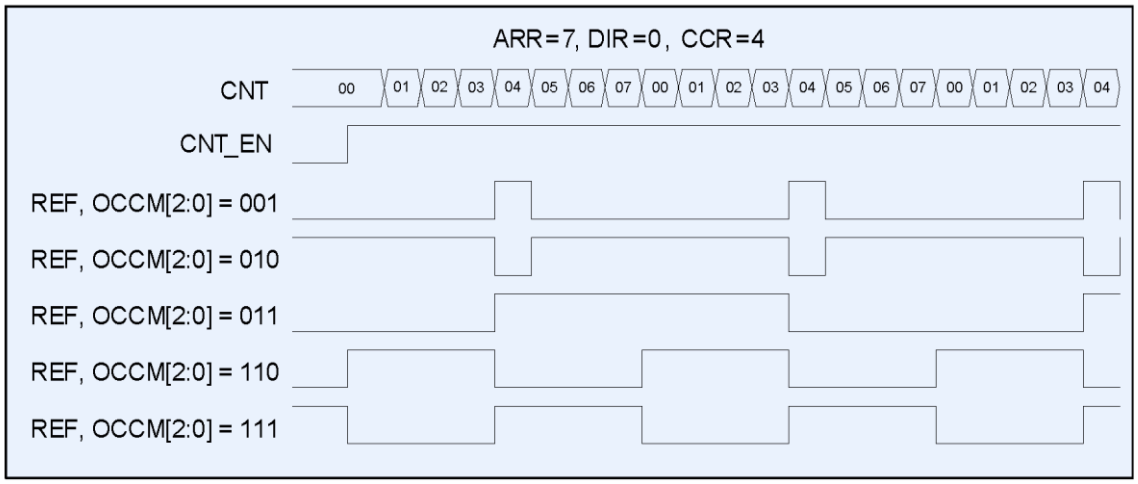


Figure 57– PWM operation timing diagram, CCR1_EN = 0

If bit CCR1_EN = 1 in the CHy_CNTRL2 register, both comparison results between the CHy_CCR and CHy_CCR1 registers and the main counter CNT are used to generate the REF signal.

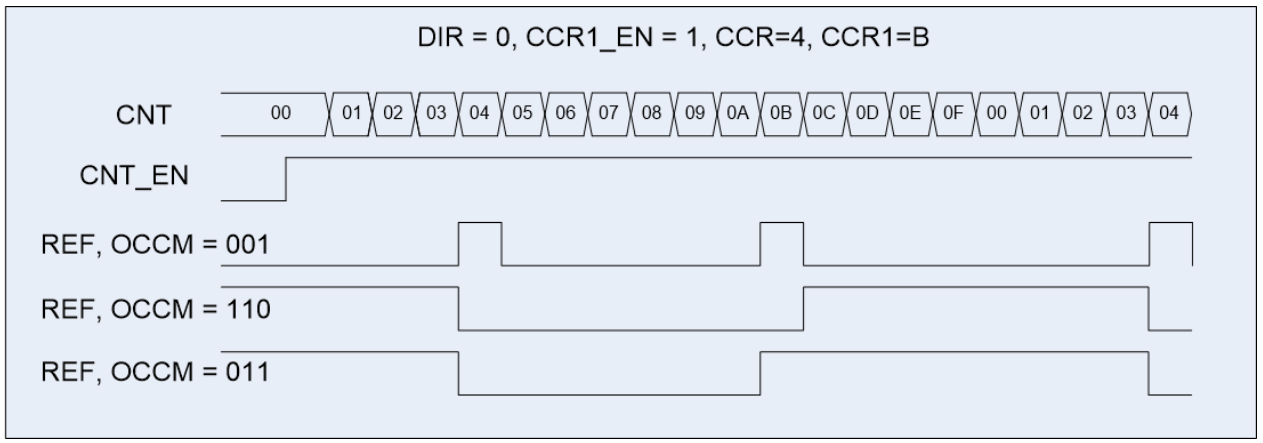


Figure 58 – PWM operation timing diagram, CCR1_EN = 1

Writing new values to the CHy_CCR and CHy_CCR1 registers takes effect immediately if bit CCR_RLD in the CHy_CNTRL2 register is set to “0”; otherwise, the CHy_CCR and CHy_CCR1 registers are updated only when CNT == 0. The process of updating the values in the CHy_CCR and CHy_CCR1 registers is reflected in the CHy_CNTRL0 register by the WR_CMPL and WR_CMPL1 flags, respectively. While the write operation is in progress, the WR_CMPL/WR_CMPL1 flag is set to “1”; when the write operation is complete, the WR_CMPL/WR_CMPL1 flag is cleared to “0”.

The REF signal can be forcibly driven to “0” using an external reset signal from the ETR input (active-high) or from the BRK input (active-low). The active level on the ETR and BRK inputs can be changed by inverting the input signal using the ETR_INV and BRK_INV bits in the BRKETR_CNTRL register, respectively.

To enable REF reset from the ETR input, the ETR_EN and OCCE bits in the CHy_CNTRL0 register must be set. An active level on the ETR input resets the REF signal to “0”. After the active level is removed from the ETR input, the REF signal remains at “0” until the next event that sets REF to “1” (see Figure 59).

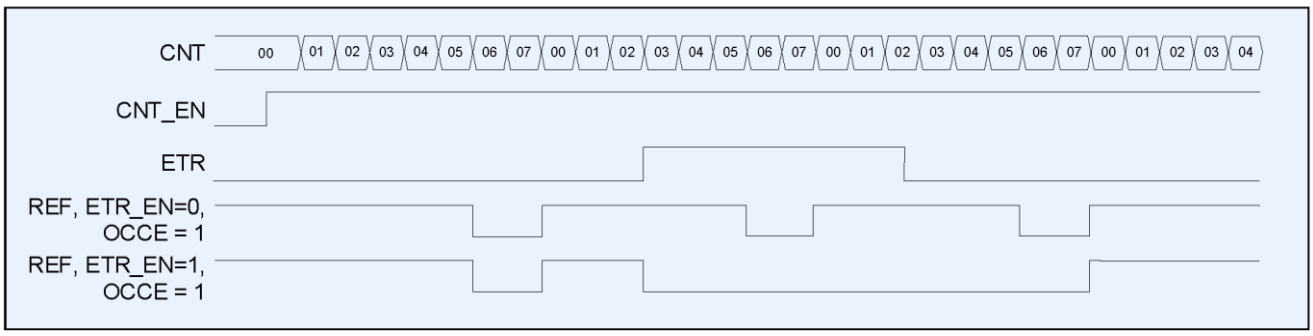


Figure 59 – REF reset timing diagram via the ETR pin

To enable REF reset from the BRK input, the BRK_EN bit in the CHy_CNTRL0 register must be set. An active level on the BRK input resets the REF signal to "0" by masking it. After the active level is removed from the BRK input, REF signal generation is immediately restored (see Figure 60).

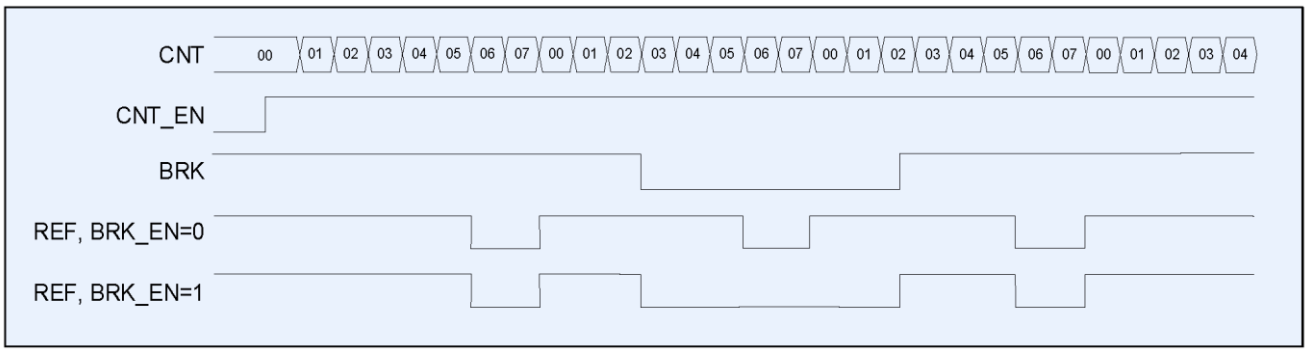


Figure 60 – REF reset timing diagram via the BRK pin

20.5.2. **Dead-time generator**

The dead-time generator block (DTG) generates a complementary pair of signals with dead time based on the REF signal. The outputs of the DTG block are routed to the output blocks as follows:

- The signal on the direct output (CHyo, CHyoe) is the inverted REF signal, whose rising edge is delayed by DTGdel relative to the falling edge of the REF reference signal;
- the signal on the inverted output (nCHyo, nCHyoe) is the REF signal, whose rising edge is delayed by DTGdel relative to the rising edge of the REF reference signal.

The dead time between the signals on the direct and inverted outputs is calculated in TIM_CLK or F_{DTS} clock cycles according to the formula

$$\mathrm{DTGdel} = \mathrm{DTG} \cdot (\mathrm{DTGx} + 1), \tag{17}$$

where DTGx is the prescaler;
DTG is the main divider.

The DTG block is controlled via the CHy_DTG register. The clock source for dead-time generation is selected by the EDTS bit. The divider values DTGx and DTG are set in the DTGx[3:0] and DTG[7:0] fields, respectively. If the delay DTGdel exceeds the high-level pulse width, the corresponding pulse is not generated.

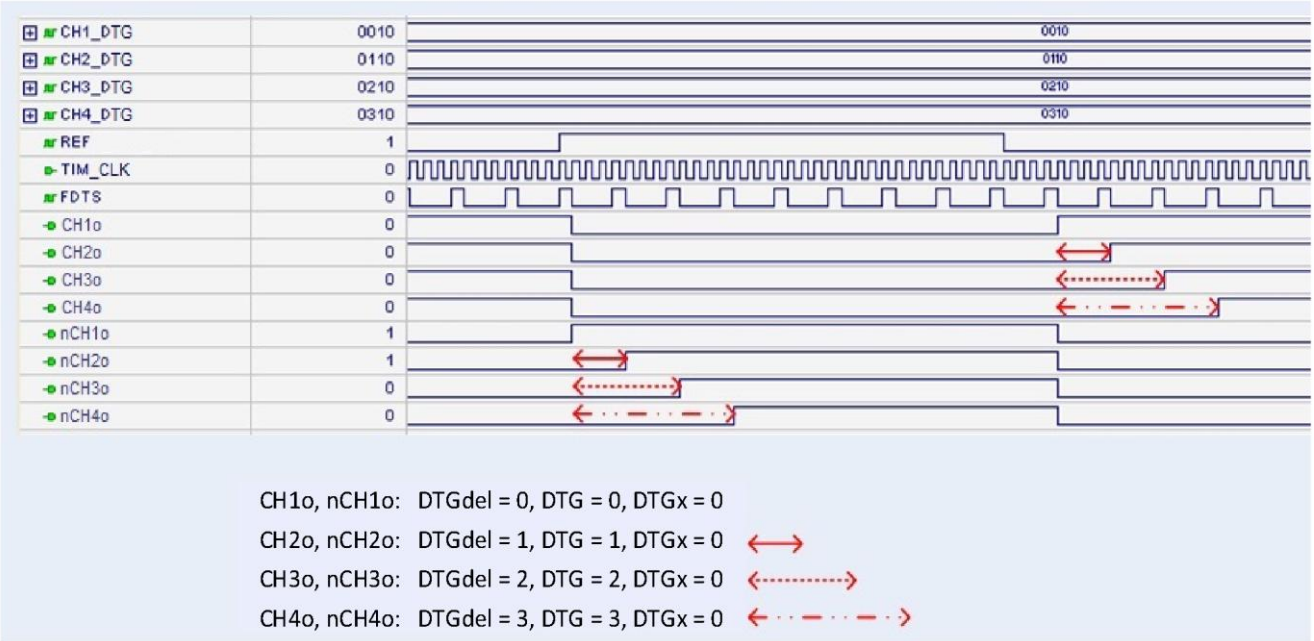


Figure 61 – DTG block operation timing diagram

The DTG block output signals can be forcibly driven to “0” using an external reset signal from the BRK input (active-low). The active level on the BRK input can be changed by inverting the input signal using the BRK_INV bit in the BRKETR_CNTRL register. To enable reset of the DTG block output signals from the BRK input, the BRK_EN bit in the CHy_CNTRL0 register must be set. An active level on the BRK input resets the DTG block output signals to “0” by masking them.

20.5.3. **Output blocks**

Each timer channel has two output blocks – direct and inverted. Each output block generates both an output signal (CHyo, nCHyo) and an output-enable signal (CHyoe, nCHyoe). For the direct (CHyo, CHyoe) and inverted (nCHyo, nCHyoe) outputs, the CHy_CNTRL1 register can select either constant levels (0 or 1) or signals derived from the REF signal. These signals include the REF signal itself as well as the signals generated by the DTG block.

The source of the output signal for the direct (CHyo) and inverted (nCHyo) outputs is selected in the SELO[1:0] and NSELO[1:0] fields of the CHy_CNTRL1 register. Additionally, each output signal for the direct (CHyo) and inverted (nCHyo) outputs can be inverted by setting the INV and NINV bits in the CHy_CNTRL1 register. The source of the output-enable signal for the direct (CHyoe) and inverted (nCHyoe) outputs is selected in the SELOE[1:0] and NSELOE[1:0] fields of the CHy_CNTRL1 register. When the output-enable signal is “0”, the corresponding pin operates as an input; when the output-enable signal is “1”, it operates as an output.

20.6. **Digital filter block**

On the ETR input path and on the timer channel inputs CHyi there is a digital filter block that removes high- and low-level input pulses whose duration is shorter than a specified threshold.

The filter configuration for the ETR input is set in the ETR_FLTR[3:0] field of the BRKETR_CNTRL register; for the CHyi channel inputs it is set in the CH_FLTR[3:0] field of the CHy_CNTRL0 register. The value written to these fields configures two filter parameters:

- Sampling frequency F_S, at which the input signal is captured into a shift register for accumulation. TIM_CLK frequency or F_{DTS} may be used as the sampling frequency F_S;
- The number of samples (filter length) N for which the input signal must remain stable in order not to be filtered out.

If, during the specified number of samples N at the frequency F_S, the input signal does not change, the input value is passed to the filter output. Otherwise, the internal accumulation counter is reset and signal capture starts again.

Thus, configuring the sampling frequency F_S and the number of samples N sets the minimum duration of input pulses that will not be filtered out. A timing diagram of filter operation when using TIM_CLK as the sampling frequency is shown in Figure 62, with F_S = TIM_CLK and N = 4.

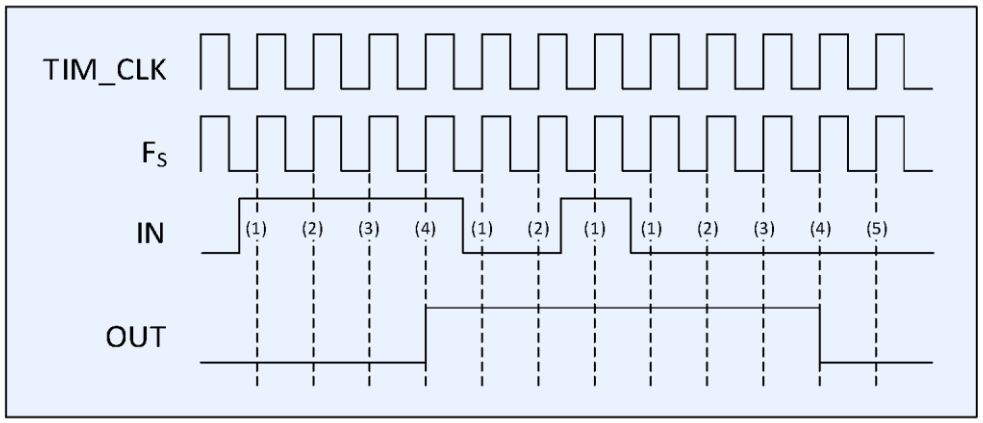


Figure 62 – Filter operation timing diagram, F_S = TIM_CLK, N = 4

To provide longer accumulation intervals for the input signal, the F_{DTS} frequency can be used. It is derived from TIM_CLK by dividing it by a programmed factor (see "F_{DTS} clock frequency").

A timing diagram of filter operation when using the F_{DTS} frequency is shown in Figure 64, with sampling frequency $F_S = F_{DTS}/2$ and N = 6.

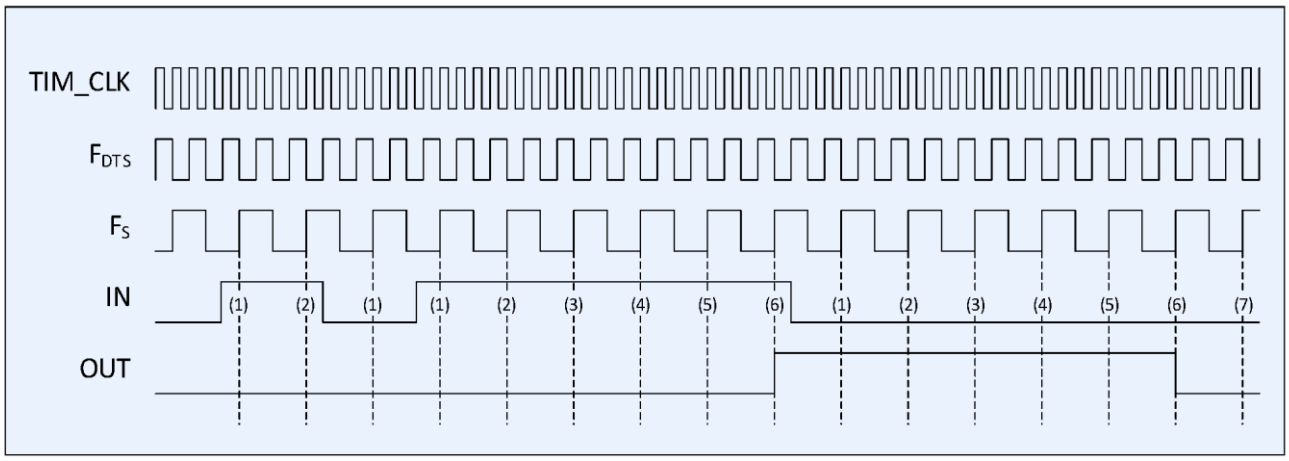


Figure 63 – Filter operation timing diagram, $F_S = F_{DTS}/2$, N = 6

Possible digital filter configurations are given in Table 338.

Table 338 – Possible filter configurations

CH_FLTR[3:0], ETR_FLTR[3:0]	Sampling frequency Fs	Number of samples N	Minimum pulse duration that will not be filtered out
0000	F_{DTS}	1	-
0001	TIM_CLK	2	2 x T_{TIM_CLK}
0010	TIM_CLK	4	4 x T_{TIM_CLK}
0011	TIM_CLK	8	8 x T_{TIM_CLK}
0100	$F_{DTS}/2$	6	12 x T_{FDTS}
0101	$F_{DTS}/2$	8	16 x T_{FDTS}
0110	$F_{DTS}/4$	6	24 x T_{FDTS}
0111	$F_{DTS}/4$	8	32 x T_{FDTS}
1000	$F_{DTS}/8$	6	48 x T_{FDTS}
1001	$F_{DTS}/8$	8	64 x T_{FDTS}
1010	$F_{DTS}/16$	5	80 x T_{FDTS}
1011	$F_{DTS}/16$	6	96 x T_{FDTS}
1100	$F_{DTS}/16$	8	128 x T_{FDTS}
1101	$F_{DTS}/32$	5	160 x T_{FDTS}
1110	$F_{DTS}/32$	6	192 x T_{FDTS}
1111	$F_{DTS}/32$	8	256 x T_{FDTS}

20.7. Status flags, interrupts, and DMA requests

During operation, the timer block monitors the state of its internal blocks and generates 17 events:

- CNT_ZERO_EVNT – the CNT counter value matches zero;
- CNT_ARR_EVNT – the CNT counter value matches the ARR register value;
- ETR_RE_EVNT – rising edge detected on the ETR input;
- ETR_FE_EVNT – falling edge detected on the ETR input;
- BRK_EVNT – high level detected on the BRK input;
- CCR_CAP_EVNT[3:0] – CNT value written to the CHy_CCR register on capture of the configured edge at the CHyi channel input; events are generated individually for each channel;
- CCR_REF_EVNT[3:0] – rising edge detected at the output of the REF reference signal generator; events are generated individually for each channel;
- CCR_CAP1_EVNT[3:0] – CNT value written to the CHy_CCR1 register on capture of the configured edge at the CHyi channel input; events are generated individually for each channel;

20.7.1. Status flags

When an event occurs, the corresponding flag in the STATUS register is set. Flags in the STATUS register are cleared by writing “0”; writing “1” has no effect. If writing “0” coincides with a new event, the new event has priority.

20.7.2. Interrupts

Based on the flags in the STATUS register, the timer block generates a single interrupt request signal INT_TMRx. The flags that participate in the interrupt request are selected performed via the interrupt enable register IE. When the interrupt request is formed, the masked flags statuses from the STATUS register are combined using a logical OR.

20.7.3. DMA requests

Based on the monitored events, the timer block generates the DMA request signal TMRxDMAREQ. The events that generate a DMA request are selected via the DMA_RE register.

20.8. **Examples**

This section provides examples of initializing timer 1 in various operating modes. Initialization for the other timers is performed in the same way.

20.8.1. **Basic counter mode**

```
RST_CLK->PER2_CLOCK |= 0x00004000; //Enable PCLK for timer 1
RST_CLK->TIM_CLOCK = 0x01000000;      //Configure and supply TIM_CLK
                                      //for timer 1 (TIM_CLK = PER1_C2)
TIMER1->CNTRL = 0x00000000; //Timer initialization mode
//Configure main counter operation
TIMER1->CNT = 0x00000000; //Initial counter value
TIMER1->PSG = 0x00000000; //TIM_CLK prescaler
TIMER1->ARR = 0x0000000F; //Counting base
TIMER1->IE = 0x00000002; //Enable interrupt when CNT = ARR
//Enable timer operation
TIMER1->CNTRL = 0x00000001; //Up-counting on TIM_CLK
```



Figure 64 – Basic counter mode

20.8.2. **Capture mode**

```
RST_CLK->PER2_CLOCK |= 0x00004000; //Enable PCLK for timer 1
RST_CLK->TIM_CLOCK = 0x01000000;      //Configure and supply TIM_CLK
                                     //for timer 1 (TIM_CLK = PER1_C2)
TIMER1->CNTRL = 0x00000000;//Timer initialization mode
//Configure main counter operation
TIMER1->CNT = 0x00000000;//Initial counter value
TIMER1->PSG = 0x00000000;//TIM_CLK prescaler
TIMER1->ARR = 0x000000FF;//Counting base
TIMER1->IE = 0x000001E0;//Enable interrupt generation on events//
TIMER1->CH1_CNTRL0 = 0x00008000;//Capture on positive edge at CH1i input,//filtering
                                     disabled
TIMER1->CH2_CNTRL0 = 0x00008010;//Capture on negative edge at CH2i input,//filtering
                                     disabled
TIMER1->CH3_CNTRL0 = 0x00008001;//Capture on positive edge at CH3i input,//filtering over//2
                                     samples at TIM_CLK
TIMER1->CH4_CNTRL0 = 0x00008011;//Capture on negative edge at CH4i input,//filtering over//2
                                     samples at TIM_CLK
//Channel output mode – channel operates as input
TIMER1->CH1_CNTRL1= 0x00000000;
TIMER1->CH2_CNTRL1= 0x00000000;
```

TIMER1->CH3_CNTRL1= 0x00000000;
TIMER1->CH4_CNTRL1= 0x00000000;
//Enable timer operation
TIMER1->CNTRL = 0x00000001;//Up-counting on TIM_CLK

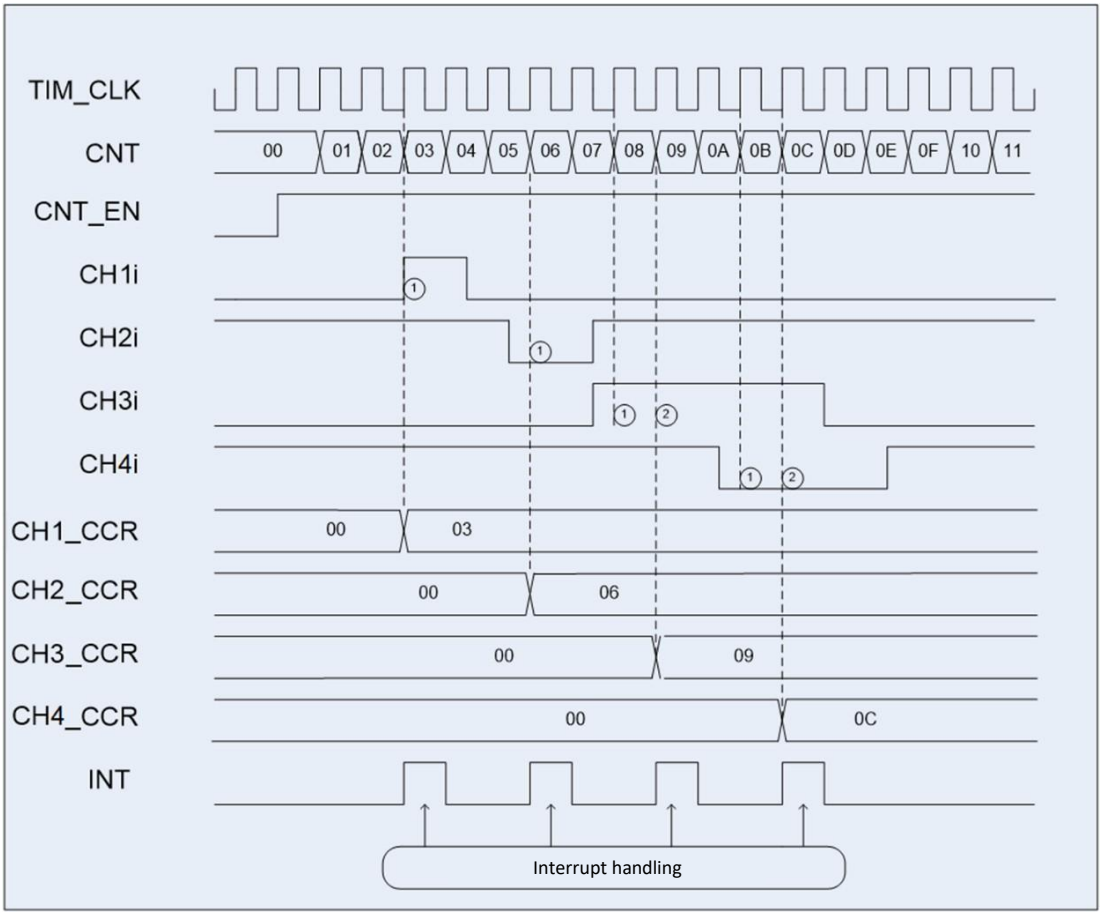


Figure 65 – Example timing diagrams in capture mode

20.8.3. **PWM mode**

RST_CLK->PER2_CLOCK |= 0x00004000;//Enable PCLK for timer 1
RST_CLK->TIM_CLOCK = 0x01000000; //Configure and supply TIM_CLK//for timer 1 (TIM_CLK = PER1_C2)
TIMER1->CNTRL = 0x00000000;//Timer initialization mode
//Configure main counter operation
TIMER1->CNT = 0x00000000; //Initial counter value
TIMER1->PSG = 0x00000000; //Frequency prescaler TIM_CLK
TIMER1->ARR = 0x00000010; Counting base
TIMER1->IE = 0x00001E00; //Enable interrupt generation on events//front edge at the REF output for all channels
//Channel operating mode – PWM
TIMER1->CH1_CNTRL0 = 0x00000200;//REF = 1 if CNT == CCR
TIMER1->CH2_CNTRL0 = 0x00000200;//REF = 1 if CNT == CCR
TIMER1->CH3_CNTRL0 = 0x00000400;//REF = 0 if CNT == CCR
TIMER1->CH4_CNTRL0 = 0x00000600;//Toggle REF if CNT == CCR
//Channel output mode – channel operates as output
//REF signal is routed to the channel outputs
TIMER1->CH1_CNTRL1= 0x00000909;
TIMER1->CH2_CNTRL1= 0x00000909;
TIMER1->CH3_CNTRL1= 0x00000909;
TIMER1->CH4_CNTRL1= 0x00000909;

//Sets the CCR values used as CNT compare thresholds in PWM mode
TIMER1->CH1_CCR = 0x00000003;
TIMER1->CH2_CCR = 0x00000006;
TIMER1->CH3_CCR = 0x00000009;
TIMER1->CH4_CCR = 0x0000000F;
//Enable timer operation
TIMER1->CNTRL = 0x00000001;//Up-counting on TIM_CLK

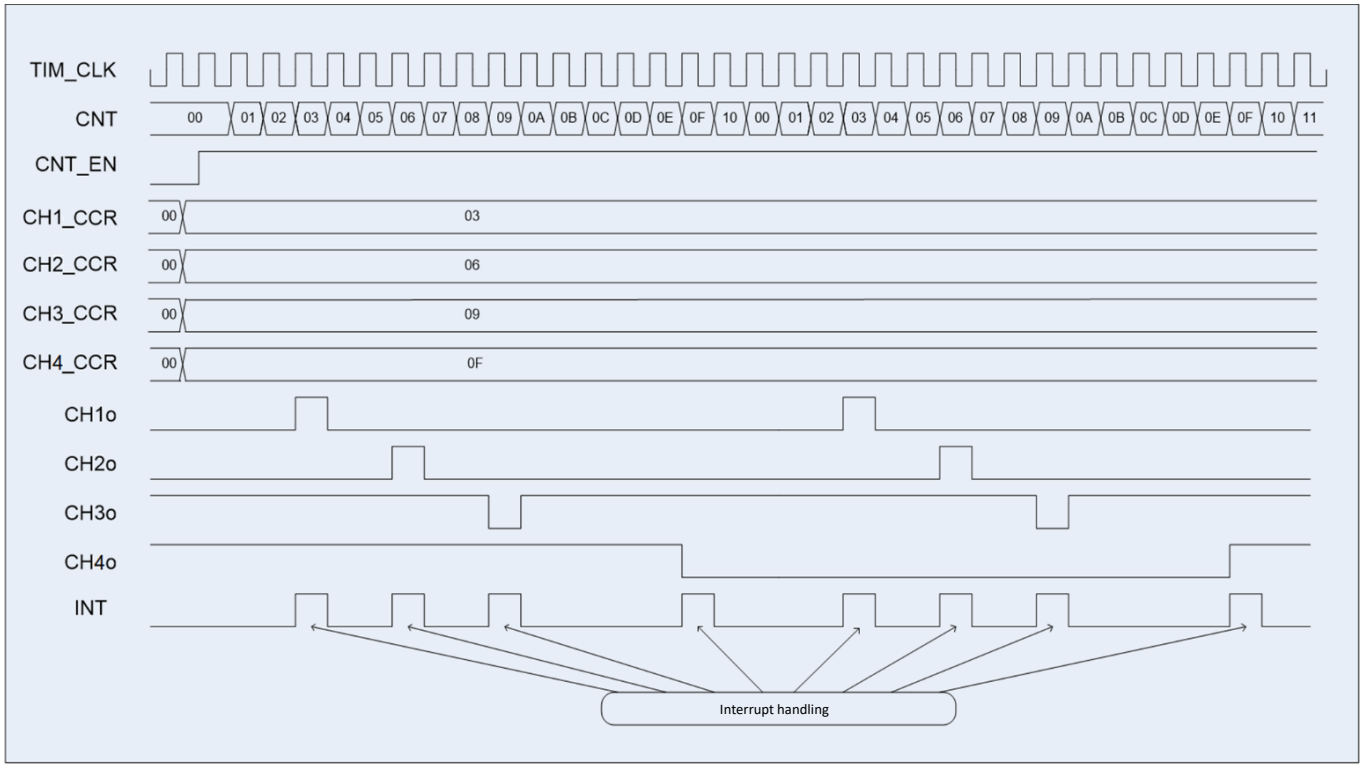


Figure 66 – Example timing diagrams in PWM mode

20.9. **Timer block register description**

Table 339 – Base addresses and offsets of timer control registers

Address	Name	Description
0x5007_0000	TIMER1	TIMER1 controller
0x5007_8000	TIMER2	TIMER2 controller
0x500D_8000	TIMER3	TIMER3 controller
0x500E_0000	TIMER4	TIMER4 controller
Offset		
0x00	CNT[31:0]	Main timer counter
0x04	PSG[31:0]	TIM_CLK prescaler for main counter clocking
0x08	ARR[31:0]	Main counter counting base
0x0C	CNTRL[31:0]	Main counter control register
0x50	BRKETR_CNTRL[31:0]	BRK and ETR input control register
0x54	STATUS[31:0]	Timer status register
0x58	IE[31:0]	Interrupt enable register
0x5C	DMA_RE[31:0]	DMA request enable register
Channel 1		
0x10	CH1_CCR[31:0]	Compare/capture register for timer channel 1
0x20	CH1_CNTRL0[31:0]	Control register 0 for timer channel 1
0x30	CH1_CNTRL1[31:0]	Control register 1 for timer channel 1
0x40	CH1_DTG[31:0]	DTG control register for timer channel 1
0x60	CH1_CNTRL2[31:0]	Control register 2 for timer channel 1
0x70	CH1_CCR1[31:0]	Compare/capture 1 register for timer channel 1
Channel 2		
0x14	CH2_CCR[31:0]	Compare/capture register for timer channel 2
0x24	CH2_CNTRL0[31:0]	Control register 0 for timer channel 2
0x34	CH2_CNTRL1[31:0]	Control register 1 for timer channel 2
0x44	CH2_DTG[31:0]	DTG control register for timer channel 2
0x64	CH2_CNTRL2[31:0]	Control register 2 for timer channel 2
0x74	CH2_CCR1[31:0]	Compare/capture 1 register for timer channel 2
Channel 3		
0x18	CH3_CCR[31:0]	Compare/capture register for timer channel 3
0x28	CH3_CNTRL0[31:0]	Control register 0 for timer channel 3
0x38	CH3_CNTRL1[31:0]	Control register 1 for timer channel 3
0x48	CH3_DTG[31:0]	DTG control register for timer channel 3
0x68	CH3_CNTRL2[31:0]	Control register 2 for timer channel 3
0x78	CH3_CCR1[31:0]	Compare/capture 1 register for timer channel 3
Channel 4		
0x1C	CH4_CCR[31:0]	Compare/capture register for timer channel 4
0x2C	CH4_CNTRL0[31:0]	Control register 0 for timer channel 4
0x3C	CH4_CNTRL1[31:0]	Control register 1 for timer channel 4
0x4C	CH4_DTG[31:0]	DTG control register for timer channel 4
0x6C	CH4_CNTRL2[31:0]	Control register 2 for timer channel 4
0x7C	CH4_CCR1[31:0]	Compare/capture 1 register for timer channel 4

20.9.1. **CNT**

Table 340 – Main timer counter CNT

Number	31…0
Access	R/W
Reset	0
	CNT[31:0]

Table 341 – Description of CNT register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…0	CNT[31:0]	Main timer counter value

20.9.2. **PSG**

Table 342 – TIM_CLK frequency divider for the PSG main counter count

Number	31..0
Access	R/W
Reset	0
	PSG[31:0]

Table 343 – Description of PSG register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…0	PSG[31:0]	Counter prescaler value The main counter operates at the frequency: TIM_CLKd = TIM_CLK/(PSG+1)

20.9.3. **ARR**

Table 344 – Main counter ARR counting base

Number	31…0
Access	R/W
Reset	0
	ARR[31:0]

Table 345 – Description of ARR register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…0	ARR[31:0]	Counting base for the main counter. CNT = [0…ARR]

20.9.4. **CNTRL**

Table 346 – Main counter control register CNTRL

Number	31…12	11…8
Access	U	R/W
Reset	0	0
	-	EVNT_SEL[3:0]

Number	7…6	5…9	3	2	1	0
Access	R/W	R/W	R/W	R/W	R/W	R/W
Reset	00	00	0	0	0	0
	CNT_MODE[1:0]	FDTS[1:0]	DIR	WR_CMPL	ARRB_EN	CNT_EN

Table 347 – Description of the CNTRL register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…12	-	Reserved.
11...8	EVNT_SEL[3:0]	Event source selection bits: 0000 – internal clock TIM_CLKd (formed by dividing TIM_CLK); 0001 – CNT == ARR in timer 1; 0010 – CNT == ARR in timer 2; 0011 – CNT == ARR in timer 3; 0100 – rising-edge event on channel 1, “Mode 1”; 0101 – rising-edge event on channel 2, “Mode 1”; 0110 – rising-edge event on channel 3, “Mode 1”; 0111 – rising-edge event on channel 4, “Mode 1”; 1000 – rising-edge event on ETR, “Mode 2”; 1001 – falling-edge event on ETR, “Mode 2”; 1010 – CNT == ARR in timer 4; 1011 – 1111 – reserved
7..6	CNT_MODE[1:0]	Main counter counting mode: 00 – up-counter when DIR = 0; down-counter when DIR = 1; 01 – bidirectional counter with automatic DIR update when CNT == 0 or CNT == ARR; 10 – up-counter when DIR = 0; down-counter when DIR = 1; 11 – reserved. The CNT_MODE[1:0] counting mode must be set according to the EVNT_SEL[3:0] field value: − EVNT_SEL[3:0] = 0000: CNT_MODE[1:0] = 00 or 01; − EVNT_SEL[3:0] != 0000: CNT_MODE[1:0] = 10
5...4	FDTS[1:0]	F_{DTS} clock divider: 00 – F_{DTS} = TIM_CLK; 01 – F_{DTS} = TIM_CLK/2; 10 – F_{DTS} = TIM_CLK/3; 11 – F_{DTS} = TIM_CLK/4

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
3	DIR	The main counter counting direction: 0 – up-counting, from 0 to ARR; 1 – down-counting, from ARR to 0
2	WR_CMPL	Write-in-progress flag for updating the CNT, PSG, and ARR registers: 0 – new data can be written; 1 – data not yet written, write in progress
1	ARRB_EN	ARR register update mode: 0 – ARR is updated at the moment ARR is written; 1 – ARR is updated when CNT == ARR
0	CNT_EN	Timer enable: 0 – timer disabled; 1 – timer enabled

20.9.5. **CHy_CCR**

Table 348 – Compare/capture register for channel “y” of timer CHy_CCR

Number	31…0
Access	R/W
Reset	0
	CCR[31:0]

Table 349 – Description of CHy_CCR register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…0	CCR[31:0]	Capture mode: CNT value at which the capture event occurred. PWM mode: CCR value that CNT is compared against

20.9.6. **CHy_CCR1**

Table 350 – Compare/capture register for channel “y” of timer CHy_CCR1

Number	31…0
Access	R/W
Reset	0
	CCR1[31:0]

Table 351 – Description of CHy_CCR1 register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…0	CCR1[31:0]	Capture mode: CNT value at which the capture event occurred. PWM mode: CCR1 value that CNT is compared against

20.9.7. **CHy_CNTRL0**

Table 352 – Control register for channel “y” of timer CHy_CNTRL0

Number	31…17	16	15	14	13
Access	U	RO	R/W	RO	R/W
Reset	0	0	0	0	0
	-	WR_CMPL1	CAP_NPWM	WR_CMPL	ETR_EN

Number	12	11…9	8	7…6	5…4	3…0
Access	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	000	0	00	00	0000
	BRK_EN	OCCM[2:0]	OCCE	CH_PSC[1:0]	CH_SEL[1:0]	CH_FLTR[3:0]

Table 353 – Description of the CHy_CNTRL0 register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...17	-	Reserved
16	WR_CMPL1	Write-in-progress flag for updating the CHy_CCR1 register: 0 – new data can be written; 1 – data not yet written, write in progress
15	CAP_NPWM	Channel operating mode: 0 – channel operates in PWM mode; 1 – channel operates in capture mode
14	WR_CMPL	Write-in-progress flag for updating the CHy_CCR register: 0 – new data can be written; 1 – data not yet written, write in progress
13	ETR_EN	Enable reset of the REF signal to “0” when a high level is present on the ETR input: 0 – disabled; 1 – enabled
12	BRK_EN	Enable reset of the REF and DTG signals to “0” when a low level is present on the BRK input: 0 – disabled; 1 – enabled
11…9	OCCM[2:0]	REF signal generation format in PWM mode: If CCR1_EN=0: 000 – always 0; 001 – 1 if CNT == CCR; 010 – 0 if CNT == CCR; 011 – toggle REF if CNT == CCR; 110 – always 0; 101 – always 1; 100 – 1 if DIR = 0 (up-counting) and CNT < CCR, otherwise 0; 0 if DIR = 1 (down-counting) and CNT > CCR, otherwise 1; 111 – 0 if DIR = 0 (up-counting) and CNT < CCR, otherwise 1; 1 if DIR = 1 (down-counting) and CNT > CCR, otherwise 0. If CCR1_EN=1: 000 – always 0; 001 – 1, if CNT==CCR or CNT==CCR1; 010 – 0 if CNT==CCR or CNT==CCR1; 011 – toggle REF if CNT == CCR or CNT == CCR1; 100 – always 0;

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
		101 – always 1; 110 – 0 if DIR = 0 (up-counting) and CCR ≤ CNT ≤ CCR1, otherwise 1; 0 if DIR = 1 (down-counting) and CCR < CNT < CCR1, otherwise 1; 111 – 1 if DIR = 0 (up-counting) and CCR ≤ CNT ≤ CCR1, otherwise 0; 1 if DIR = 1 (down-counting) and CCR < CNT < CCR1, otherwise 0. The condition CCR < CCR1 must be satisfied.
8	OCCE	ETR enable: 0 – disabled; 1 – enabled
7…6	CH_PSC[1:0]	Input channel prescaler: 00 – no division; 01 – /2; 10 – /4; 11 – /8
5…4	CH_SEL[1:0]	Selection of the event on the CHyi input channel used to latch the main counter value (CNT register) into the CHy_CCR register: 00 – positive edge on the CHyi input channel; 01 – negative edge on the CHyi input channel; 10 – positive edge from other channels: – for channel 1, from channel 2; – for channel 2, from channel 3; – for channel 3, from channel 4; – for channel 4, from channel 1; 11 – positive edge from other channels: – for channel 1, from channel 3; – for channel 2, from channel 4; – for channel 3, from channel 1; – for channel 4, from channel 2
3…0	CH_FLTR[3:0]	Filter configuration on channel “y” input. Selection of sampling frequency Fs and number of samples N: 0000 – no filtering, $F_S = F_{DTS}$; 0001 – F_S = TIM_CLK, N = 2; 0010 – F_S = TIM_CLK, N = 4; 0011 – F_S = TIM_CLK, N = 8; 0100 – $F_S = F_{DTS}/2$, N = 6; 0101 – $F_S = F_{DTS}/2$, N = 8; 0110 – $F_S = F_{DTS}/4$, N = 6; 0111 – $F_S = F_{DTS}/4$, N = 8; 1000 – $F_S = F_{DTS}/8$, N = 6; 1001 – $F_S = F_{DTS}/8$, N = 8; 1010 – $F_S = F_{DTS}/16$, N = 5; 1011 – $F_S = F_{DTS}/16$, N = 6; 1100 – $F_S = F_{DTS}/16$, N = 8; 1101 – $F_S = F_{DTS}/32$, N = 5; 1110 – $F_S = F_{DTS}/32$, N = 6; 1111 – $F_S = F_{DTS}/32$, N = 8

20.9.8. **CHy_CNTRL1**

Table 354 – Control register 1 for channel “y” of timer CHy_CNTRL1

Number	31…13	12	11…10	9…8	7…5	4	3…2	1…0
Access	U	R/W	R/W	R/W	U	R/W	R/W	R/W
Reset	0	0	00	00	0	0	00	00
	-	NINV	NSELO[1:0]	NSELOE[1:0]	-	INV	SELO[1:0]	SELOE[1:0]

Table 355 – Description of the CHy_CNTRL1 register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...13	-	Reserved.
12	NINV	Inversion of the inverted output nCHy: 0 – output is not inverted; 1 – output is inverted
11…10	NSELO[1:0]	Signal source selection for the inverted output nCHy: 00 – nCHyo outputs 0; 01 – nCHyo outputs 1; 10 – nCHyo outputs the nREF signal; 11 – nCHyo outputs the signal from DTG
9…8	NSELOE[1:0]	Operating mode of the inverted output nCHy: 00 – nCHyoe outputs 0; 01 – nCHyoe outputs 1; 10 – nCHyoe outputs the nREF signal; 11 – nCHyoe outputs the signal from the DTG block. When nCHyoe = 0, the channel pin is high impedance; when nCHyoe = 1, the channel pin operates as an output
7…5	-	Reserved
4	INV	Inversion of the direct output CHy: 0 – output is not inverted; 1 – output is inverted
3…2	SELO[1:0]	Signal source selection for the direct output CHy: 00 – CHyo outputs 0; 01 – CHyo outputs 1; 10 – CHyo outputs the REF signal; 11 – CHyo outputs the signal from the DTG block
1…0	SELOE[1:0]	Operating mode of the direct output CHy: 00 – CHyoe outputs 0; 01 – CHyoe outputs 1; 10 – CHyoe outputs the REF signal; 11 – CHyoe outputs the signal from the DTG block. When CHyoe = 0, the channel pin operates as an input; when CHyoe = 1, the channel pin operates as an output.

20.9.9. **CHy_CNTRL2**

Table 356 – Control register 2 for channel “y” of timer CHy_CNTRL2

Number	31…5	4	3	2	1…0
Access	U	R/W	R/W	R/W	R/W
Reset	0	0	0	0	00
	-	EVNT_DLY	CCR_RLD	CCR1_EN	CH_SEL1[1:0]

Table 357 – Description of CHy_CNTRL2 register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...5	-	Reserved
4	EVNT_DLY	Capture event delay before updating the CHy_CCR and CHy_CCR1 registers: 0 – the capture event signal is asserted when the event is detected, and the CHy_CCR and CHy_CCR1 registers are updated one TIM_CLK cycle later; 1 – the capture event signal is asserted simultaneously with updating the contents of the CHy_CCR and CHy_CCR1 registers
3	CCR_RLD	CHy_CCR and CHy_CCR1 register update mode: 0 – update may occur at any time; 1 – update is performed only when CNT == 0
2	CCR1_EN	CHy_CCR1 register enable: 0 – CHy_CCR1 is not used; 1 – CHy_CCR1 is used
1…0	CH_SEL1[1:0]	Selection of the event on the CHyi input channel used to latch the main counter value (CNT register) into the CHy_CCR1 register: 00 – positive edge on the CHyi input channel; 01 – negative edge on the CHyi input channel; 10 – negative edge from other channels: for channel 1, from channel 2; for channel 2, from channel 3; for channel 3, from channel 4; for channel 4, from channel 1; 11 – negative edge from other channels: for channel 1, from channel 3; for channel 2, from channel 4; for channel 3, from channel 1; for channel 4, from channel 2

20.9.10. **CHy_DTG**

Table 358 – Dead-time generator control register CHy_DTG

Number	31…16	15…8	7…5	4	3…0
Access	U	R/W	U	R/W	R/W
Reset	0	00000000	000	0	0000
	-	DTG[7:0]	-	EDTS	DTGx[3:0]

Table 359 – Description of CHy_DTG register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…16	-	Reserved.
15…8	DTG[7:0]	Main frequency divider DTG. Delay DTGdel = DTG · (DTGx + 1)
7…5	-	Reserved.
4	EDTS	DTG frequency: 0 – TIM_CLK; 1 – F_{DTS}
3…0	DTGx[3:0]	DTG prescaler

20.9.11 **BRKETR_CNTRL**

Table 360 – BRKETR_CNTRL register for BRK and ETR input control

Number	31…8	7…4	3…2	1	0
Access	U	R/W	R/W	R/W	R/W
Reset	0	0000	00	0	0
	-	ETR_FLTR[3:0]	ETR_PSC[1:0]	ETR_INV	BRK_INV

Table 361 – BRKETR_CNTRL register bit description

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...8	-	Reserved
7…4	ETR_FLTR[3:0]	Filter configuration on the ETR input. Selection of sampling frequency Fs and number of samples N: 0000 – no filtering, $F_S = F_{DTS}$; 0001 – F_S = TIM_CLK, N = 2; 0010 – F_S = TIM_CLK, N = 4; 0011 – F_S = TIM_CLK, N = 8; 0100 – $F_S = F_{DTS}/2$, N = 6; 0101 – $F_S = F_{DTS}/2$, N = 8; 0110 – $F_S = F_{DTS}/4$, N = 6; 0111 – $F_S = F_{DTS}/4$, N = 8; 1000 – $F_S = F_{DTS}/8$, N = 6; 1001 – $F_S = F_{DTS}/8$, N = 8; 1010 – $F_S = F_{DTS}/16$, N = 5; 1011 – $F_S = F_{DTS}/16$, N = 6; 1100 – $F_S = F_{DTS}/16$, N = 8; 1101 – $F_S = F_{DTS}/32$, N = 5; 1110 – $F_S = F_{DTS}/32$, N = 6; 1111 – $F_S = F_{DTS}/32$, N = 8
3…2	ETR_PSC[1:0]	Asynchronous prescaler for the ETR input clock: 00 – no division; 01 – /2; 10 – /4; 11 – /8

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
1	ETR_INV	ETR input inversion: 0 – not inverted; 1 – inverted
0	BRK_INV	BRK input inversion: 0 – not inverted; 1 – inverted

20.9.12 **STATUS**

Table 362 – STATUS timer status register

Number	31...17	16…13	12…9
Access	U	R/W	R/W
Reset	0	0	0
	-	CCR_CAP1_EVNT[3:0]	CCR_REF_EVNT[3:0]

Number	8…5	4	3
Access	R/W	R/W	R/W
Reset	0	0	0
	CCR_CAP_EVNT[3:0]	BRK_EVNT	ETR_FE_EVNT

Number	2	1	0
Access	R/W	R/W	R/W
Reset	0	0	0
	ETR_RE_EVNT	CNT_ARR_EVNT	CNT_ZERO_EVNT

Table 363 – Description of STATUS register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…17	-	Reserved
16…13	CCR_CAP1_EVNT[3:0]	Event: CNT counter value written to the CHy_CCR1 register on capture of the configured edge at the CHyi channel input: 0 – no event; 1 – event occurred. Cleared by writing “0”. If the write occurs simultaneously with a new event, the new event has priority. Bit 0 – channel 1; Bit 3 – channel 4

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
12…9	CCR_REF_EVNT[3:0]	Event: rising edge at the output of the REF reference signal generator: 0 – no event; 1 – event occurred. Cleared by writing “0”. If the write occurs simultaneously with a new event, the new event has priority. Bit 0 – channel 1; Bit 3 – channel 4
8…5	CCR_CAP_EVNT[3:0]	Event: CNT counter value written to the CHy_CCR register on capture of the configured edge at the CHyi channel input: 0 – no event; 1 – event occurred. Cleared by writing “0”. If the write occurs simultaneously with a new event, the new event has priority. Bit 0 – channel 1; Bit 3 – channel 4
4	BRK_EVNT	Event: high level on the BRK input: 0 – no event; 1 – event occurred. Cleared by writing “0”, provided a low level is present on the BRK input.
3	ETR_FE_EVNT	Event: falling edge on the ETR input: 0 – no event; 1 – event occurred. Cleared by writing “0”. If the write occurs simultaneously with a new event, the new event has priority
2	ETR_RE_EVNT	Event: rising edge on the ETR input: 0 – no event; 1 – event occurred. Cleared by writing “0”. If the write occurs simultaneously with a new event, the new event has priority
1	CNT_ARR_EVNT	Event: CNT equal to ARR: 0 – no event; 1 – event occurred. Cleared by writing “0”. If the write occurs simultaneously with a new match event, the new event has priority. If, from the moment of the match until the software clears the flag, the CNT and ARR registers do not change, the flag is not set again.
0	CNT_ZERO_EVNT	Event: CNT equal to zero: 0 – no event; 1 – event occurred. Cleared by writing “0”. If the write occurs simultaneously with a new match event, the new event has priority. If, from the moment of the match until the software clears the flag, the CNT register does not change, the flag is not set again.

20.9.13 **IE**

Table 364 – IE timer interrupt enable register

Number	31...17	16…13	12…9
Access	U	R/W	R/W
Reset	0	0	0
	-	CCR_CAP1_EVNT_IE[3:0]	CCR_REF_EVNT_IE[3:0]

Number	8…5	4	3
Access	R/W	R/W	R/W
Reset	0	0	0
	CCR_CAP_EVNT_IE[3:0]	BRK_EVNT_IE	ETR_FE_EVNT_IE

Number	2	1	0
Access	R/W	R/W	R/W
Reset	0	0	0
	ETR_RE_EVNT_IE	CNT_ARR_EVNT_IE	CNT_ZERO_EVNT_IE

Table 365 – Description of IE register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…17	-	Reserved
16...13	CCR_CAP1_EVNT_IE[3:0]	Interrupt enable flag for the CCR_CAP1_EVNT[3:0] event in the STATUS register: 0 – interrupt disabled; 1 – interrupt enabled. Bit 0 – channel 1; Bit 3 – channel 4
12…9	CCR_REF_EVNT_IE[3:0]	Interrupt enable flag for the CCR_REF_EVNT[3:0] event in the STATUS register: 0 – interrupt disabled; 1 – interrupt enabled. Bit 0 – channel 1; Bit 3 – channel 4
8…5	CCR_CAP_EVNT_IE[3:0]	Interrupt enable flag for the CCR_CAP_EVNT[3:0] event in the STATUS register: 0 – interrupt disabled; 1 – interrupt enabled. Bit 0 – channel 1; Bit 3 – channel 4

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
4	BRK_EVNT_IE	Interrupt enable flag for the BRK_EVNT event in the STATUS register: 0 – interrupt disabled; 1 – interrupt enabled.
3	ETR_FE_EVNT_IE	Interrupt enable flag for the ETR_FE_EVNT event in the STATUS register: 0 – interrupt disabled; 1 – interrupt enabled.
2	ETR_RE_EVNT_IE	Interrupt enable flag for the ETR_RE_EVNT event in the STATUS register: 0 – interrupt disabled; 1 – interrupt enabled.
1	CNT_ARR_EVNT_IE	Interrupt enable flag for the CNT_ARR_EVNT event in the STATUS register: 0 – interrupt disabled; 1 – interrupt enabled.
0	CNT_ZERO_EVNT_IE	Interrupt enable flag for the CNT_ZERO_EVNT event in the STATUS register: 0 – interrupt disabled; 1 – interrupt enabled.

20.9.14 **DMA_RE**

Table 366 – DMA_RE DMA request enable register

Number	31...17	16…13	12…9
Access	U	R/W	R/W
Reset	0	0	0
	-	CCR_CAP1_EVNT_RE[3:0]	CCR_REF_EVNT_RE[3:0]

Number	8…5	4	3
Access	R/W	R/W	R/W
Reset	0	0	0
	CCR_CAP_EVNT_RE[3:0]	BRK_EVNT_RE	ETR_FE_EVNT_RE

Number	2	1	0
Access	R/W	R/W	R/W
Reset	0	0	0
	ETR_RE_EVNT_RE	CNT_ARR_EVNT_RE	CNT_ZERO_EVNT_RE

Table 367 – Description of DMA_RE register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...17	-	Reserved
16…13	CCR_CAP1_EVNT_RE[3:0]	DMA request enable flag for the CCR_CAP1_EVNT[3:0] event: 0 – DMA request disabled; 1 – DMA request enabled. Bit 0 – channel 1; Bit 3 – channel 4
12…9	CCR_REF_EVNT_RE[3:0]	DMA request enable flag for the CCR_REF_EVNT[3:0] event: 0 – DMA request disabled; 1 – DMA request enabled. Bit 0 – channel 1; Bit 3 – channel 4
8…5	CCR_CAP_EVNT_RE [3:0]	DMA request enable flag for the CCR_CAP_EVNT[3:0] event: 0 – DMA request disabled; 1 – DMA request enabled. Bit 0 – channel 1; Bit 3 – channel 4
4	BRK_EVNT_RE	DMA request enable flag for the BRK_EVNT event: 0 – DMA request disabled; 1 – DMA request enabled.
3	ETR_FE_EVNT_RE	DMA request enable flag for the ETR_FE_EVNT event: 0 – DMA request disabled; 1 – DMA request enabled.
2	ETR_RE_EVNT_RE	DMA request enable flag for the ETR_RE_EVNT event: 0 – DMA request disabled; 1 – DMA request enabled.
1	CNT_ARR_EVNT_RE	DMA request enable flag for the CNT_ARR_EVNT event: 0 – DMA request disabled; 1 – DMA request enabled.
0	CNT_ZERO_EVNT_RE	DMA request enable flag for the CNT_ZERO_EVNT event: 0 – DMA request disabled; 1 – DMA request enabled.

21 SAR ADC controller

The integrated circuit includes a 10-bit successive-approximation ADC. It can digitize signals from three external analog pins and from two internal channels connected to the temperature sensor and the reference voltage source. The sampling rate is up to 500,000 conversions per second.

The ADC controller can:

- Digitize one of three external channels;
- Digitize the on-chip temperature sensor value;
- Digitize the on-chip reference voltage source value;
- Digitize the battery domain voltage;
- Perform automatic scanning of selected channels;
- Generate an interrupt when the converted value goes beyond specified limits.

A conversion requires at least 25 ADCS clock cycles. The conversion clock can be either the divided peripheral clock PCLKd or the ADC_CLK frequency generated in the "Clock signals" block. The clock source is selected by the Cfg_REG_CLKS bit. Inside the ADC controller, the clock frequency can be divided using the Cfg_REG_DIVCLK[3:0] bits. The C_ADCS frequency must not exceed 28 MHz.

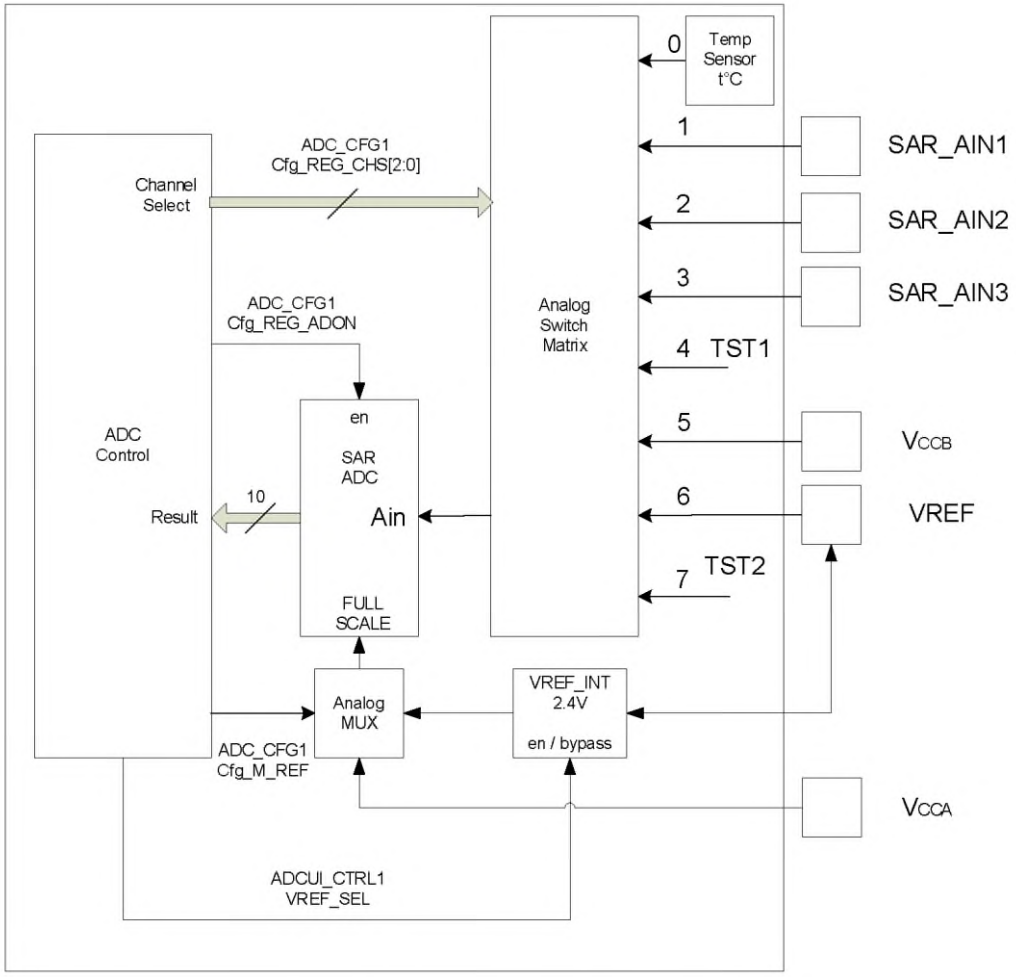


Figure 64 – ADC block diagram

To enable the ADC, the Cfg_REG_ADON bit must be set.

21.1. External channel conversion

In the ADC1_CFG register, the Cfg_REG_CHS[2:0] bits must be set to the channel number corresponding to the selected pin. Conversion can be performed using the internal reference U_{CCA} (Cfg_M_REF = 0) or an external reference (Cfg_M_REF = 1), in which case the reference is taken from the VREF pin. The Cfg_REG_CHCH, Cfg_REG_RNGC, and Cfg_REG_SAMPLE bits must be cleared.

To start a conversion, write 1 to the Cfg_REG_GO bit. After the conversion is complete, the Flg_REG_EOCIF bit in the ADC1_STATUS register is set. The conversion result is stored in the ADC1_RESULT register. After the result is read, the Flg_REG_EOCIF bit is cleared.

If, after the first conversion, the result is not read and a second conversion is performed, the ADC1_RESULT register will contain the value of the last conversion, and in addition to the Flg_REG_EOCIF bit, the Flg_REG_OVERWRITE bit will be set. The Flg_REG_OVERWRITE flag can be cleared only by writing to the ADC1_STATUS register.

21.2. **Sequential conversion of multiple channels**

For automatic sequential conversion of multiple channels or a single channel, set 1 in the bits of the ADC1_CHSEL register corresponding to the channels to be converted. These bits must be set before configuring the ADC, that is, before writing to the ADC1_CFG register. Conversion can be performed using the internal reference (Cfg_M_REF = 0) or an external reference (Cfg_M_REF = 1), in which case the reference is taken from the VREF pins. The Cfg_REG_RNGC bit must be cleared, and the Cfg_REG_SAMPLE and Cfg_REG_CHCH bits must be set. The DelayGO bits can be used to set a pause between conversions when cycling through the channels. With the ADC enabled (Cfg_REG_ADON = 1) and Cfg_REG_SAMPLE = 1, the conversion process starts automatically, and the Cfg_REG_GO bit is controlled by hardware.

After the conversion is complete, the Flg_REG_EOCIF bit in the ADC1_STATUS register is set. The conversion result is stored in the ADC1_RESULT register. After the result is read, the Flg_REG_EOCIF bit is cleared.

If, after the first conversion, the result is not read and a second conversion is performed, the ADC1_RESULT register will contain the value of the last conversion, and in addition to the Flg_REG_EOCIF bit, the Flg_REG_OVERWRITE bit will be set. The Flg_REG_OVERWRITE flag can be cleared only by writing to the ADC1_STATUS register.

For sequential conversion of the same channel, you can either select only one channel in the ADC1_CHSEL register and set the Cfg_REG_CHCH bit to 1, or set the channel number in the Cfg_REG_CHS[2:0] bits and clear the Cfg_REG_CHCH bit to 0. In this case, the sequential conversion process is performed only for that channel. Sequential conversion of the temperature sensor value and the reference voltage source can be performed only in single-channel sequential conversion mode.

21.3. **Conversion with range checking**

If it is necessary to ensure that digitized values stay within an allowed range, it is possible to set the lower and upper limits in the ADC1_L_LEVEL and ADC1_H_LEVEL registers. If the Cfg_REG_RNGC bit is set and a conversion result goes beyond these limits, the Flg_REG_AWOIFEN flag is set. The result register will hold the conversion result.

The conversion result is calculated using the formula

$$\text{input voltage at the ADC} = \frac{\text{ADC full-scale voltage} \cdot \textit{ADC1\_RESULT} \text{ register value.}}{1024}. \quad (18)$$

21.4. **Temperature sensor**

The ADC can be used to convert the temperature sensor signal. To select the temperature sensor as the conversion source, set channel 0 in the Cfg_REG_CHS bits. After that, you can start the conversion process. To start a conversion, write 1 to the Cfg_REG_GO bit. After the conversion is complete, the Flg_REG_EOCIF bit in the ADC1_STATUS register is set. The conversion result is stored in the ADC1_RESULT register. After the result is read, the Flg_REG_EOCIF bit is cleared.

If, after the first conversion, the result is not read and a second conversion is performed, the ADC1_RESULT register will contain the value of the last conversion, and in addition to the Flg_REG_EOCIF bit, the Flg_REG_OVERWRITE bit will be set. The Flg_REG_OVERWRITE flag can be cleared only by writing to the ADC1_STATUS register.

For sequential conversion of only the temperature sensor, you can either select only channel 0 in the ADC1_CHSEL register and set the Cfg_REG_CHCH bit to 1, or set channel 0 in the Cfg_REG_CHS[2:0] bits and clear the Cfg_REG_CHCH bit to 0. In this case, the sequential conversion process is performed only for that channel.

The temperature sensor parameters are not specified. Depending on the required accuracy, calibration at two or three points may be sufficient. For higher-accuracy measurements, a calibration table must be created. Calibration is performed individually for each integrated circuit.

21.4.1. **Temperature calculation formula**

The die temperature is calculated using the formula

$$TEMP = 25 - \frac{\frac{\text{ADC full-scale voltage} \cdot ADC1\_RESULT \text{ register value.}}{1024} - 1.403}{0.0037}. \quad (19)$$

For example, when reading channel 0, the ADC outputs code = 625, and the full-scale voltage is an external 2.5 V source connected to the VREF pin of the integrated circuit. Then the die temperature is

$$TEMP = 25 - \frac{\frac{2.5 \cdot 625}{1024} - 1.403}{0.0037} = -8.2\ ^{\circ}C. \quad (20)$$

21.5. **Internal capacitance charge time**

The conversion process consists of two stages: first, the internal capacitance is charged to the level of the external signal, and then the charge level of the internal capacitance is converted to a digital code. For accurate conversion of the external signal into a digital code, during the first stage the internal capacitance must charge up to the external signal level. This time is determined by the nominal value of the internal capacitance, the input resistance of the ADC path, and the output resistance of the signal source. The formula below is used to determine the maximum allowable source output resistance R_{AIN} to ensure the required conversion accuracy.

$$R_{AIN} < \frac{T_{track}}{C_{ADC} \cdot \ln(2^N)} - R_{ADC}\,, \tag{21}$$

Where C_{ADC} – ADC internal capacitance (~2–3 pF);
N – required accuracy in bits;
R_{ADC} – input resistance of the ADC path (~1500 Ω);
T_{track} – internal capacitance charge time, in clock cycles, defined by the formula

$$T_{track} = 4 \cdot T_{C\_ADCS} + N_{PCLKd} \cdot T_{PCLKd} = \frac{4}{f_{C\_ADCS}} + \frac{(ConvDelay)}{f_{PCLKd}}, \tag{22}$$

Where ConvDelay – additional delay before the start of conversion, determined by the DelayGo[2:0] bits according to Tables 381–383;
f_{C_ADCS} – SAR ADC operating frequency;
f_{PCLKd} – derived from the formula

$$f_{PCLKd} = \frac{f_{PCLK}}{2^{Cfg\ REG\ DIVCLK}}. \tag{23}$$

For 10-bit conversion accuracy of ±1 LSB, N = 10. The charge time T_{track} is determined by the DelayGo[2:0] bits and the ADC internal circuitry and is given in Tables 381–383. The charge time T_{track} must be at least 1 μs.

21.6. **ADC internal capacitance charge time and conversion time**

Table 368 – ADC internal capacitance charge time and conversion time. C_ADCS = ADC_CLK/PCLKd (Cfg_REG_CLKS = 1/0), ADC_CLK = PCLKd = PCLK

DelayGo[2:0]	Additional delay before the start of conversion	Total ADC capacitance charge time Ttrack before the start of conversion	Total ADC conversion time
000	1xPCLKd	4xC_ADCS+1xPCLKd	25xC_ADCS+1xPCLKd
001	3xPCLKd	4xC_ADCS+3xPCLKd	25xC_ADCS+3xPCLKd
010	3xPCLKd	4xC_ADCS+3xPCLKd	25xC_ADCS+3xPCLKd
011	5xPCLKd	4xC_ADCS+5xPCLKd	25xC_ADCS+5xPCLKd
100	5xPCLKd	4xC_ADCS+5xPCLKd	25xC_ADCS+5xPCLKd
101	7xPCLKd	4xC_ADCS+7xPCLKd	25xC_ADCS+7xPCLKd
110	7xPCLKd	4xC_ADCS+7xPCLKd	25xC_ADCS+7xPCLKd
111	9xPCLKd	4xC_ADCS+9xPCLKd	25xC_ADCS+9xPCLKd

Table 369 – ADC internal capacitance charge time and conversion time. C_ADCS = PCLKd (Cfg_REG_CLKS = 0)

DelayGo[2:0]	Additional delay before the start of conversion	Total ADC capacitance charge time Ttrack before the start of conversion	Total ADC conversion time
000	0xPCLKd	4xC_ADCS+0xPCLKd	25,5xC_ADCS+2xPCLK+0xPCLKd
001	0xPCLKd	4xC_ADCS+0xPCLKd	25,5xC_ADCS+2xPCLK+0xPCLKd
010	2xPCLKd	4xC_ADCS+2xPCLKd	25,5xC_ADCS+2xPCLK+2xPCLKd
011	2xPCLKd	4xC_ADCS+2xPCLKd	25,5xC_ADCS+2xPCLK+2xPCLKd
100	4xPCLKd	4xC_ADCS+4xPCLKd	25,5xC_ADCS+2xPCLK+4xPCLKd
101	4xPCLKd	4xC_ADCS+4xPCLKd	25,5xC_ADCS+2xPCLK+4xPCLKd
110	6xPCLKd	4xC_ADCS+6xPCLKd	25,5xC_ADCS+2xPCLK+6xPCLKd
111	6xPCLKd	4xC_ADCS+6xPCLKd	25,5xC_ADCS+2xPCLK+6xPCLKd

Table 370 – ADC internal capacitance charge time and conversion time. C_ADCS = ADC_CLK (Cfg_REG_CLKS = 1)

DelayGo[2:0]	Additional delay before the start of conversion	Total ADC capacitance charge time Ttrack before the start of conversion	Total ADC conversion time
000	1xPCLKd	4xC_ADCS+1xPCLKd	25xC_ADCS+1,5xPCLK+1xPCLKd
001	2xPCLKd	4xC_ADCS+2xPCLKd	25xC_ADCS+1,5xPCLK+2xPCLKd
010	3xPCLKd	4xC_ADCS+3xPCLKd	25xC_ADCS+1,5xPCLK+3xPCLKd
011	4xPCLKd	4xC_ADCS+4xPCLKd	25xC_ADCS+1,5xPCLK+4xPCLKd
100	5xPCLKd	4xC_ADCS+5xPCLKd	25xC_ADCS+1,5xPCLK+5xPCLKd
101	6xPCLKd	4xC_ADCS+6xPCLKd	25xC_ADCS+1,5xPCLK+6xPCLKd
110	7xPCLKd	4xC_ADCS+7xPCLKd	25xC_ADCS+1,5xPCLK+7xPCLKd
111	8xPCLKd	4xC_ADCS+8xPCLKd	25xC_ADCS+1,5xPCLK+8xPCLKd

In addition to accuracy determined by the ADC internal capacitance charge time, conversion accuracy is affected by errors arising from process variations and noise, characterized by the parameters E_{DLADC}, E_{ILADC} and E_{OFFADC}.

To correctly set ADC operating modes, the ADC1_CFG register must be configured before setting the Cfg_REG_GO bit; otherwise, the new configuration takes effect only starting from the next conversion.

21.7. **Description of the ADC controller block registers**

Table 371 – Description of the ADC controller block registers

Base Address	Name	Description
0x5004_0000	ADC	ADC controller
Offset		
0x00	ADC1_CFG	ADC control register
0x08	ADC1_H_LEVEL	ADC high limit register
0x10	ADC1_L_LEVEL	ADC low limit register
0x18	ADC1_RESULT	ADC result register

Base Address	Name	Description
0x5004_0000	ADC	ADC controller
Offset		
0x20	ADC1_STATUS	ADC status register
0x28	ADC1_CHSEL	ADC channel selection register
0x30	ADC_TRIM	ION configuration register

21.7.1. **ADC1_CFG**

Table 372 – ADC1_CFG Register

Number	31…28	27…25	24…19
Access	U	R/W	U
Reset	0	0	0
	-	DelayGo[2:0]	-

Number	18..16	15…12	11	10	9
Access	U	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0
	-	Cfg_REG_DIVCLK [3:0]	Cfg_M_ REF	Cfg_REG_RNGC	Cfg_REG_CHC H

Number	8, 7	6…4	3	2	1	0
Access	U	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0
	-	Cfg_REG_ CHS[2:0]	Cfg_REG_ SAMPLE	Cfg_REG_ CLKS	Cfg_REG_ GO	Cfg_REG_ ADON

Table 373 – Description of the ADC1_CFG register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...28	-	Reserved
27…25	DelayGo[2:0]	Delay before starting the next conversion after the previous one completes in sequential channel scan mode; see Tables 368–370
24...19	-	Must be 0 for correct operation
18…16	-	Must be 010 for correct operation
15...12	Cfg_REG_DIVCLK[3:0]	Peripheral clock division factor selection: 0000 – PCLKd = PCLK; 0001 – PCLKd = PCLK/2; 0010 – PCLKd = PCLK/4; 0011 – PCLKd = PCLK/8; … 1011 – PCLKd = PCLK/2048. Other values: PCLKd = PCLK

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
11	Cfg_M_REF	ADC scale selection: 0 – ADC scale from 0 to U_{CCA}; 1 – ADC scale from 0 to VREF
10	Cfg_REG_RNGC	Automatic level monitoring enable: 1 – enabled: interrupt generated when the result goes outside the range defined in the limit registers; 0 – disabled
9	Cfg_REG_CHCH	Channel switching mode: 1 – switching enabled (channels selected in the Sl_Ch_Ch_REF channel selection register are scanned); 0 – only the selected channel is used
8, 7	-	Must be 0 for correct operation
6…4	Cfg_REG_CHS[2:0]	Analog input channel selection for conversion: 111 – channel 0 (temperature sensor); 110 – channel 1 (connected to SAR_AIN1); 101 – channel 2 (connected to SAR_AIN2); 100 – channel 3 (connected to SAR_AIN3); 011 – channel 4 (test mode); 010 – channel 5 (connected to VDD_BD); 001 – channel 6 (connected to VREF); 000 – channel 7 (test mode)
3	Cfg_REG_SAMPLE	ADC start mode selection: 1 – sequential: automatic start after completion of the previous conversion; 0 – single conversion
2	Cfg_REG_CLKS	ADC clock source selection: 1 – ACLK (generated from the selected clock source; see section “Clock signals”); 0 – PCLKd (derived from the core clock; see formula (23))
1	Cfg_REG_GO	Start of conversion: Writing “1” starts the conversion process; the bit is cleared automatically. A write while Cfg_REG_SAMPLE = 1 does not affect ADC operation; the bit is set and cleared automatically.
0	Cfg_REG_ADON	ADC enable: 1 – enabled; 0 – disabled

21.7.2. **ADC1_H_LEVEL**

Table 374 – ADC1_H_LEVEL Register

Number	31…10	9…0
Access	U	R/W
Reset	0	0
	-	REG_H_LEVEL[9:0]

Table 375 – Description of the ADC1_H_LEVEL register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…10	-	Reserved
9…0	REG_H_LEVEL[9:0]	Upper limit of the allowed range

21.7.3. **ADC1_L_LEVEL**

Table 376 – ADC1_L_LEVEL Register

Number	31…10	9…0
Access	U	R/W
Reset	0	0
		REG_L_LEVEL[9:0]

Table 377 – Description of the ADC1_L_LEVEL register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31..10	-	Reserved
9…0	REG_L_LEVEL[9:0]	Lower limit of the allowed range

21.7.4. **ADC1_RESULT**

Table 378 – ADC1_RESULT Register

Number	31…19	18…16	15…10	9…0
Access	U	RO	U	RO
Reset	0	0	0	0
	-	CHANNEL[2:0]	-	RESULT[9:0]

Table 379 – Description of the ADC1_RESULT register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...19	-	Reserved

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
18...16	CHANNEL[2:0]	Conversion result channel: 111 – channel 0 (temperature sensor); 110 – channel 1 (connected to SAR_AIN1); 101 – channel 2 (connected to SAR_AIN2); 100 – channel 3 (connected to SAR_AIN3); 011 – channel 4 (test mode); 010 – channel 5 (connected to VDD_BD); 001 – channel 6 (connected to VREF); 000 – channel 7 (test mode)
15...10	-	Reserved
9…0	RESULT[9:0]	Conversion result value

21.7.5. **ADC1_STATUS**

Table 380 – ADC1_STATUS Register

Number	31…5	4	3	2	1	0
Access	U	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0
	-	ECOIF_IE	AWOIF_IE	Flg_REG_ EOCIF	Flg_REG_ AWOIFEN	Flg_REG_OVERWRITE

Table 381 – Description of the ADC1_STATUS register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…5	-	Reserved
4	ECOIF_IE	Interrupt enable flag for the Flg_REG_ECOIF event: 0 – interrupt not generated; 1 – interrupt generated
3	AWOIF_IE	Interrupt enable flag for the Flg_REG_AWOIFEN event: 0 – interrupt not generated; 1 – interrupt generated
2	Flg_REG_EOCIF	The flag is set when conversion is complete and the data has not yet been read. Cleared by reading the result from the ADC1_RESULT register: 1 – conversion result is ready; 0 – no result
1	Flg_REG_AWOIFEN	The flag is set when the conversion result is higher than the upper limit or lower than the lower limit of the automatic level monitoring range. Cleared only by writing zero to this bit. 0 – result within the allowed range; 1 – result outside the allowed range
0	Flg_REG_OVERWRITE	Indicates that the data in the result register has been overwritten. Cleared only by writing zero to this bit. 0 – no overwrite of an unread result has occurred; 1 – an unread conversion result

21.7.6. **ADC1_CHSEL**

Table 382 – ADC1_CHSEL Register

Number	31…8	7…0
Access	U	R/W
Reset	0	0
	-	Sl_Ch_Ch_REF[7:0]

Table 383 – Description of the ADC1_CHSEL register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…8	-	Must be 0 for correct operation
7...0	Sl_Ch_Ch_REF[7:0]	Selection of channels for automatic scan: 0 – corresponding channel is not included in the scan; 1 – corresponding channel is included in the scan. The channels participating in the scan correspond to the bit positions set in this register.

21.7.7. **ADC_TRIM**

Table 384 – ADC_TRIM Register

Number	31..6	5…1	0
Access	U	R/W	U
Reset	0	10000	0
	-	BG_TRIM[4:0]	-

Table 385 – Description of the ADC_TRIM register bits

Bit positions	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…6	-	Reserved
5…1	BG_TRIM[4:0]	Bandgap reference voltage trim. It is recommended to use the TRIM_BG bit values programmed into Flash memory during production.
0	-	Reserved

21.8. **SAR_AINx pins**

Special-purpose SAR_AINx pins are implemented in the IC; unlike other pins, they do not have protection diodes to any power supply. Voltages higher than the supply voltage are allowed on these pins, but must not exceed the values specified in Table 594.

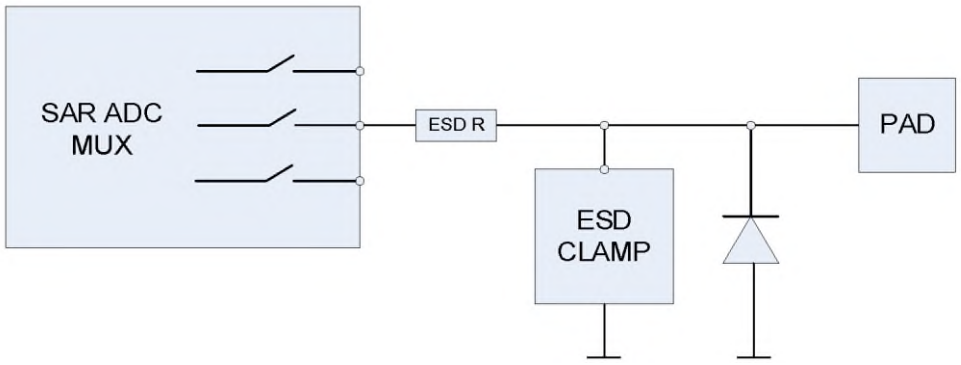


Figure 65 – SAR_AINx pin block diagram

22 I2C interface controller

The I2C interface is a two-wire, bidirectional serial communication channel that provides a simple and efficient method of data exchange between devices. The I2C interface controller in the IC operates only in master mode. Compatibility with multi-master systems is supported: collision detection and bus arbitration mechanisms are implemented to prevent data loss when two or more masters attempt to transmit data simultaneously. Operation with slave devices that use SCL clock stretching to slow down the data transfer rate is supported.

22.1. I2C controller configuration

The I2C controller generates the SCL clock signal using an internal prescaler DIV, whose division factor is set in the PRL (lower) and PRH (upper) registers. The minimum DIV value is 1. If the calculated DIV value does not require the most significant part, the PRH register must be set to 0. DIV must be configured only when the I2C controller is disabled (EN_I2C = 0).

The estimated I2C bus baud rate is defined by the following formula:

$$F_{scl} = \frac{\text{HCLK}}{5 \cdot (\text{DIV} + 1)} . \quad (24)$$

The I2C controller monitors the actual state of the SCL line. If, after completing the low level, the controller releases SCL but the line remains at logic low (for example, due to SCL stretching by a slave device or a slow rising edge on SCL), the start of the next clock cycle is delayed until SCL transitions to logic high. As a result, the actual baud rate may be lower than the calculated value given by formula (24).

The I2C controller is enabled by setting the EN_I2C bit in the CMD register. The I2C controller may be disabled (clearing EN_I2C) only after the bus transaction is complete, that is, after the STOP condition has been transmitted. Violation of this requirement may lead to I2C bus lock-up.

Writes to the CMD control register are allowed only when the I2C controller is enabled (EN_I2C = 1). Any attempt to write to the CMD register while the I2C controller is inactive (EN_I2C = 0) is ignored.

22.2. **System configuration**

I2C systems use a serial data line SDA and a clock line SCL. All devices connected to these two lines operate in open-drain mode, forming a wired-AND on the line via external pull-up resistors from both lines to the supply.

Data transfer between the master and the slave is carried over SDA and synchronized by SCL. After the data transfer phase is completed, a single acknowledge bit is transferred in the opposite direction. Each received bit is latched by the receiving side when SCL is high and may be changed by the transmitter when SCL is low. A transition on SDA while SCL is high is interpreted as a control condition (see “START signal” and “STOP signal”).

22.3. **I2C protocol**

A typical I2C transfer consists of four phases:

– START signal;
– Address transfer;
– Data transfer;
– STOP signal.

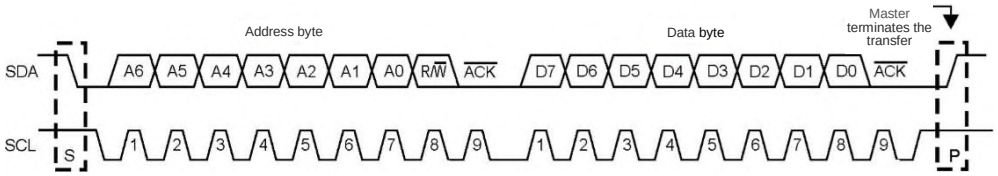


Figure 66 – I2C receive/transmit in 7-bit mode

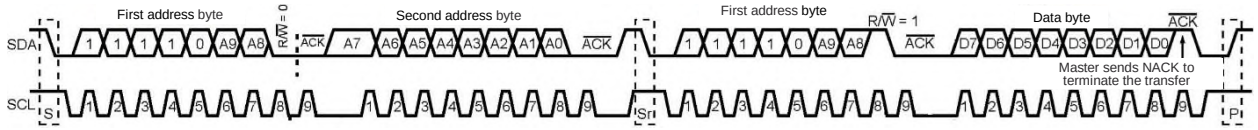


Figure 67 – I2C receive in 10-bit mode

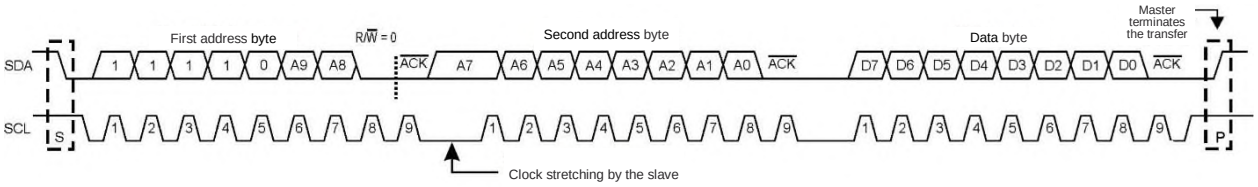


Figure 68 – I2C transmit in 10-bit mode

22.3.1. **START signal**

When the bus is idle, that is, no device is transmitting (SCL and SDA are both high), the master can initiate a transfer by generating a START signal on the line. The START signal or S bit is defined as a high-to-low transition on SDA while SCL is high. The presence of a START signal does not yet mark the beginning of the data transfer.

A repeated START signal is a regular START signal that is not preceded by a STOP signal. This method allows the master to start communication with another slave, or with the same slave in a different operating mode (for example, read after write or vice versa), without returning the bus to the idle state.

The interface controller generates a START signal when a 1 is written to the START bit in the CMD register with the WR or RD bit set. To initiate a standard transfer, the START and WR bits in the CMD register should be set, which causes generation of a START condition on the line and then transmission of the slave address and the RW bit stored in the TXD register. Depending on the current state of SCL, either a START or a repeated START signal is generated

22.3.2. **Address transfer**

The first data byte transmitted by the master immediately after the START signal is the slave address. This is a 7-bit address followed by the RW bit. The RW bit defines the subsequent data transfer direction: 0 – write data to the slave device; 1 – read data from the slave device. A single bus cannot contain multiple slave devices with the same address. The slave device whose address matches the address in the message acknowledges reception by asserting ACK and pulling SDA low during the 9th SCL clock pulse.

Transmission of the 7-bit address is performed together with the START signal. The slave address and the RW bit should be written to the TXD register, after which the START and WR bits in the CMD register are set. The controller then transmits the START condition and the address onto the line.

The controller also supports transmission of a 10-bit slave address by generating multiple address transfer cycles (see Figures 67 and 68). First, a START signal and the first address byte with RW = 0 (write to the slave device) are transmitted as described above. After this transfer is complete, the second address byte is written to the TXD register, and then the WR bit in the CMD register is set. If the transfer direction must be changed to RW = 1 (read from the slave device), then, after the second address byte is transmitted, a repeated START condition and the first address byte with RW = 1 are transmitted.

22.3.3. **Data transfer**

After the address has been successfully acknowledged by a slave device, the data transfer can be started in the direction defined by the RW bit in the master frame. Each transmitted byte is acknowledged by an ACK on the 9th SCL clock pulse. If the slave device issues NACK (no acknowledge), the master may generate either a STOP signal to terminate the transfer or a repeated START signal to begin a new transfer cycle.

If the master is the receiving device and issues NACK, the slave releases SDA and the master may generate a STOP or a repeated START signal.

To write data to a slave device, data is written to the TXD register and the WR bit in the CMD register is set. To read data from a device, the RD bit in the CMD register is set, while simultaneously writing the required value to the ACK bit. During a transfer, the interface controller sets the TR_PROG flag in the STA register. When the transfer is complete, this flag is cleared and the INT flag is set. If at the same time the EN_INT enable bit in the CTR register is set, an interrupt is generated to the interrupt controller. The RXD register contains valid received data after TR_PROG is cleared or the INT flag is set. A new read or write cycle can be started only when the TR_PROG flag is cleared.

22.3.4. **STOP signal**

The master terminates the connection by generating a STOP signal. The STOP signal, or P bit, is defined as a low-to-high transition on SDA while SCL is high.

The STOP signal is generated by writing 1 to the STOP bit in the CMD register.

22.4. **Description of I2C controller registers**

Table 386 – Description of I2C controller registers

Base Address	Name	Description
0x5003_0000	I2C	I2C controller
Offset		
0x00	PRL	Least significant part of the frequency prescaler
0x04	PRH	Upper part of the frequency prescaler
0x08	CTR	I2C controller control
0x0C	RXD	Data received over I2C
0x10	STA	I2C status
0x14	TXD	Data transmitted over I2C
0x18	CMD	I2C control

22.4.1. **PRL**

Table 387 – PRL Register

Number	31…8	7…0
Access	U	R/W
Reset	0	0xFF
	-	PR[7:0]

Table 388 – Description of PRL register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…8	-	Reserved
7…0	PR[7:0]	Least significant part of the prescaler

22.4.2. **PRH**

Table 389 – PRH Register

Number	31…8	7…0
Access	U	R/W
Reset	0	0xFF
	-	PR[15:8]

Table 390 – Description of PRH register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...8	-	Reserved
7…0	PR[15:8]	Upper part of the prescaler

22.4.3. **CTR**

Table 391 – CTR Register

Number	31…8	7	6	5…0
Access	U	R/W	R/W	U
Reset	0	0	0	0
	-	EN_I2C	EN_INT	-

Table 392 – Description of CTR register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...8	-	Reserved
7	EN_I2C	I2C controller enable: 0 – disabled; 1 – enabled
6	EN_INT	I2C interrupt enable: 0 – disabled; 1 – enabled
5…0	-	Reserved

22.4.4. **RXD**

Table 393 – RXD Register

Number	31…8	7…0
Access	U	R/O
Reset	0	0
	-	RXD[7:0]

Table 394 – Description of RXD register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...8	-	Reserved
7…0	RXD[7:0]	Last byte received over I2C

22.4.5. **STA**

Table 395 – STA Register

Number	31…8	7	6	5	4…2	1	0
Access	U	R/O	R/O	R/O	U	R/O	R/O
Reset	0	0	0	0	0	0	0
	-	RX_ACK	BUSY	LOST_ARB	-	TR_PROG	INT

Table 396 – Description of STA register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…8	-	Reserved
7	RX_ACK	ACK/NACK received from the slave: 0 – ACK received; 1 – NACK received. This bit is cleared to 0 when arbitration is lost.
6	BUSY	I2C bus state: 0 – after a STOP signal has been received; 1 – after a START signal has been detected
5	LOST_ARB	Arbitration lost: 0 – no arbitration loss; 1 – arbitration lost. This bit is set if:
		– a STOP signal is detected that was not initiated by this controller; – the controller attempts to drive SDA high, but SDA remains low. The bit is cleared to 0 after a START signal is detected.
4…2	-	Reserved

1	TR_PROG	Transfer in progress: 0 – transfer complete; 1 – data transfer in progress. This bit reflects the combination of the RD and WR bits of the CMD register.
0	INT	Interrupt flag. 0 – no interrupt; 1 – interrupt. The flag is set if: – a byte transfer has completed; – arbitration has been lost. An interrupt to the processor is generated if the EN_INT bit is set.

22.4.6. **TXD**

Table 397 – TXD Register

Number	31…8	7…0
Access	U	R/W
Reset	0	0
	-	TXD[7:0]

Table 398 – Description of TXD register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...8	-	Reserved
7…0	TXD[7:0]	Byte for transmission over I2C. When transmitting the first address byte, bit 0 defines the transfer mode: 0 – write to the slave device; 1 – read from the slave device

22.4.7. **CMD**

Table 399 – CMD Register

Number	31…8	7	6	5	4	3	2, 1	0
Access	U	R/W	R/W	R/W	R/W	R/W	U	R/W
Reset	0	0	0	0	0	0	0	0
	-	START	STOP	RD	WR	ACK	-	CLR_INT

Table 400 – Description of CMD register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...8	-	Reserved
7	START	Send START signal. 0 – no action; 1 – start sending a START signal. A START signal is transferred only if the WR or RD bit is set. Cleared automatically when arbitration is lost and after the transfer completes.
6	STOP	Send STOP signal. 0 – no action; 1 – start sending a STOP signal. Cleared automatically when arbitration is lost and after the transfer completes.
5	RD	Read from slave device: 0 – no action; 1 – start a read operation. Cleared automatically when arbitration is lost and after the transfer completes.
4	WR	Write to slave device: 0 – no action; 1 – start a write operation. Cleared automatically when arbitration is lost and after the transfer completes.
3	ACK	Send ACK: 0 – send ACK; 1 – send NACK. After a NACK is sent, this bit is not cleared automatically; it is cleared by writing 0.
2, 1	-	Reserved
0	CLR_INT	Clear INT interrupt flag. Writing 1 clears the interrupt.

23 SSP controller

The Synchronous Serial Port (SSP) module implements a synchronous serial communication interface in both master and slave modes and provides data exchange with a connected slave or master peripheral device according to one of the following protocols:

- SPI interface (Motorola);
- SSI interface (Texas Instruments);
- Microwire interface (National Semiconductor).

In both master and slave modes, the SSP module provides:

- parallel-to-serial conversion of data placed in the transmitter FIFO buffer (eight 32-bit data locations);
- serial-to-parallel conversion of data and its storage in the receive FIFO buffer (eight 32-bit data locations).

The module generates interrupt requests on the following events:

- service required for the receiver or transmitter FIFO buffers;
- receiver FIFO overrun;
- data present in the receiver FIFO buffer after a timeout expires;
- data present in the receiver FIFO;
- no data in the transmitter FIFO;
- no data in the transmitter shift register.

General information about the module is provided in the following subsections:

- SPI interface characteristics;
- Microwire interface characteristics;
- SSI interface characteristics.

23.1. Key characteristics of the SSP module

- Operates in both master and slave modes;
- Software-controlled data exchange rate;
- Consists of independent transmit and receive FIFO (First-In First-Out) buffers (eight 32-bit entries each);
- Software selection of one of the serial interfaces: SPI, Microwire, SSI;
- Programmable data frame length from 4 to 32 bits;
- Independent interrupt masking for transmit FIFO buffer, receive FIFO buffer, receive buffer overrun, data present in the receive FIFO, transmit FIFO empty, and transmit shift register empty;
- Loopback test capability by connecting input to output;
- Support for Direct Memory Access (DMA).

The block diagram of the module is shown in Figure 69.

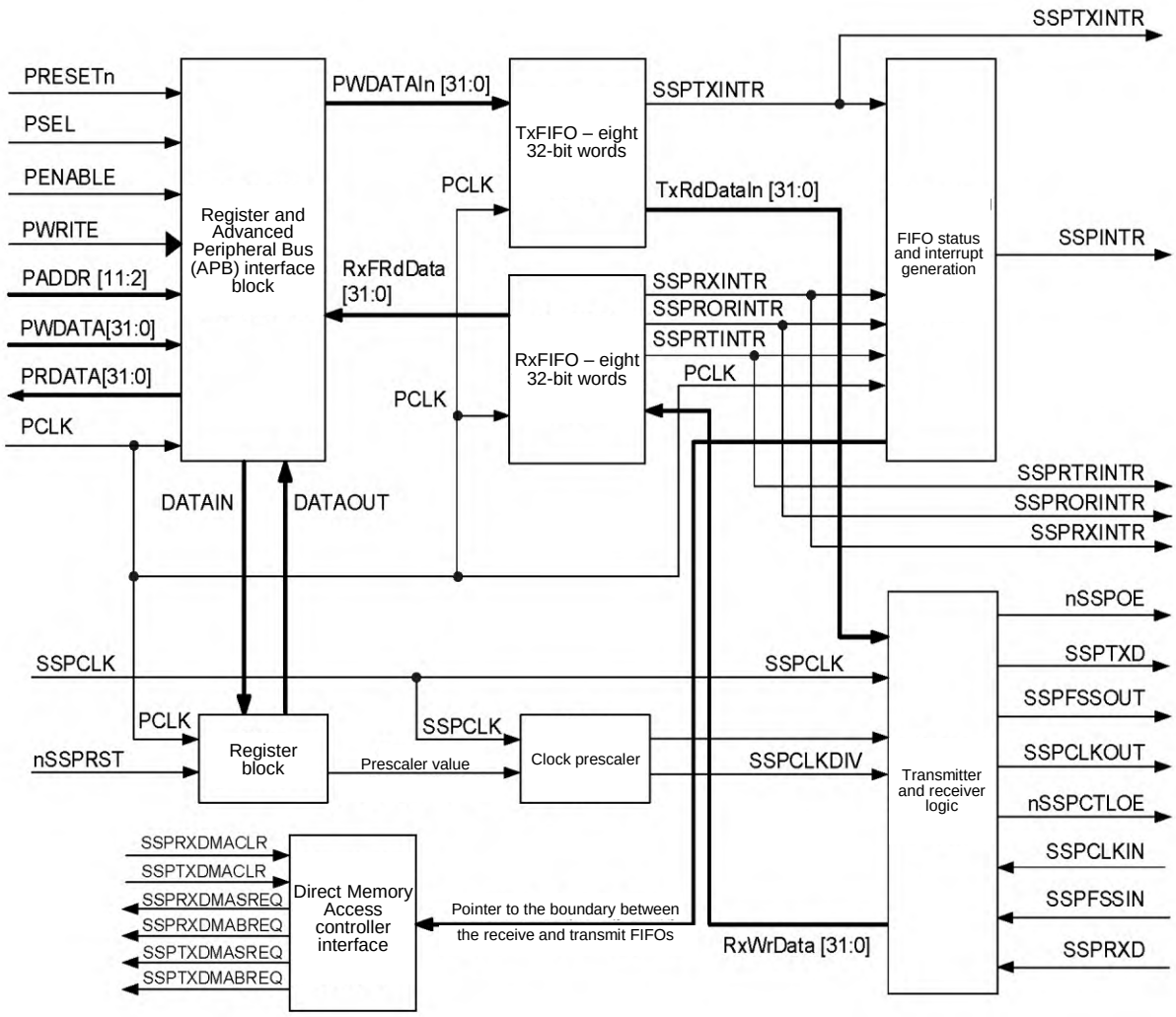


Figure 69 – SSP module block diagram

23.1.1. **Programmable parameters**

The following key parameters can be configured in software:

- Peripheral operating mode – master or slave;
- Operation enable/disable;
- Frame format;
- Data transfer rate;
- Clock phase and polarity;
- Data block length – 4 to 32 bits;
- Transmit FIFO reset;
- Interrupt masking.

23.1.2. **SPI interface characteristics**

The Motorola SPI synchronous serial interface provides:

- Full-duplex data transfer over a four-wire connection;
- Software configuration of clock phase and polarity.

23.1.3. **Microwire interface characteristics**

The National Semiconductor Microwire interface provides:

- Half-duplex data transfer using 8-bit control sequences.

23.1.4. **SSI interface characteristics**

The Texas Instruments SSI interface provides:

- Full-duplex data transfer over a four-wire connection;
- The ability to place the data line in a third state (high-impedance).

23.2. **SSP module overview**

The SSP module is a synchronous serial communication interface that can operate as either a master or a slave device and supports the Motorola SPI, National Semiconductor Microwire, and Texas Instruments SSI protocols.

The module performs the following functions:

- Conversion of data received from a peripheral device from serial to parallel format;
- Conversion of data transmitted to a peripheral device from parallel to serial format;
- Data, control information, and status information are read and written by the CPU;
- Receive and transmit paths are buffered by FIFO memories that can store up to eight data words of up to 32 bits independently for receive and transmit.

Serial data is transmitted over SSPTXD and received on SSPRXD.

The SSP module includes programmable prescalers that generate the SSPCLKOUT data transfer clock (the signal driven on SSPSCKOUT and used to synchronize transmit and receive) from the signal applied to SSPCLK (the signal supplied to the SSP module by the clock generator block). The baud rate can exceed 2 MHz, depending on the SSPCLK frequency and the characteristics of the connected peripheral device.

The data transfer mode, frame format, and number of data bits are configured in software using the CR0 and CR1 control registers.

The module generates the following independently maskable interrupt requests:

- SSPTXINTR – transmit buffer service request;
- SSPRXINTR – receive buffer service request;
- SSPRORINTR – receive FIFO overrun;
- SSPRTINTR – timeout waiting for data to be read from the receive FIFO;
- SSPRNEINTR – data present in the receive FIFO;
- SSPTFEINTR – transmit FIFO empty;
- SSPTNBSYINTR – transmit shift register empty.

In addition, a combined interrupt signal SSPINTR is generated when any of the above independent, unmasked interrupt sources is active; this signal is routed to the CLIC controller.

The module also generates Direct Memory Access (DMA) request signals for use with the DMA controller.

Depending on the SSP operating mode, SSPFSSOUT is used either for frame synchronization (SSI interface, active high) or as a slave-select signal (SPI and Microwire interfaces, active low).

23.2.1. **Clock generation block**

In master mode, the module generates the SSPCLK data clock using an internal frequency divider consisting of two cascaded counters without a reset circuit.

The prescaler division factor from 2 to 254 in steps of 2 is set by writing the corresponding value to the CPSR register. Because the least significant bit of the division factor is not used, division by an odd factor is not possible. This guarantees generation of a symmetric clock signal (with equal high and low half-periods).

The clock signal generated in this way is then applied to a second clock divider, whose output provides the SSPCLK data transfer clock.

The division factor of the second divider is programmed from 1 to 256 by writing the corresponding value to the CR0 control register.

23.2.2. **Transmit FIFO buffer**

The transmit buffer is 32 bits wide, 8 words deep, and organized as a FIFO (First In First Out). Data written by the CPU remain in the buffer until they are read by the transmit data block.

When the SPI controller operates in slave mode, reading from an empty FIFO results in invalid data being output.

23.2.3. Receive FIFO buffer

The receive buffer is 32 bits wide, 8 words deep, and organized as a FIFO (First In First Out). Data received from the peripheral device is stored in this buffer by the receive block until it is read by the CPU.

23.2.4. Data transmit/receive block

23.2.4.1. Master mode

In master mode, the module generates the SSPCLK data transfer clock for connected slave devices. As described above, this signal is generated by dividing the frequency of the SSPCLK signal.

The transmit block sequentially reads data from the transmit FIFO buffer and converts them from parallel to serial format. The resulting stream of serial data and frame-synchronization elements, clocked by SSPCLKOUT, is then transmitted over the SSPTXD line to the slave devices.

The receive block converts data arriving synchronously on the SSPRXD line from serial to parallel format. It then loads it into the receive FIFO buffer, from which they can be read by the CPU.

23.2.4.2. Slave mode

In slave mode, the data transfer clock signal is generated by one of peripheral devices connected to the module and is applied to the SSPCLKIN line.

In this case, the transmit block, clocked by this external signal, reads data from the FIFO buffer and converts it from parallel to serial format. It then outputs the serial data stream and frame-synchronization elements to the SSPTXD line.

Similarly, the receive block converts data arriving on the SSPRXD line synchronously with SSPCLKIN from serial to parallel format and then loads it into the receive FIFO buffer, from which they can be read by the CPU.

23.2.5. Interrupt generation block

The SSP module generates independent, maskable active-high interrupts. In addition, a combined interrupt is generated by OR-ing these independent interrupt signals.

The combined interrupt signal is supplied to the CLIC interrupt controller, providing an additional way to mask the module as a whole and simplifying the design of modular device drivers.

23.3. **Direct Memory Access interface**

The module provides an interface to the DMA controller in accordance with the interaction scheme between the transceiver and the DMA controller.

23.4. **Transceiver configuration**

After reset, operation of the transceiver blocks is disabled until the configuration procedure is completed.

For this, master or slave mode and the data transfer protocol (Motorola SPI, Texas Instruments SSI, or National Semiconductor Microwire) are selected, and then the required configuration is written to the CR0 and CR1 control registers.

To set the required data rate, the parameters of the clock generation block are selected with regard to the SSPCLK frequency, and the corresponding value is written to the PSR register.

23.5. **Enabling the transceiver**

The transceiver is enabled by setting the SSE bit in the CR1 control register. The transmit FIFO buffer may either be preloaded by writing up to eight 32-bit words before this bit is set, or it may be filled with transmit data during interrupt service.

After the module is enabled, the transceiver starts data exchange on the SSPTXD and SSPRXD lines.

23.6. **Clock frequency signals relationships**

The module imposes a constraint on the relationship between the CPU_CLK and SSPCLK clock frequencies. The SSPCLK frequency must be less than or equal to CPU_CLK.

$$F_{SSPCLK} \leq F_{PCLK.}$$

Meeting this requirement guarantees that control signals transferred from the SSPCLK clock domain to the CPU_CLK clock domain are synchronized within a time shorter than the duration of one frame transfer.

In slave mode, the SSPCLKIN signal from the external master device is applied to synchronization, delay, and edge-detection circuits. Three SSPCLK cycles are required to detect an edge on SSPCLKIN. The SSPTXD signal has a shorter setup time relative to the falling edge of SSPCLKIN, which is used to read data from the line. The setup and hold times of SSPRXD relative to SSPCLKIN must be chosen with sufficient margin to ensure correct data reading. For correct operation, the SSPCLK frequency must be at least 12 times the maximum expected SSPCLKIN frequency.

The selected SSPCLK frequency must support the required range of data rates. The ratio of the minimum SSPCLK frequency to the maximum SSPCLKOUT frequency is 12 in slave mode and 2 in master mode.

Thus, in master mode, to achieve a maximum baud rate of 1.8432 Mbit/s, the SSPCLK frequency must be at least 3.6864 MHz. In this case, the CPSR register must be set to 2, and the SCR[7:0] field in the CR0 register must be set to 0.

In slave mode, to achieve the same data rate, an SSPCLK frequency of at least 22.12 MHz must be used. Here, the CPSR register must be set to 12, and the SCR[7:0] field in the CR0 register must be set to 0.

The ratio of the maximum SSPCLK frequency to the minimum SSPCLKOUT frequency is 254 × 256.

The minimum allowable SSPCLK frequency is defined by the following set of relations, which must all be satisfied simultaneously:

$$\begin{cases} F_{SSPCLK}\ (min) \geq 2 \times F_{SSPCLKOUT}\ (\max) \text{ [master mode]}, \\ F_{SSPCLK}\ (min) \geq 12x \times F_{SSPCLKIN}\ (\max) \text{[slave mode]}. \end{cases}$$

When the SSPFRX signal in the CR0 control register is enabled in SPI mode, this relation takes the form:

$$\begin{cases} F_{SSPCLK}\ (min) \geq 2 \times F_{SSPCLKOUT}\ (\max) \text{[master mode]}, \\ F_{SSPCLK}\ (min) \geq 4 \times F_{SSPCLKIN}\ (\max) \text{ [slave mode]}. \end{cases}$$

Similarly, the maximum allowable SSPCLK frequency is defined by the following set of relations, which must all be satisfied simultaneously:

$$\begin{cases} F_{SSPCLK}\ (max) \leq 254 \times 256 \times F_{SSPCLKOUT}\ (\min) \text{[master mode]}, \\ F_{SSPCLK}\ (max) \leq 254 \times 256 \times F_{SSPCLKIN}\ (\min) \text{ [slave mode]}. \end{cases}$$

23.7. **Programming the CR0 control register**

The CR0 register is used to:

- set the data transfer rate;
- select one of three data transfer protocols;
- select the data word length;
- enable high-speed mode for the SPI slave device.

The data transfer rate depends on the external SSPCLK clock frequency and the division factor of the clock generation block. This division factor is defined by the combination of the SCR (Serial Clock Rate) field in the CR0 register and the CPSDVSR (clock prescale divisor value) field in the CPSR register.

The data frame format is selected by the FRF field, and the data word length is selected by the DSS field of the CR0 register.

The SSPFRX signal (available starting from revision 2) is used to enable high-speed mode in the SPI slave module and to enable synchronization of the SSPRXD signal.

For the Motorola SPI protocol, the clock polarity and phase are additionally configured by the SPO and SPH bits.

23.8. **Programming the CR1 control register**

The CR1 register is used to:

- select master or slave operating mode of the transceiver;
- enable loopback test mode;
- enable or disable the module;
- perform a software reset of the transmit FIFO.

Master mode is selected by writing 0 to the MS field of the CR1 register (this value is also set automatically after reset).

Writing 1 to the MS field switches the transceiver to slave mode. In slave mode, enabling or disabling generation of the SSPTXD transmit signal is controlled by the SOD (slave mode SSPTXD output disable) bit in the CR1 register. This function is useful when several slave devices are connected to a common line.

To enable transceiver operation, the SSE (Synchronous Serial Port Enable) bit must be set to 1.

To perform a software reset of the transmit FIFO, the RESTxFIFO bit must be set to 1. After the reset is completed, RESTxFIFO is automatically cleared to 0.

23.9. **Data clock signal generation**

The data clock is generated by dividing the SSPCLK clock frequency. At the first stage, the SSPCLK frequency is divided by an even factor CPSDVSR in the range from 2 to 254, programmed through the CPSR register. The resulting signal is then applied to a divider with a division factor of (1 + SCR) from 1 to 256, where SCR is programmed through the CR0 register.

The resulting SSPCLK data clock frequency is defined by the following relation:

$$F_{SSPCLKOUT} = \frac{F_{SSPCLK}}{CPSDVR \cdot (1 + SCR)} \quad (25)$$

For example, if the SSPCLK frequency is 3.6864 MHz and CPSDVSR = 2, the SSPCLKOUT frequency lies in the range from 7.2 kHz to 1.8432 MHz.

23.10. **Data frame format**

Each data frame contains 4 to 32 bits of data, depending on the programmed value. Data transmission starts from the most significant bit. Three basic frame structures can be selected:

- Texas Instruments SSI;
- Motorola SPI;
- National Semiconductor Microwire.

In all frame formats, the SSPCLK clock signal is generated only when the transceiver is ready for data exchange. Driving SSPCLK to the inactive state is used as a receiver timeout indication, meaning that unprocessed data remains in the receive buffer after the specified time interval has expired.

In SPI and Microwire modes, the SSPFSS transmit frame-synchronization output is active low and is held low for the entire duration of the data frame transfer.

In the Texas Instruments SSI frame format, before each data frame the SSPFSS output generates a pulse with a duration of one data-clock period. In this mode, the SSP transceiver and the slave peripheral devices drive data onto the line on the rising edge of SSPCLK and read data from the line on the falling edge of this signal.

Unlike the full-duplex SSI and SPI data transfer modes, the National Semiconductor Microwire mode uses a special half-duplex data exchange between master and slave devices. In this mode, before data frame transmission begins, the master device sends a special 8-bit control sequence to the external slave device. While this sequence is being transmitted, the receiver does not process any input data. After the sequence has been transmitted and decoded by the slave device, it waits one clock period after the last bit of the control sequence and then sends the requested data to the master. The data block from the slave device can be 4 to 32 bits long, so the total frame length ranges from 13 to 41 bits.

23.10.1. **Texas Instruments SSI synchronous transfer format**

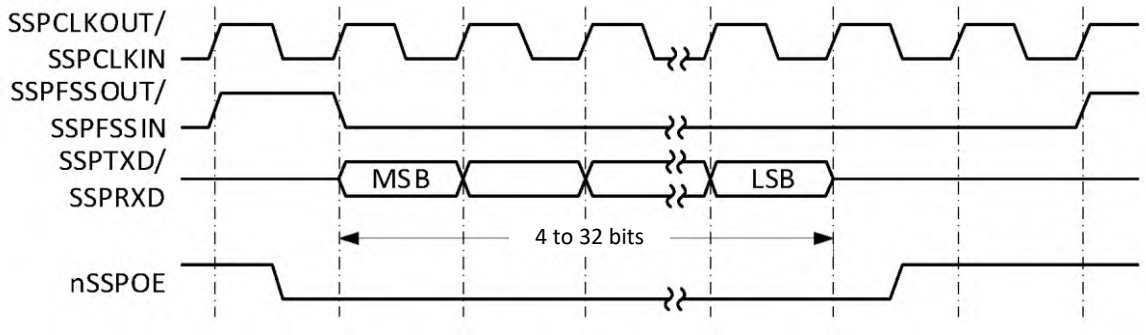


Figure 70 – SSI protocol synchronous transfer format (single transfer)

In this mode, when the SSP transceiver is inactive, the SSPCLK and SSPFSS signals are driven to low, and the SSPTXD data line is held in the high impedance state.

After at least one element appears in the transmit FIFO buffer, the SSPFSSOUT signal is driven high for a time corresponding to one SSPCLK period. The value from the FIFO buffer is loaded into the transmit block shift register. On the next rising edge of SSPCLK, the most significant bit of the frame (2–32 data bits) is driven onto the SSPTXD line, etc.

In receive mode, both the SSP module and the external slave device shift data bits into their shift registers on the falling edge of SSPCLK. After the least significant bit has been shifted in, the received data is transferred from the shift register to the receive FIFO buffer on the next rising edge of SSPCLK.

Timing diagrams for Texas Instruments SSI synchronous serial transfer protocol are shown in Figure 70 (single frame transfer) and Figure 71 (continuous frame transfer).

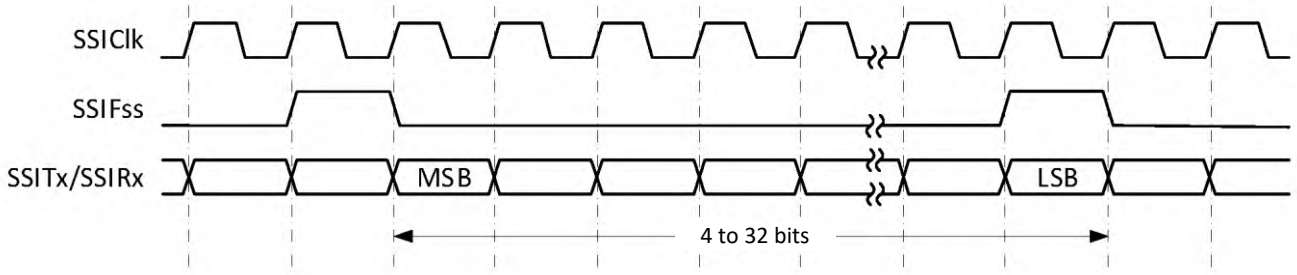


Figure 71 – SSI protocol synchronous transfer format (continuous transfer)

23.10.2. **Motorola SPI synchronous transfer format**

The Motorola SPI interface uses four signal lines, with SSPFSS serving as the slave-select signal. A key feature of the SPI protocol is the ability to select the idle level and phase of the SSPCLK signal (when the transceiver is inactive) by setting the SPO and SPH bits in the CR0 control register.

23.10.2.1. **Clock polarity selection – SPO bit**

If SPO is 0, the SSPCLK line is driven low in the idle state. Otherwise, when no data is being transferred, SSPCLK is driven high in the idle state.

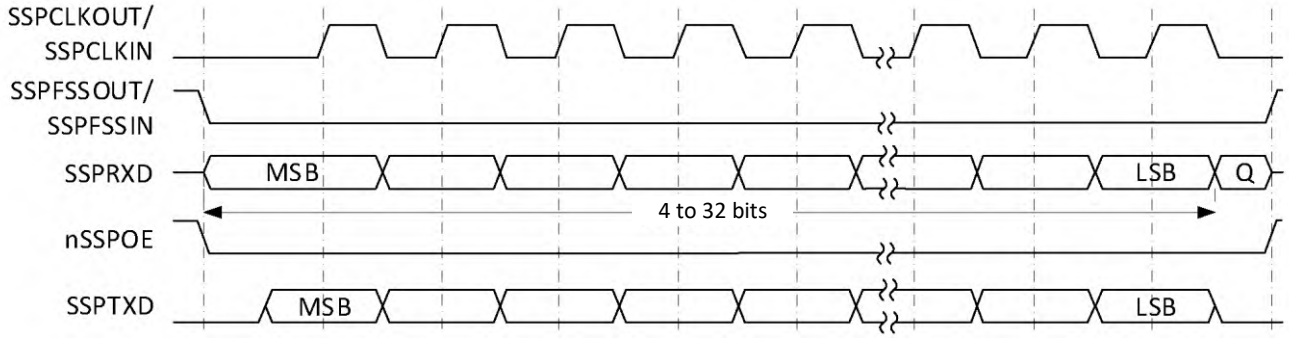
23.10.2.2. **Clock phase selection – SPH bit**

The SPH bit defines which clock edge is used to sample data and update the output state on the line.

If SPH is 0, the receiver registers data on the rising edge of the clock; otherwise, data is registered on the falling edge.

23.10.3. Motorola SPI synchronous transfer format, SPO=0, SPH=0

Timing diagrams for SPI synchronous serial transfer with SPO = 0 and SPH = 0 are shown in Figure 72 (single transfer) and Figure 73 (continuous transfer).



Q is a signal with an undefined level.

Figure 72 – SPI protocol synchronous transfer format, SPO = 0, SPH = 0 (single transfer)

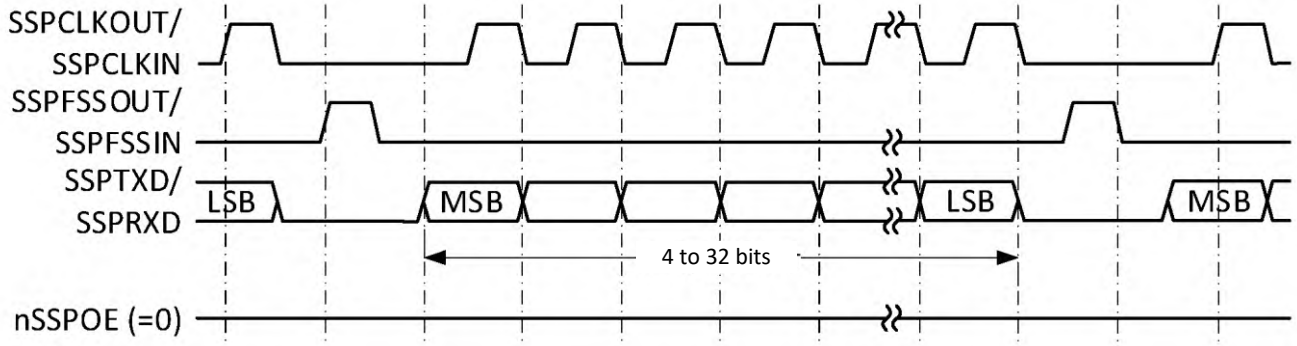


Figure 73 – SPI protocol synchronous transfer format, SPO = 0, SPH = 0 (continuous transfer)

In this mode, while waiting for the transceiver:

- the SSPCLK signal is low;
- the SSPFSS signal is high;
- the SSPTXD signal is driven to the high-impedance state.

If the module is enabled and the transmit FIFO buffer contains valid data, SSPFSS is driven low, which indicates the start of data transfer and allows the slave device to drive data onto the master SSPRXD input line. At the same time, the SSPTXD line switches from the high-impedance state to the active state.

After half a SSPCLK period, the value of the first data bit is generated in SSPTXD. By this time, data must already be present on the data transfer lines at both the master and the slave devices. After the next half-period, SSPCLK is driven high.

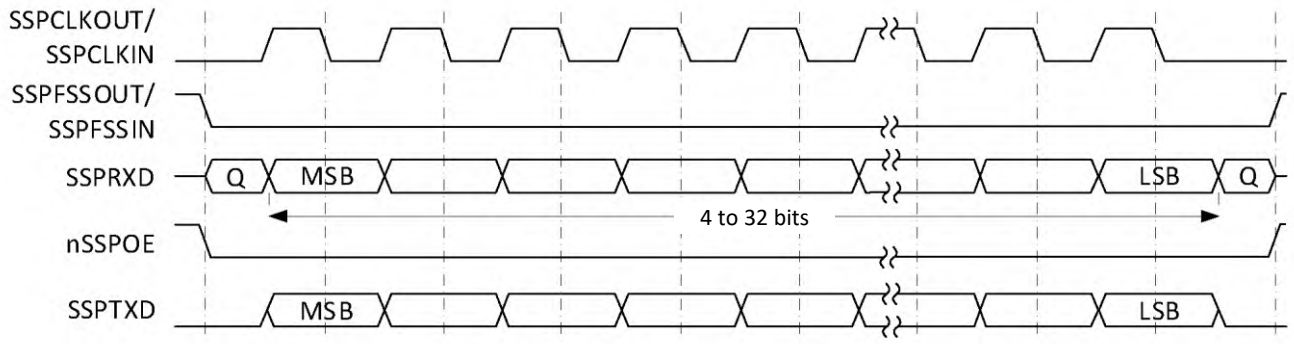
From that point on, data is registered on the rising edge and driven onto the line on the falling edge of SSPCLK.

For a single data-word transfer, after the last bit has been received, SSPFSS is driven high one SSPCLK period later.

In continuous transfer mode, SSPFSS must generate high-level pulses between transfers of each data word. This is because in SPH = 0 mode a low level on the slave-select line blocks loading into the shift register. Therefore, the master drives SSPFSS high at the end of each frame, allowing new data to be written. After the last bit of the data block is received, SSPFSS is driven to the standby state one SSPCLK period later.

23.10.4. Motorola SPI synchronous transfer format, SPO=0, SPH=1

Timing diagrams for SPI synchronous serial transfer with SPO=0 and SPH=1 are shown in Figure 74 (single and continuous transfer).



Q is a signal with an undefined level.

Figure 74 – SPI protocol synchronous transfer format, SPO=0, SPH=1

In this mode, while waiting for the transceiver:

- the SSPCLK signal is low;
- the SSPFSS signal is high;
- the SSPTXD signal is driven to the high-impedance state.

If the module is enabled and the transmit FIFO buffer contains valid data, SSPFSS is driven low, which indicates the start of data transfer and allows the slave device to drive data onto the master SSPRXD input line. The SSPTXD output pin switches from the high-impedance state to the active state.

After half a SSPCLK period, the first data bits are generated in the data lines of both the master and the slave devices. At the same time, SSPCLK starts and a rising edge is formed on it.

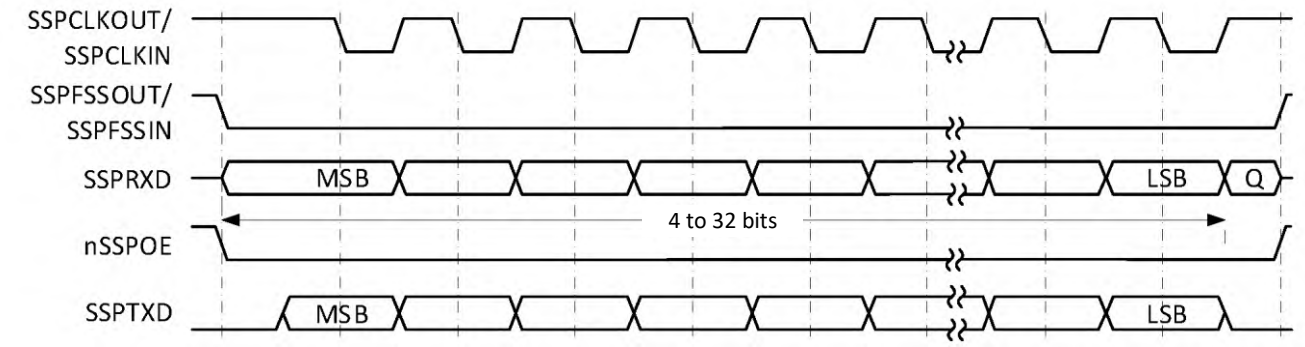
From that point on, data is registered on the falling edge and driven onto the line on the rising edge of SSPCLK.

For a single data-word transfer, after the last bit has been received, SSPFSS is driven high one SSPCLK period later.

In continuous transfer mode, SSPFSS remains low for the entire duration and is driven high after the last bit of the data block has been received, as in the single-word transfer mode.

23.10.5. **Motorola SPI synchronous transfer format, SPO=1, SPH=0**

Timing diagrams for SPI synchronous serial transfer with SPO=1 and SPH = 0 are shown in Figure 75 (single transfer) and Figure 76 (continuous transfer).



Q is a signal with an undefined level.

Figure 75 – SPI protocol synchronous transfer format, SPO=1, SPH=0 (single transfer)

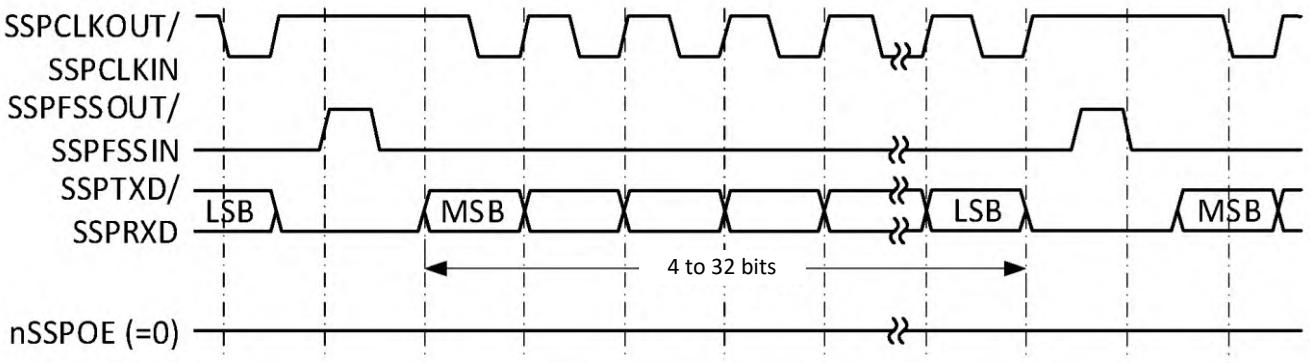


Figure 76 – SPI protocol synchronous transfer format, SPO=1, SPH = 0 (continuous transfer)

In this mode, while waiting for the transceiver:

– the SSPCLK signal is high;

– the SSPFSS signal is high;

– the SSPTXD signal is driven to the high-impedance state.

If the module is enabled and the transmit FIFO buffer contains valid data, SSPFSS is driven low, which indicates the start of data transfer and allows the slave device to drive data onto the master SSPRXD input line. The SSPTXD output pin switches from the high-impedance state to the active state.

After half a SSPCLK period, the value of the first data bit is generated in SSPTXD. By this time, data must already be present on the data transfer lines at both the master and the slave devices. After the next half-period, SSPCLK is driven low.

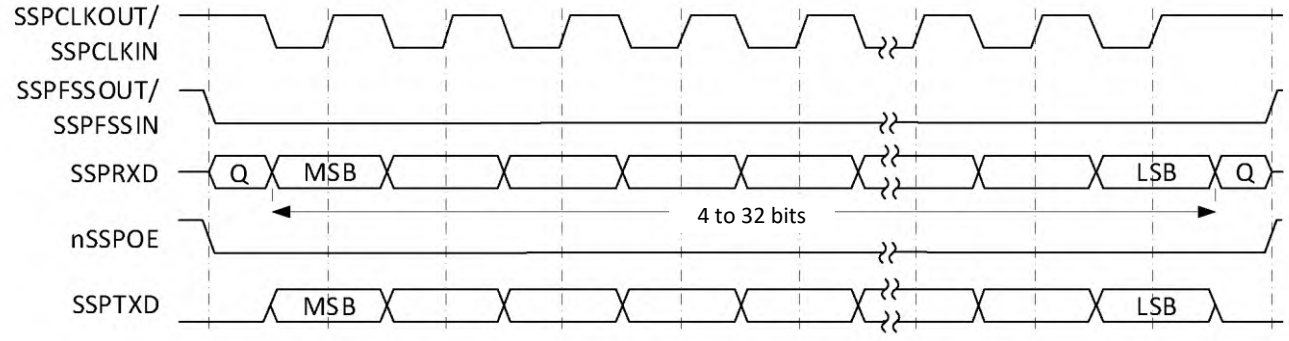
From that point on, data is registered on the falling edge and driven onto the line on the rising edge of SSPCLK.

For a single data-word transfer, after the last bit has been received, SSPFSS is driven high one SSPCLK period later.

In continuous transfer mode, SSPFSS must generate high-level pulses between transfers of each data word. This is because in SPH = 0 mode a low level on the slave-select line blocks loading into the shift register. Therefore, the master drives SSPFSS high at the end of each frame, allowing new data to be written. After the last bit of the data block is received, SSPFSS is driven to the standby state one SSPCLK period later.

23.10.6. **Motorola SPI synchronous transfer format, SPO=1, SPH=1**

Timing diagrams for SPI synchronous serial transfer with SPO=1 and SPH=1 are shown in Figure 77 (single and continuous transfer).



Q is a signal with an undefined level.

Figure 77 – SPI protocol synchronous transfer format, SPO=1, SPH=1

In this mode, while waiting for the transceiver:

– the SSPCLK signal is high;
– the SSPFSS signal is high;
– the SSPTXD signal is driven to the high-impedance state.

If the module is enabled and the transmit FIFO buffer contains valid data, SSPFSS is driven low, which indicates the start of data transfer and allows the slave device to drive data onto the master SSPRXD input line. The SSPTXD output pin switches from the high-impedance state to the active state.

After half a SSPCLK period, the first data bits are generated in the data lines of both the master and the slave devices. At the same time, SSPCLK starts and a rising edge is formed on it.

From that point on, data is registered on the rising edge and driven onto the line on the falling edge of SSPCLK.

For a single data-word transfer, after the last bit has been received, SSPFSS is driven high one SSPCLK period later.

In continuous transfer mode, SSPFSS remains low for the entire duration and is driven high after the last bit of the data block has been received, as in the single-word transfer mode.

23.10.7. National Semiconductor Microwire synchronous transfer format

Timing diagrams for Microwire synchronous serial transfer are shown in Figure 78 (single transfer) and Figure 79 (continuous transfer).

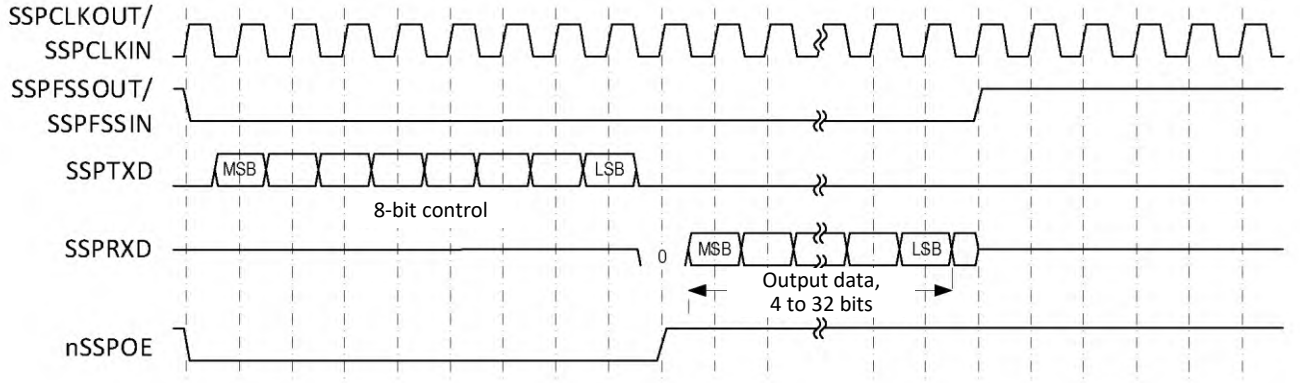


Figure 78 – Microwire protocol synchronous transfer format (single transfer)

The Microwire data transfer protocol is largely similar to the SPI protocol, except that transfer is performed in half-duplex mode and uses control sequences. Each data transfer begins with the master device sending a special 8-bit control sequence. While this sequence is being transmitted, the receiver does not process any input data. After the sequence has been transmitted and decoded by the slave device, it waits one clock period after the last bit of the control sequence and then sends the requested data to the master. The data block from the slave device can be 4 to 32 bits long, so the total frame length ranges from 13 to 41 bits.

In this mode, while waiting for the transceiver:

- the SSPCLK signal is low;
- the SSPFSS signal is high;
- the SSPTXD signal is driven to the high-impedance state.

Transition to data transfer mode occurs after the control byte is written to the transmit FIFO buffer. On the falling edge of SSPFSS, data from the buffer is loaded into the transmit block shift register and then driven onto the SSPTXD line bit by bit, starting from the most significant bit. The SSPFSS line remains low for the entire frame transfer. The SSPRXD line is in the high-impedance state during this time.

The external slave device receives data bits on the rising edge of SSPCLK. After the last bit of the control sequence is received, the sequence is decoded within one clock period, after which the slave device sends the requested data to the SSP module. Data bits are driven onto SSPRXD on the falling edge of SSPCLK. The master device, in turn, registers them on the rising edge of this clock signal. In the case of a single data transfer, after the last bit of the data word is received, SSPFSS is driven high for one clock period. This serves as a command to transfer the received data word from the shift register into the receive FIFO buffer.

Note: The external device may place the receive line in the high impedance state on the falling edge of SSPCLK after the last data bit is received, or after SSPFSS is driven high.

Continuous transfer starts and ends in the same way as a single transfer. However, the SSPFSS line is held low for the entire data transfer session. The control byte of the next frame is sent immediately after the least significant bit of the current frame is received. Data from the shift register is transferred into the receive FIFO buffer after the least significant bit of the next word is sampled on the falling edge of SSPCLK.

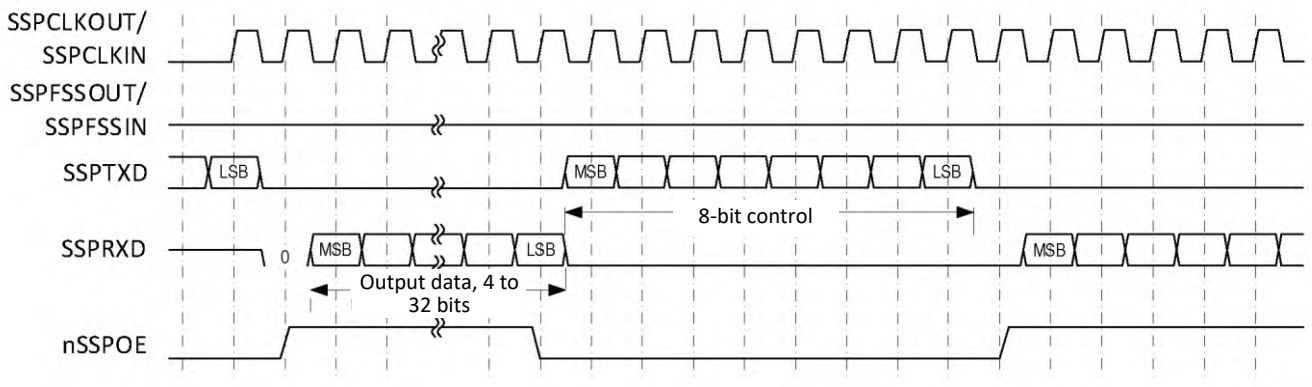


Figure 79 – Microwire protocol synchronous transfer format (continuous transfer)

23.10.7.1. **Timing requirements for the SSPFSS signal relative to the SSPCLK clock signal in Microwire mode**

An SSP module operating in Microwire mode as a slave registers data on the rising edge of SSPCLK after SSPFSS is driven low. Master devices that generate the SSPCLK signal must guarantee sufficient setup and hold times for SSPFSS relative to the rising edge of SSPCLK.

These requirements are shown in Figure 80. With respect to the rising edge of SSPCLK used to register data in the slave SSP receiver, the SSPFSS setup time must be at least twice the SSPCLK period used by the module. With respect to the previous rising edge of SSPCLK, the hold time must be at least one period of this clock signal.

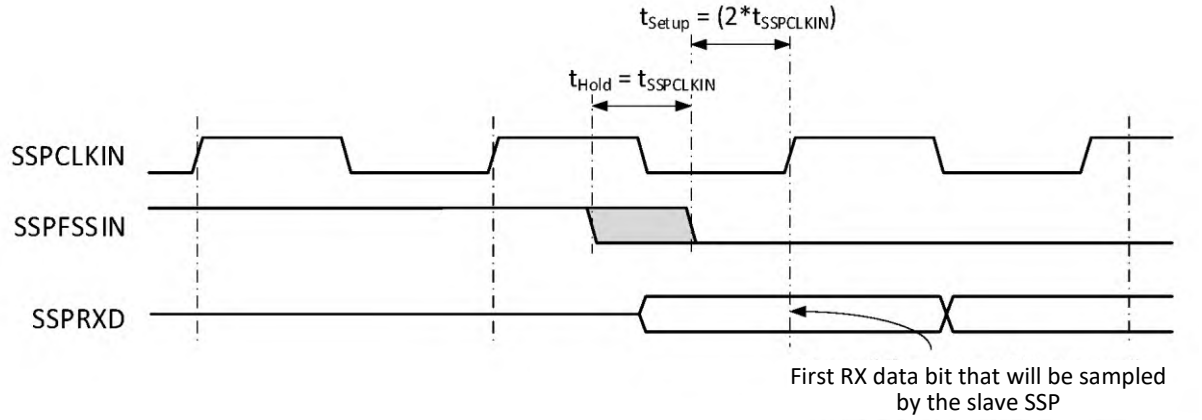


Figure 80 – Microwire format, SSPFSSIN setup and hold time requirements

23.11. **Examples of module configuration in master and slave modes**

Figures 81–83 show examples of connecting the SSP module to peripheral devices operating in master or slave mode.

Note: The SSP module does not support dynamic switching between master and slave modes. Each transceiver must be initially configured in one of these modes.

Figure 81 shows joint operation of three SSP modules, with one configured as a master and two as slave devices. The master device can transmit data to both slaves over the SSPTXD line in circular manner.

For return data transfer, one of the slave modules enables propagation of the signal from its SSPTXD line to the master SSPRXD input.

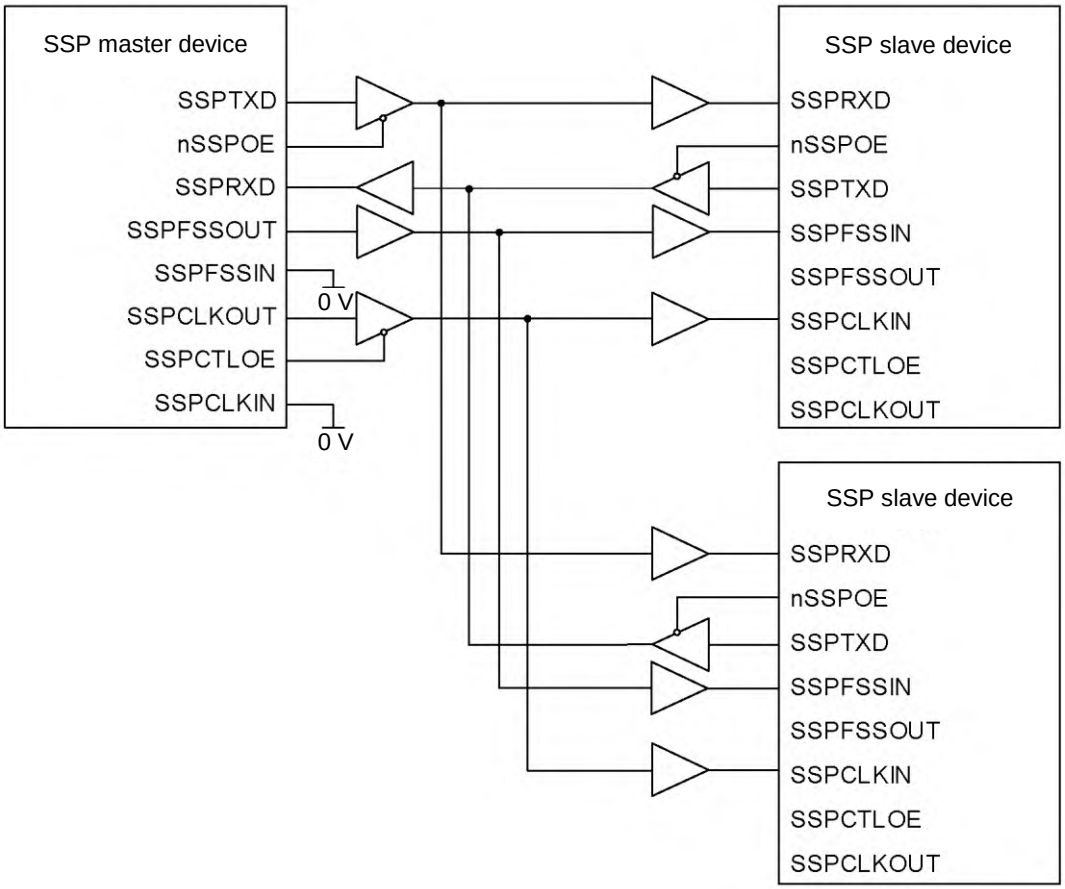


Figure 81 – SSP master device connected to two SSP slaves

Figure 82 shows an SSP module configured as a master connected to two slave devices that support the Motorola SPI protocol. The external devices are configured as slaves by driving their Slave Select (SS) signal low. As in the previous example, the master device can transmit data to the slaves in circular manner the SSPTXD line. Return data transfer to the master SSPRXD input is performed by only one slave at a time over its MISO line.

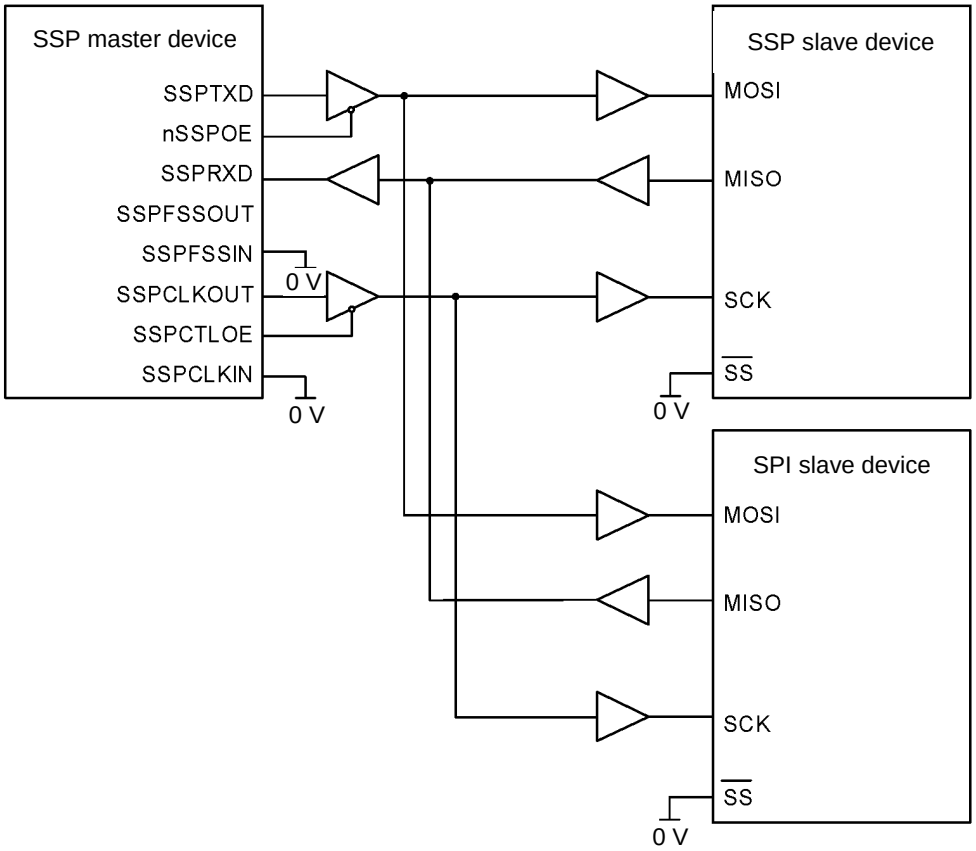


Figure 82 – SSP master device connected to two slaves supporting SPI

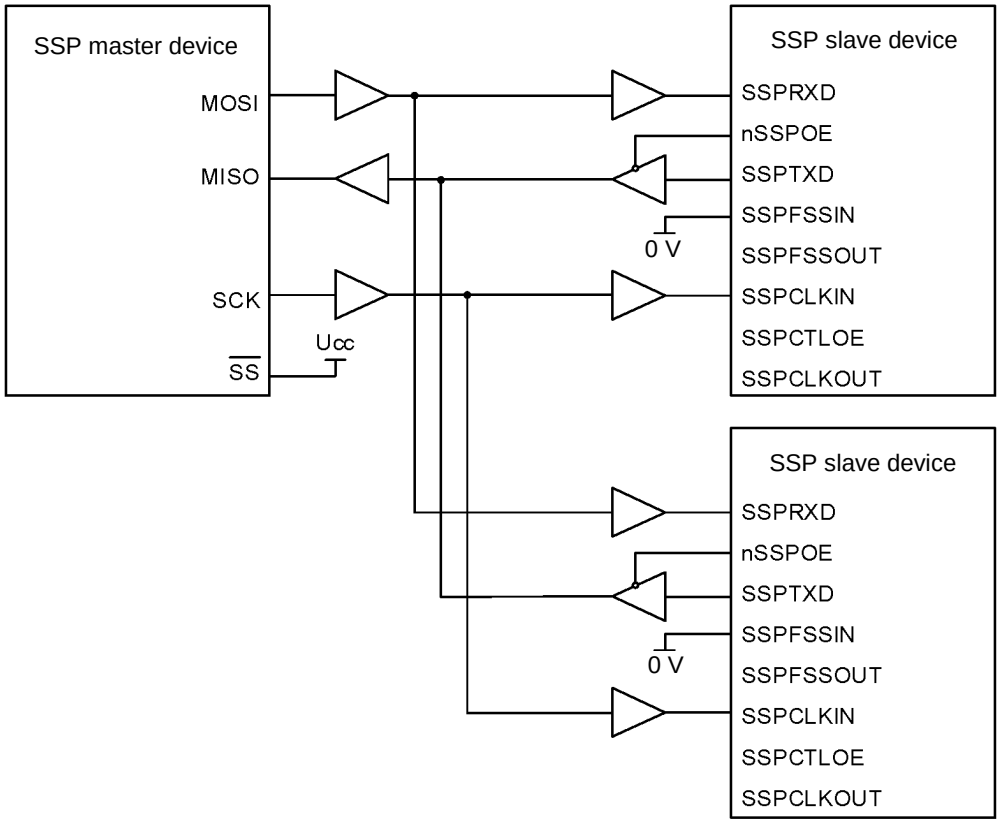


Figure 83 – SPI master device connected to two SSP slave modules

Figure 83 shows a master device that supports the Motorola SPI protocol connected to two SSP modules configured for operation in slave mode. In this case, the master Slave Select (SS) line is held high. The master device transmits data over the MOSI line in a circular fashion to both slave modules.

For return data transfer, one of the slave modules drives its SSPTXD line active, enabling propagation of the signal from SSPTXD to the master SSPRXD input.

23.12. **Direct Memory Access interface**

The SSP module provides an interface for connection to a Direct Memory Access controller. Operation in this mode is controlled by the DMACR DMA control register.

The DMA interface includes the following signals:

- for receive:
- SSPRXDMASREQ – single-word transfer request, asserted by the transceiver. The signal is driven active when the receive FIFO buffer contains at least one word.
- SSPRXDMABREQ – block transfer request, asserted by the transceiver. The signal is driven active when the receive FIFO buffer contains four or more words.
- SSPRXDMACLR – DMA request clear, asserted by the DMA controller to clear a received request. If a block transfer was requested, the clear signal is generated during transfer of the last data word in the block.
- for transmit:
- SSPTXDMASREQ – single-word transfer request, asserted by the transceiver. The signal is driven active when the transmit FIFO buffer contains at least one free location.
- SSPTXDMABREQ – block transfer request, asserted by the transceiver. The signal is driven active when the transmit FIFO buffer contains four or fewer words.
- SSPTXDMACLR – DMA request clear, asserted by the DMA controller to clear a received request. If a block transfer was requested, the clear signal is generated during transfer of the last data word in the block.

Block and single-word transfer requests are not mutually exclusive and can be asserted simultaneously. For example, if the receive FIFO buffer fill level exceeds the threshold of four words, both the single-word and block transfer request signals are asserted. If the amount of data in the receive buffer is below the threshold, only the single-word transfer request is asserted. This is useful when the total data volume is smaller than the block size. For example, suppose 19 characters must be received. In this case, the DMA controller performs four block transfers of four characters each, and the remaining three characters are transferred in three single-word transfers.

Note – For these last three characters the SSP controller does not initiate a block transfer.

Each DMA request asserted by the transceiver remains active until it is cleared by the corresponding DMACLR signal.

After the clear signal is deasserted, the transceiver can again generate a DMA request when the conditions described above are met. All DMA requests are cleared when the transceiver is disabled, as well as when the DMA enable signal is deasserted.

Table 401 provides the threshold fill levels of the receive and transmit buffers required to assert the DMABREQ block transfer request.

Table 401 – Parameters for triggering block data transfer requests in DMA mode

Threshold level	Data block length	
	Transmit buffer (number of empty locations)	Receive buffer (number of filled locations)
½	4	4

Figure 84 shows the timing diagrams for single-word and block DMA requests, including the effect of the DMACLR signal. All signals must be synchronized to PCLK.

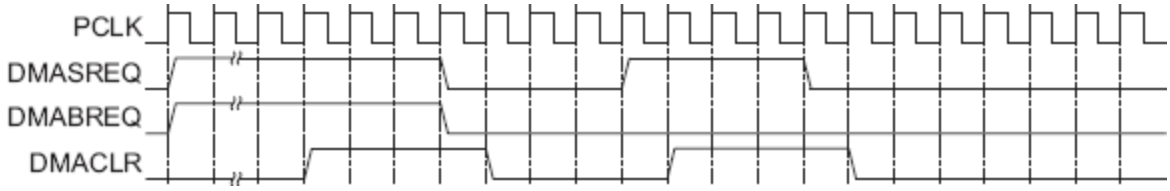


Figure 84 – DMA mode transfer timing diagrams

23.13. **Interrupts**

The module provides seven maskable interrupt request lines, combined into a single output signal that is a logical OR of the individual sources.

Interrupt request signals:

- SSPRXINTR – receive FIFO buffer service request;
- SSPTXINTR – transmit FIFO buffer service request;
- SSPRORINTR – receive FIFO buffer overrun;
- SSPRTINTR – receiver timeout;

- SSPRNEINTR – data present in the receive FIFO;
- SSPTFEINTR – transmit FIFO empty;
- SSPTNBSYINTR – transmit shift register empty;
- SSPINTR – logical OR of SSPRXINTR, SSPTXINTR, SSPRTINTR, SSPRORINTR, SSPRNEINTR, SSPTFEINTR, and SSPTNBSYINTR.

Each individual interrupt request signal can be masked by setting the corresponding bit in the SSPIMSC interrupt mask register. Setting a bit to 1 enables the corresponding interrupt; setting it to 0 disables it.

Availability of both individual interrupt lines and the combined interrupt line makes it possible to handle interrupts either through a system-wide handling procedure or using a modular device driver.

The receiver and transmitter interrupts SSPRXINTR and SSPTXINTR are output separately from interrupts related to changes in the device state. This allows these request signals to be used to coordinate data reads and writes with reaching the configured threshold levels of the receive and transmit FIFO buffers.

The conditions for each interrupt source can be read either from the raw interrupt status register SSPRIS or from the masked interrupt status register SSPMIS.

23.13.1. **SSPRXINTR**

Receive FIFO fill interrupt. Asserted if the receive buffer contains four or more unread data words.

23.13.2. **SSPTXINTR**

Transmit FIFO fill interrupt. Asserted if the transmit buffer contains four or fewer valid data words.

The interrupt condition does not depend on the enable state of the SSP module. This makes it possible to organize interaction between the software and the transmitter in two ways. First, data can be written to the buffer in advance, before enabling the transmitter and interrupts. Second, the module and interrupt generation can be enabled first, and the transmit buffer can then be filled within the interrupt service routine.

23.13.3. **SSPRORINTR**

Receive FIFO overrun interrupt is asserted if the buffer is already full and the receiver block attempts to write one more word. In this case, the received data word is registered in the receiver shift register but is not written into the receive buffer.

23.13.4. **SSPRTINTR**

Receiver timeout interrupt occurs if the receive FIFO buffer is not empty and no new data has arrived at the receiver input for a timeout interval equal to 32 SSPCLKOUT cycles (in both master and slave modes). This mechanism guarantees that the presence of unprocessed data in the receive FIFO is known to the user.

The timeout interrupt is cleared either after the data has been read from the receive buffer until it becomes empty, or after new data words are received on the SSP_RXD input line. It can also be cleared by writing 1 to the RTIC bit in the SSPTICR interrupt clear register.

23.13.5. **SSPRNEINTR**

Receive FIFO not-empty interrupt. This signal is asserted when at least one word is written into the receive FIFO and remains asserted until all data has been read.

23.13.6. **SSPTFEINTR**

Transmit FIFO empty interrupt. Remains active until at least one word is written into the FIFO. After all data has been transmitted, it returns to the active state.

23.13.7. **SSPTNBSYINTR**

Transmit shift register empty interrupt.

23.13.8. **SSPINTR**

of the interrupt request signals described above are combined into a single line by OR'ing SSPRXINTR, SSPTXINTR, SSPRNEINTR, SSPTNBSYINTR, SSPTFEINTR, SSPRTINTR, and SSPRORINTR, taking masking into account. The combined output can be connected to the system interrupt controller, which allows additional masking of requests at the peripheral-device level.

23.14. **Software control of the module**

23.14.1. **General**

The following addresses are reserved and must not be used in normal operation:

- addresses with offsets in the range +0x028… +0x07C and +0xFD0… +0xFDC are reserved for possible future extensions of the module;
- addresses with offsets in the range +0x080… +0x088 are reserved for testing.

23.14.2. **Description of SSP controller registers**

Information on the SSP controller registers is given in Table 402.

Table 402 – Summary of SSP module registers

Base address	Name				Description
0x5000_0000	SSP1				SSP1 interface controller registers
0x500C_8000	SSP2				SSP2 interface controller registers
0x500D_0000	SSP3				SSP3 interface controller registers
Offset	Name	Type	Reset value	Size, bits	Description
0x000	CR0	RW	0x0000	18	Control register 0
0x004	CR1	RW	0x0	5	Control register 1
0x008	DR	RW	0x----	32	Receive FIFO buffer (read) Transmit FIFO buffer (write)
0x00C	SR	RO	0x03	5	Status register
0x010	CPSR	RW	0x00	8	Clock divider register
0x014	IMSC	RW	0x0	7	Interrupt mask register
0x018	RIS	RO	0x8	7	Unmasked interrupt status register
0x01C	MIS	RO	0x0	7	Masked interrupt status register
0x020	ICR	WO	0x0	2	Interrupt clear register
0x024	DMACR	RW	0x0	2	Direct Memory Access control register

Note: The “Type” column indicates the register access type:
RW – read/write;
RO – read-only;
WO – write-only.

23.14.2.1. **CR0 register**

Control register 0

The CR0 register contains seven bit fields that control SSP module blocks.

The CR0 bit functions are given in Table 403.

Table 403 – CR0 register format

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…18	-	Reserved
17	SSPFRX	For a slave module, enables SPI fast mode and activates RXD signal synchronization: 1 – slave fast mode enabled; 0 – normal slave mode
16	DSS[4]	Most significant bit of the data word length: 0 – DSS[3:0]; 1 – DSS[3:0] + 16 bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
15…8	SCR	Serial transfer rate. The SCR field value is used when generating the data transfer clock signal. The bit rate satisfies the relation $\frac{F_{SSPCLK}}{(CPSDVR * (1 + SCR))}$, where CPSDVR is an even number in the range 2 to 254 (see CPSR register), and SCR is a number from 0 to 255.
7	SPH	SSPCLKOUT phase (used only in Motorola SPI mode). See subsection “Motorola SPI synchronous transfer format”.
6	SPO	SSPCLKOUT polarity (used only in Motorola SPI mode). See subsection “Motorola SPI synchronous transfer format”.
5…4	FRF	Data frame format: 00 – Motorola SPI protocol; 01 – Texas Instruments SSI protocol; 10 – National Semiconductor Microwire protocol; 11 – reserved
3…0	DSS[3:0]	Least significant 4 bits of the data word length. If DSS[4] = 0: 0000 – reserved 0001 – reserved 0010 – reserved 0011 – 4 bits 0100 – 5 bits 0101 – 6 bits … 1110 – 15 bits 1111 – 16 bits If DSS[4] = 1: 0000 – 17 bits 0001 – 18 bits 0010 – 19 bits 0011 – 20 bits 0100 – 21 bits 0101 – 22 bits … 1110 – 31 bits 1111 – 32 bits

23.14.2.2. **CR1 register**

Control register 1

The CR1 register contains five bit fields that control SSP module blocks.

The CR1 bit functions are given in Table 404.

Table 404 – CR1 register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
15…5		Reserved, read value is undefined. When writing, this bit must be set to 0.
4	RESTxFIFO	Software reset of the transmit FIFO: 0 – normal transmit FIFO operation; 1 – transmit FIFO reset active. Automatically cleared to 0 after the reset is completed.
3	SOD	Disabling output lines in slave mode. This bit is used only in slave mode (MS = 1). It allows bidirectional data exchange in systems that contain one master and several slave devices. The SOD bit must be set when this slave SSP module must not currently drive data on the SSP_TXD line. In this case, data lines of slave devices can be wired in parallel. 0 – SSP_TXD control in slave mode enabled; 1 – SSP_TXD control in slave mode disabled
2	MS	Selection of master or slave mode: 0 – master module (default after reset); 1 – slave module
1	SSE	Enabling the transceiver: 0 – operation disabled; 1 – operation enabled
0	LBM	Loopback test: 0 – normal transceiver operation; 1 – transmitter shift-register output connected to the receiver shift-register input

23.14.2.3. **DR register**

Data register

The SSPDR register is 32 bits wide and is used to read received data and write data to be transmitted.

A read operation provides access to the last unread word in the receive FIFO buffer. Writing data into this FIFO buffer is performed by the receiver block.

A write operation loads the next word into the transmit FIFO buffer. The transmitter block takes data from this buffer. The taken data is placed into the transmitter shift register, and then outputs it serially on the SSP_TXD line at the configured bit rate.

If the selected data word length is less than 32 bits, data must be right-aligned before being written to the SSPDR register. The transmitter block ignores unused bits. Received data words are automatically right-aligned by the receiver block.

In National Semiconductor Microwire mode the SSP module, by default, operates with 8-bit data words (the most significant byte is ignored). The size of the received data word is configured in software. The receive and transmit FIFO buffers are not cleared automatically, even if the SSE bit is set to 0. This allows the transmit buffer to be preloaded with the required data before the module operation is enabled.

The DR register bit assignments are given in Table 408.

Table 405 – DR register format

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…0	DATA	Received data, 32 bits (read). Transmit data, 32 bits (write). If the selected data word length is less than 32 bits, data must be right-aligned before being written to the SSPDR register. The transmitter block ignores unused bits. Received data words are automatically right-aligned by the receiver block.

23.14.2.4. **SR Register**

Status register

The SR register is read-only and contains information about the state of the receive and transmit FIFO buffers and about whether the SSP module is busy.

The SR register bit assignments are given in Table 406.

Table 406 – SR register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
15…5	-	Reserved. Read value is undefined. When writing, this bit must be written as 0.
4	BSY	Module busy flag: 0 – SSP module is idle; 1 – SSP module is currently transmitting and/or receiving data, or the transmit FIFO buffer is not empty
3	RFF	Receive FIFO buffer full: 0 – not full; 1 – full

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
2	RNE	Receive FIFO buffer not empty: 0 – empty; 1 – not empty
1	TNF	Transmit FIFO buffer not full: 0 – full; 1 – not full
0	TFE	Transmit FIFO buffer empty: 0 – not empty; 1 – empty

23.14.2.5. **CPSR register**

Clock divider register

The CPSR register is used to configure the clock divider factor. The written value must be an integer in the range from 2 to 254. The least significant bit of the register is forced to zero. If an odd number is written to the CPSR register, a subsequent read returns that number with the least significant bit cleared to zero.

The CPSR register bit assignments are given in Table 407.

Table 407 – CPSR register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
15…8	-	Reserved. Read value is undefined. When writing, this bit must be written as 0.
7…0	CPSDVSR	Clock division factor. The written value must be an integer in the range from 2 to 254. The least significant bit of the register is forced to zero

23.14.2.6. **IMSC Register**

Interrupt mask set/clear register

When read, this register returns the current mask value. When written, it sets or clears the mask for the corresponding interrupt. Writing 1 to a bit enables the corresponding interrupt; writing 0 disables it.

After reset, all bits of the mask register are cleared to 0.

The IMSC register bit assignments are given in Table 408.

Table 408 – IMSC register

Bit number		Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…7			Reserve. Do not modify. Reading returns zeros
6		TNBSYIM	Interrupt mask for transmit shift register empty: 1 – not masked; 0 – masked
5		TFEIM	Interrupt mask for transmit FIFO empty: 1 – not masked; 0 – masked
4		RNEIM	Interrupt mask for data present in receive FIFO: 1 – not masked; 0 – masked
3		TXIM	Interrupt mask for transmit FIFO buffer is 50% full or less: 1 – not masked; 0 – masked
2		RXIM	Interrupt mask for receive FIFO buffer is 50% full or less: 1 – not masked; 0 – masked
1		RTIM	Interrupt mask for receiver timeout (receive FIFO buffer not empty and no read attempts within the timeout interval): 1 – not masked; 0 – masked
0		RORIM	Interrupt mask for receive FIFO overrun: 1 – not masked; 0 – masked

23.14.2.7. **RIS Register**

Interrupt status register

This register is read-only and contains the current interrupt status before masking is applied. Any data written to this register are ignored.

The RIS register bit assignments are given in Table 409.

Table 409 – RIS register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…7		Reserve. Do not modify. Reading returns zeros
6	TNBSYRIS	Status before the SSPTNBSYINTR interrupt masking
5	TFERIS	Status before the SSPTFEINTR interrupt masking
4	RNERIS	Status before the SSPRNEINTR interrupt masking
3	TXRIS	Status before the SSPTXINTR interrupt masking
2	RXRIS	Status before the SSPTXINTR interrupt masking
1	RTRIS	Status before the SSPRTINTR interrupt masking
0	RORRIS	Status before the SSPRORINTR interrupt masking

23.14.2.8. **MIS Register**

Masked interrupt status register

The MIS register is read-only and contains the current interrupt status after masking is applied. Any data written to this register are ignored.

The MIS register bit assignments are given in Table 410.

Table 410 – MIS register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…7		Reserve. Do not modify. Reading returns zeros
6	TNBSYMIS	Status of masked SSPTNBSYINTR interrupt
5	TFEMIS	Status of masked SSPTFEINTR interrupt
4	RNEMIS	Status of masked SSPRNEINTR interrupt
3	TXMIS	Status of masked SSPTXINTR interrupt
2	RXMIS	Status of masked SSPRXINTR interrupt
1	RTMIS	Status of masked SSPRTINTR interrupt
0	RORMIS	Status of masked SSPRORINTR interrupt

23.14.2.9. **ICR Register**

Interrupt clear register

The ICR register is write-only and is used to clear an interrupt flag for a given event by writing 1 to the corresponding bit. Writing 0 to any bit in this register is ignored.

The ICR register bit assignments are given in Table 411.

Table 411 – ICR Register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
15…2	-	Reserved. When writing, this bit must be written as 0.
1	RTIC	Clear SSPRTINTR interrupt
0	RORIC	Clear SSPRORINTR interrupt

23.14.2.10. **DMACR register**

Direct memory access control register

The DMACR register is readable and writable. After reset, all bits in this register are cleared to 0.

The DMACR register bit assignments are given in Table 412.

Table 412 – DMACR Register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
15…2	-	Reserved. Reads return 0. When writing, this bit must be written as 0.
1	TXDMAE	DMA usage for transmission: 1 – DMA request generation for servicing the transmit FIFO buffer is enabled; 0 – DMA request generation for servicing the transmit FIFO buffer is disabled
0	RXDMAE	DMA usage for reception: 1 – DMA request generation for servicing the receive FIFO buffer is enabled; 0 – DMA request generation for servicing the receive FIFO buffer is disabled

24 UART controller

The Universal Asynchronous Receiver-Transmitter (UART) module is a peripheral device of the integrated circuit.

The controller includes a codec (ENDEC – Encoder/Decoder) for the serial interface of infrared data transmission in accordance with the SIR (Serial Infra Red) protocol of the Infrared Data Association (IrDA).

24.1. Key characteristics of the UART module

The UART module can be programmed either as a universal asynchronous receiver-transmitter or for infrared data communication (SIR).

The module contains independent receive (16 words by 13 bits) and transmit (16 words by 9 bits) FIFO (First In First Out) buffers, which reduces the interrupt load on the central processor.

FIFO can be disabled by software, limiting the buffer size to a single word.

The baud rate can be configured by dividing the clock frequency of the reference generator in the range (1×16 – 65535×16). Fractional clock division factors are allowed, which makes it possible to use any reference generator with a frequency above 3.6864 MHz.

The module supports standard elements of the asynchronous communication protocol – start and stop bits, as well as a parity bit, which are added before transmission and removed after reception.

Interrupts from the transmit FIFO buffer, the receive FIFO buffer, the receiver timeout, modem status changes, and error detection can be masked independently.

The UART module supports the use of the DMA controller to organize data transfers and reduce the load on the integrated circuit core.

The module also supports detection of false start bits. Generation and detection of a line break condition are supported.

Hardware flow control can be implemented. The fully programmable UART has the following characteristics:

- data length of 5, 6, 7, 8, or 9 bits;
- generation and parity check (the parity bit can be set for even or odd parity, has a fixed value, or not transmitted);
- generation of 1 or 2 stop bits;
- baud rate up to UART_CLK/16 baud.

The IrDA SIR infrared data codec ensures:

- software selection between data transfer over the asynchronous receiver-transmitter lines and over the IrDA SIR infrared codec lines;
- support for operation at bit rates up to 115,200 bit/s in half-duplex mode;
- support for bit duration in normal mode (3/16) and in low-power mode (1.41–2.23 μs);
- programmable division of the UART_CLK reference frequency to obtain the required bit duration in low-power mode.

24.2. **Programmable parameters**

The following key parameters can be configured in software:

- baud rate – integer and fractional parts;
- number of data bits;
- number of stop bits;
- parity mode;
- enable or disable the FIFO buffers (data queue depth – 16 entries or one entry, respectively);
- interrupt threshold for FIFO fill level (1/8, 1/4, 1/2, 3/4, and 7/8);
- internal clock generator frequency (nominal value 1.8432 MHz) can be set in the range from 1.42 to 2.12 MHz to allow generation of shortened bit durations in low-power mode (for IR communication);
- hardware flow-control mode.

24.3. **Differences from the 16C650 UART controller**

The controller differs from the industry-standard 16C650 asynchronous receiver-transmitter in the following characteristics:

- receive FIFO interrupt thresholds: 1/8, 1/4, 1/2, 3/4, and 7/8;
- transmit FIFO interrupt thresholds: 1/8, 1/4, 1/2, 3/4, and 7/8;
- different internal register address map and bit assignments;
- modem status signals cannot be modified.

The following 16C650 controller capabilities are not supported:

- 1.5 stop-bit duration (only 1 or 2 stop bits are supported);

– independent configuration of receiver and transmitter clock frequencies.

24.4. **Functional capabilities**

The unit provides the following functions:

– Conversion of data received from a peripheral device from serial to parallel format;
– Conversion of data transmitted to a peripheral device from parallel to serial format.

Data, control information, and module status information are read and written by the CPU. Transmit and receive paths are buffered by internal FIFO memory that can store up to 16 words of 13 bits in receive mode and up to 16 words of 9 bits in transmit mode.

The transceiver module:

– contains a programmable generator that forms the clock signal for both transmit and receive data from the internal UART_CLK clock signal;
– provides functionality comparable to the industry-standard UART 16C650 controller;
– supports data exchange at the following maximum rates:
– in IrDA mode – up to 460,800 bit/s;
– in low-power IrDA mode – up to 115,200 bit/s.

The transceiver operating mode and baud rate are controlled by the line control register LCR_H and by the integer (IBRD) and fractional (FBRD) baud-rate divisor registers.

The unit can generate:

– independent maskable interrupts from the receiver (including receive timeout), the transmitter, modem status changes, and error detection;
– a common interrupt asserted when any of the independent unmasked interrupts is active;
– DMA request signals for use with the DMA controller.

If a framing error, parity error, or break condition occurs, the corresponding error bit is set and stored together with the data in the FIFO buffer. If the buffer overruns, the corresponding overrun bit is set immediately, and writes into the receive FIFO are blocked.

It is possible to programmatically limit the FIFO size to a single byte, which allows implementation of a conventional asynchronous serial interface with double buffering.

The following modem-status input lines are supported: Clear To Send (CTS), Data Carrier Detect (DCD), Data Set Ready (DSR), and Ring Indicator (RI), as well as the output lines Request to Send (RTS) and Data Terminal Ready (DTR).

Hardware flow control over the nUARTCTS and nUARTRTS lines is supported.

The serial infrared interface block for data transmission according to the IrDA SIR protocol implements the ENDEC data-exchange protocol. When it is enabled, data exchange is performed not via UARTTXD and UARTRXD, but via nSIROUT and SIRIN.

In this mode the unit drives UARTTXD to the passive state (logic high) and stops responding to modem status changes and to the signal on UARTRXD. The SIR ENDEC protocol supports data exchange only in half-duplex mode, that is, transmission is not possible during reception and reception is not possible during transmission.

In accordance with the IrDA SIR physical-layer specification, the delay between transmit and receive must be at least 10 ms.

24.5. **Description of UART operation**

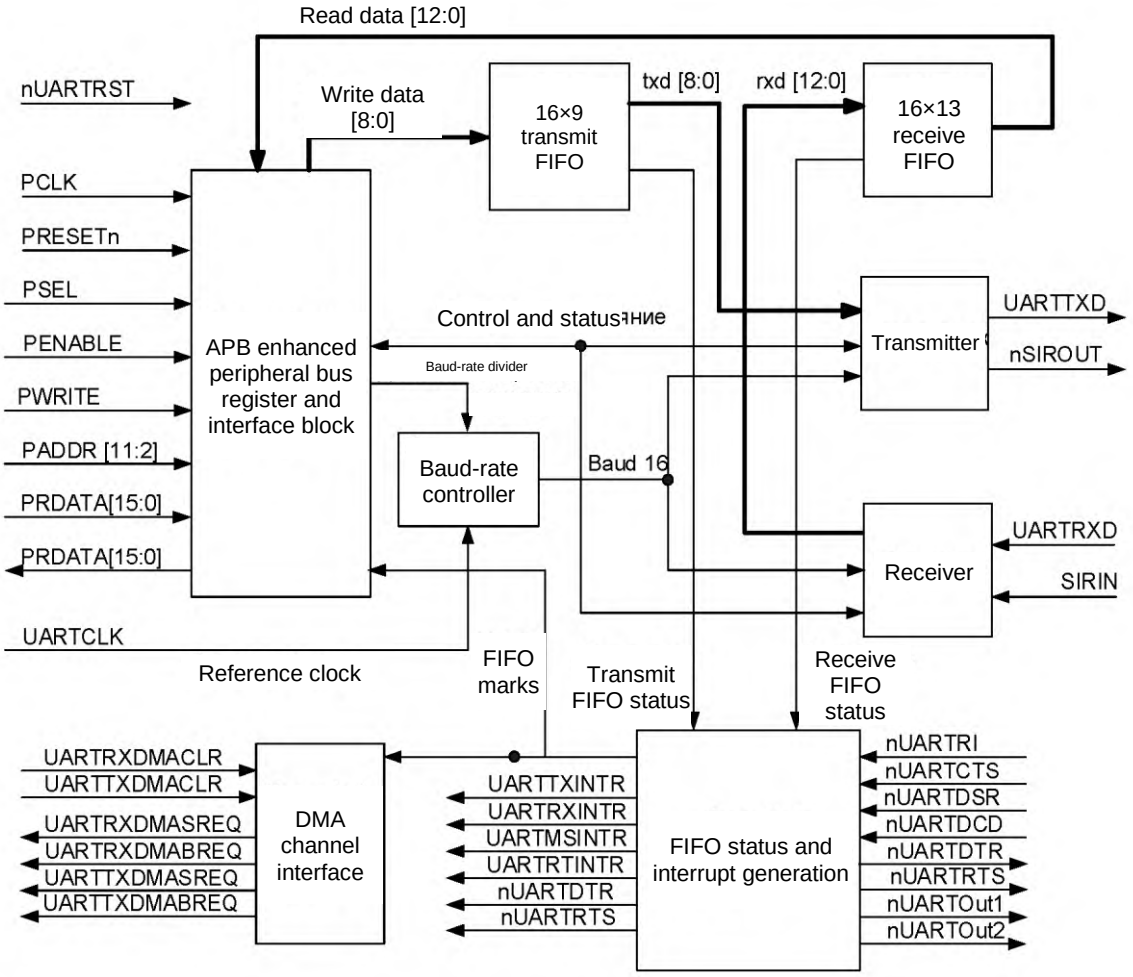


Figure 85 – Block diagram of the universal asynchronous receiver-transmitter (UART)

24.5.1. **Transceiver clock generator**

The generator contains counters without a reset circuit that form the internal clock signals Baud16 and IrLPBaud16.

The Baud16 signal is used to synchronize the control circuits of the serial data exchange receiver and transmitter. It is a pulse train with a width of one UART_CLK period and with a frequency 16 times higher than the data rate.

The IrLPBaud16 signal is intended to synchronize the pulse-generating circuit that generates the pulse width required for infrared data exchange in low-power mode.

24.5.2. **Transmit FIFO buffer**

The transmit buffer is 9 bits wide, 16 words deep, and organized as a FIFO (First In, First Out). Data written by the central processor over the APB bus are stored in the buffer until they are read by the transmit-data logic. The transmit FIFO can be disabled; in this case it operates as a single-byte buffer register.

24.5.3. **Receive FIFO buffer**

The receive buffer is 13 bits wide, 16 words deep, and organized as a FIFO (First In, First Out). Data received from the peripheral device and the associated error codes are stored in it by the receive-data logic until they are read by the central processor over the APB bus. The receive FIFO buffer can be disabled; in this case it operates as a single-byte buffer register.

24.5.4. **Transmitter block**

The transmitter logic converts data read from the transmit buffer from parallel to serial format. The control logic outputs a serial bit stream in the following order: start bit, data bits beginning with the least significant bit, parity bit, and then stop bits, in accordance with the configuration programmed in the control register.

24.5.5. **Receiver block**

The receiver logic converts data received from the peripheral device from serial to parallel format after detecting a valid start pulse. In addition, checks are performed for receive buffer overrun, parity errors, framing errors, and line break. Error indications are also stored in the output buffer.

24.5.6. **Interrupt generation block**

The controller generates independent maskable active-high interrupts. In addition, a combined interrupt is generated by OR-ing these independent interrupt signals.

The combined interrupt signal may be connected to the system interrupt controller, providing an additional level of masking for the device as a whole and simplifying implementation of modular device drivers.

Alternatively, the individual interrupt request signals from the transceiver may be routed to the system interrupt controller. In that case, the handling procedure can read data for all interrupt sources in a single pass. This approach is useful when access time to peripheral registers is significantly shorter than the CPU clock period in a real-time system.

For more details see the “Interrupts” subsection.

24.5.7. **Direct Memory Access interface**

The module provides an interface to the DMA controller in accordance with the interaction scheme between the transceiver and the DMA controller.

24.5.8. **Synchronization block and registers**

The controller supports both asynchronous and synchronous operation of the CPU_CLK and UART_CLK clock generators. The synchronization registers and acknowledgment logic are permanently active. This has negligible impact on device characteristics. Control signals are synchronized in both directions of the data flow, that is, from the CPU_CLK domain to the UART_CLK domain and back.

24.6. **IrDA SIR codec operation**

A block diagram of the codec is shown in Figure 86.

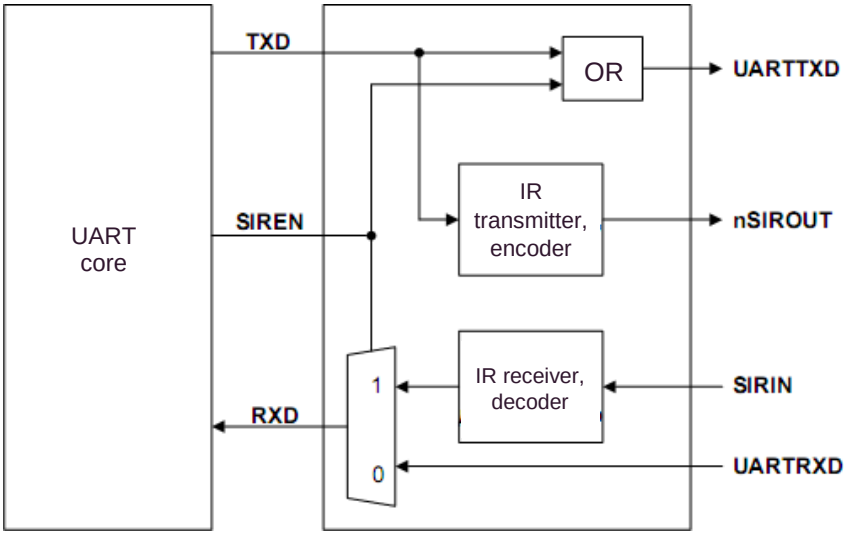


Figure 86 – IrDA codec block diagram

24.6.1. **IR transmitter encoder**

The encoder converts the data stream from the asynchronous transmitter output, which uses non-return-to-zero (NRZ) modulation. The physical-layer specification of the IrDA SIR protocol requires return-to-zero inverted (RZI) modulation, in which transmission of a logical zero corresponds to emission of a single infrared optical pulse. The resulting pulse stream is applied to an amplifier and then to an IR LED.

In IrDA mode, pulse duration is specified as 3 periods of the internal Baud16 clock, i.e., 3/16 of the bit transmission time.

In low-power IrDA mode, the pulse width is defined as 3/16 of the bit time at a baud rate of 115200 bit/s. This is implemented by generating three periods of the IrLPBaud16 clock with a nominal frequency of 1.8432 MHz, obtained by dividing the UART_CLK frequency. The IrLPBaud16 frequency is set by writing the corresponding division factor to the ILPR register.

The encoder output is active low. When a logical one is transmitted, the encoder output remains low, when a logical zero is transmitted, a pulse is generated and the output briefly goes high.

In both normal and low-power IrDA modes, the use of fractional baud-rate divisors increases jitter on the data pulse edges. With fractional divisors, jitter arises because intervals between Baud16 clock pulses are irregular: over time, a Baud16 period contains a varying number of UART_CLK periods. In the worst case, jitter in the IR pulse stream can reach three UART_CLK periods. According to the IrDA SIR specification, jitter must not exceed 13%. If the UART_CLK frequency is greater than 3.6834 MHz and the baud rate is less than or equal to 115,200 bit/s, jitter does not exceed 9%. Thus the standard requirements are satisfied.

24.6.2. **IR receiver decoder**

The decoder converts the return-to-zero data stream from the IR signal receiver into a non-return-to-zero data stream and applies it to the UART receiver input. In the inactive state, the decoder input is high. The encoder output signal has polarity opposite to the decoder input.

The start bit is detected when the decoder input goes low.

Note: To prevent false UART triggering by noise pulses, the SIRIN input ignores pulses with a duration less than:

- 3/16 of the Baud16 period in IrDA mode;
- 3/16 of the IrLPBaud16 period in low-power IrDA mode.

24.7. **UART operation description**

24.7.1. **Module reset**

The transceiver and the codec can be reset by the common processor reset signal. Register values after reset are described in the subsection “Software control of the module.”

24.7.2. **Clock signals**

The UART_CLK frequency F_{UART_CLK} must support the required data-rate range.

$$F_{UART\_CLK}(\text{min}) \geq 16 \cdot \text{baud\_rate\_max};$$

$$F_{UART\_CLK}(\text{max}) \leq 16 \cdot 65535 \cdot \text{baud\_rate\_min}.$$

For example, to support data rates from 110 to 460,800 baud, F_{UART_CLK} must be in the range from 7.3728 to 115.34 MHz.

The F_{UART_CLK} frequency must also be selected so that the configured baud rate meets the specified accuracy requirements.

There is also a constraint on the ratio between F_{CPU_CLK} and F_{UART_CLK}. F_{UART_CLK} must not be more than 5/3 of F_{CPU_CLK}.

$$F_{UART\_CLK} \leq \frac{5}{3} \cdot F_{CPU\_CLK}.$$

For example, in UART mode with a maximum baud rate of 921,600 baud at F_{UART_CLK} = 14.7456 MHz, the F_{CPU_CLK} frequency must be at least 8.85276 MHz. This ensures that the UART controller has enough time to write received data to the FIFO buffer.

24.7.3. **Universal asynchronous receiver-transmitter operation**

Control information is stored in the line control register LCR. This register is 30 bits wide internally, but is accessed externally over the APB bus through the following registers:

- LCR_H – defines:
- data transfer parameters;
- word length;
- buffering mode;
- number of stop bits;
- parity mode;
- break-signal generation;
- IBRD – defines the integer part of the baud-rate divisor;
- FBRD – defines the fractional part of the baud-rate divisor.

Note: The three registers can be updated correctly in two ways:

- write IBRD, write FBRD, write LCR_H;
- write FBRD, write IBRD, write LCR_H.

To change only one of the registers (IBRD or FBRD), the following sequence must be used:

- write IBRD (or FBRD), write LCR_H.

24.7.4. **Clock division factor**

The baud-rate divisor is 22 bits wide, with 16 bits allocated to the integer part and 6 bits to the fractional part. Support for fractional divisors makes it possible to exchange data using standard baud rates up to 230,400 while using a UART_CLK clock of arbitrary frequency above 3.6864 MHz.

The integer part of the divisor is written to the 16-bit IBRD register. The 6-bit fractional part is written to the FBRD register. The divisor value is related to these register contents as follows:

$$\text{Divisor} = \frac{F_{UART\_CLK}}{16 \cdot \text{data rate}} \quad (26)$$

$$= IBRD + FBRD,$$

where *IBRD* is the integer part of the divisor

FBRD is the fractional part.



Figure 87 – Divisor

The 6-bit value written to the FBRD register is calculated by taking the fractional part of the required divisor, multiplying it by 64 (2^n, where n is the FBRD width), and rounding to the nearest integer:

$$m = \text{integer}(\text{FBRD} \cdot 2^n + 0{,}5\,, \quad (27)$$

where *integer* discards the fractional part, n = 6.

Inside the module an internal Baud16 signal is generated, it is a pulse train with a pulse width equal to one UART_CLK period and an average frequency 16 times higher than the required data rate.

24.7.5. **Transmission and reception of data**

Transmit and receive data is stored in 16-entry FIFO buffers, each entry of the receive FIFO buffer stores one data byte plus four modem-status bits.

Transmit data is written into the transmitter FIFO buffer. If the transceiver is enabled, transmission of a frame starts with the parameters specified in the LCR_H line control register. Transmission continues until the transmitter FIFO buffer is empty. After a word is written to the transmitter FIFO buffer, the BUSY signal goes high. It remains high for the entire duration of data transmission. BUSY returns low only after the transmitter FIFO buffer becomes empty and the last data bit (including stop bits) has been sent. The BUSY signal may remain high even if the transceiver is disabled.

For each data bit (in the receive line), three samples of the level are taken, and the bit value is determined by majority voting.

If the receiver is inactive (the UARTRXD input line is held at logic 1) and the input signal transitions from high to low (a start bit is detected), a counter clocked by Baud16 is started, samples of the receiver input are then registered every eight Baud16 ticks (in asynchronous transceiver) mode, or every four Baud16 ticks (in IR data exchange mode). The higher sampling rate in IR mode is required to correctly handle the data pulses according to the IrDA SIR protocol.

The start bit is considered valid if the UARTRXD signal remains at low logic for eight Baud16 samples after the counter is started. Otherwise, the transition to 0 is treated as a false start and ignored.

If a valid start bit is detected, the receiver begins registering data at the receiver input. Each subsequent data bit is sampled every 16 Baud16 ticks (the duration of one bit time). All data bits (according to the configured parameters) and the parity bit (if parity checking is enabled) are registered.

Finally, the presence of a valid stop bit (logic high on UARTRXD) is checked. If this condition is not met, a framing-error flag is set. After a complete data word has been received, it is written to the receive FIFO buffer together with four error-status bits associated with that word.

24.7.6. **Error bits**

Three error-indication bits associated with a received data symbol are stored in bit positions [11:9] of the data word in the receiver FIFO buffer. There is also a FIFO-overrun error indicator stored in bit position 12 of the data word.

Table 413 shows the assignment of all data word bits in a receiver FIFO buffer.

Table 413 – Assignment of data-word bits in the receiver FIFO buffer:

FIFO buffer bit	Assignment
12	Buffer overrun indicator
11	Line break error
10	Parity check error
09	Frame generation error
08…00	Received data

24.7.7. **Buffer overrun bit**

The overrun bit is not directly associated with a specific symbol in the receiver buffer. Overrun is latched if the receiver FIFO buffer is full at the moment when the next data symbol has been completely received (and is stored in the shift register). In this case the data in the shift register is not written to the receiver buffer and are lost when reception of the next symbol begins. As soon as a free location appears in the receiver buffer, the next received data symbol is written to the FIFO buffer together with the current value of the overrun indication. After the data word has been successfully written, the overrun indication is cleared.

24.7.8. **FIFO buffer disable**

The receiver and transmitter FIFO buffers can be disabled. In this case the UART receiver and transmitter each have only a single-byte buffer register. The overrun bit is then set when a new data symbol has already been received but the previous symbol has not yet been read.

In this implementation the FIFO buffers are not physically disconnected, the required functionality is achieved by logical handling of flags. When the FIFO buffer is disabled and the transmitter shift register is empty (not in use), a written data byte is placed directly into the shift register, bypassing the buffer register.

24.7.8.1. **Loopback test**

Loopback testing (connecting the transmitter output to the receiver input) is performed by setting bit LBE in the controller control register CR to 1.

24.7.9. **Operation of the IrDA SIR infrared data codec**

The codec connects the asynchronous data stream produced by the receiver-transmitter to the half-duplex IrDA SIR serial interface. No analog signal processing is performed. The codec generates a digital data stream for the asynchronous receiver input and processes the digital data stream from the transmitter output.

Two operating modes are provided.

- IrDA mode – a logical zero level is driven on the nSIROUT line as a high-level pulse with a duration of 3/16 of the selected data bit period. A logical one is transmitted as a constant low level. The resulting output signal is then applied to the infrared transmitter, which emits an optical pulse for each zero bit. On the receive side, the optical pulses drive the base of the IR-receiver phototransistor, which in turn produces a low logic level. This results in a low level at the SIRIN input.
- Low-power IrDA mode – the duration of the transmitted IR pulses is set to three periods of the internal reference signal IrLPBaud16 (1.63 μs at the nominal frequency of 1.8432 MHz). This mode is enabled by setting bit SIR_LP in the control register CR.

In both IrDA mode and low-power IrDA mode:

- encoding is performed based on data bits generated by the module's asynchronous transmitter;
- during reception, the decoded bits are further processed by the asynchronous receiver block.

According to the IrDA SIR physical-layer specification, data exchange must be half-duplex, and the delay between transmission and reception must be at least 10 ms. This delay is implemented in software. It is required because the IR transmit LED can affect the nearby IR receiver, which may distort the received signal or even drive the receive path into saturation. The delay between the end of transmission and the start of reception is referred to as receiver latency or receiver settling (readiness) time.

The IrLPBaud16 signal is generated by dividing the UART_CLK frequency by a factor written to the ILPR register.

The division factor is calculated using the formula

$$\text{Division factor} = \frac{F_{UART\,CLK}}{IrLPBaud16,} \tag{28}$$

where the nominal value of *IrLPBaud16* is 1.8432 MHz. The division factor must be chosen so that the following condition holds:

$$1.48\ \text{MHz} < IrLPBaud16 < 2.12\ \text{MHz}. \tag{29}$$

24.7.9.1. **Loopback test**

Loopback testing is performed by setting bit LBE in the controller control register CR to 1 and simultaneously setting bit SIRTEST in the test-control register TCR to 1.

In this mode, the data driven on nSIROUT output must be routed back to the SIRIN input.

Note: This is the only case in which the test register is used in normal operation mode of the module.

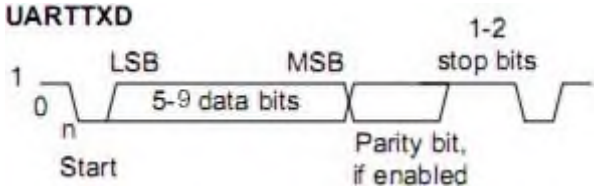


Figure 88 – Data frame

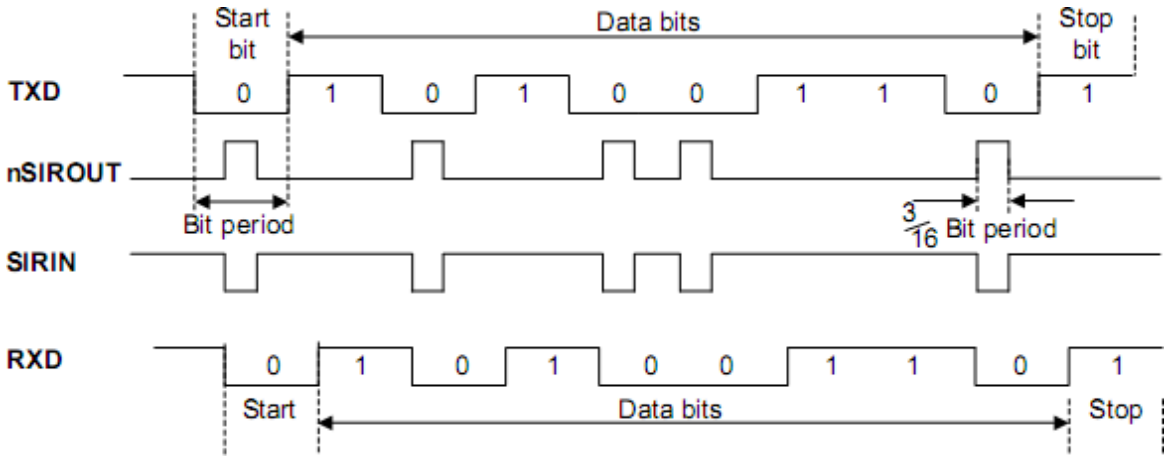


Figure 89 – IrDA data modulation

24.8. **Modem control lines**

The universal asynchronous receiver-transmitter module can be used in both data terminal equipment (DTE) and data communications equipment (DCE) modes.

Table 414 – Modem control assignments in DTE and DCE modes

Signal	Assignment	
	Data terminal equipment mode	Data communication equipment mode
nUARTCTS	Ready to send data	Request to send data
nUARTDSR	Data source ready	Data terminal ready
nUARTDCD	Data signal detected	-
nUARTRI	Ring indicator	-
nUARTRTS	Request to send data	Ready to send data
nUARTDTR	Data terminal ready	Data source ready
nUARTOut1	-	Data signal detected
nUARTOut2	-	Ring indicator

24.8.1. **Hardware flow control**

A software-enabled hardware flow control mode allows the data exchange to be paused and resumed using the nUARTRTS and nUARTCTS signals. An example of the interaction between two serial devices with hardware flow control is shown in Figure 102.

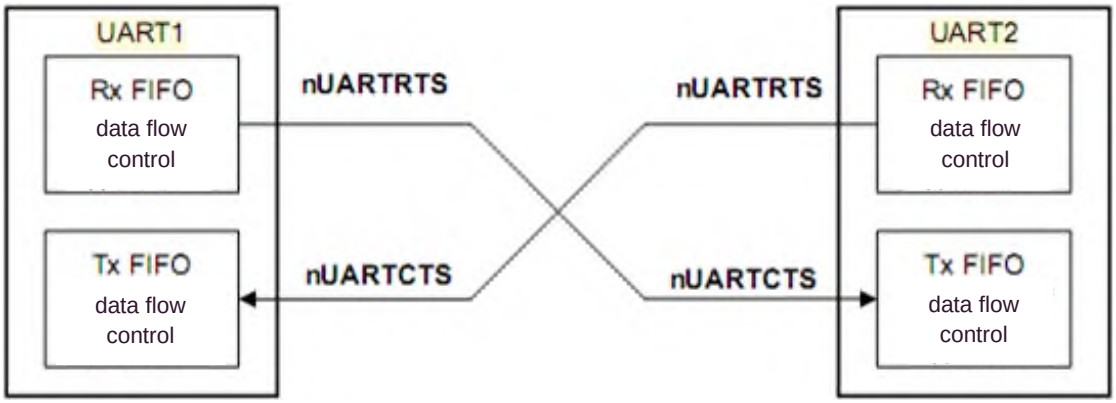


Figure 90 – Interaction between two serial devices with hardware flow control

If RTS-based flow control is enabled, the nUARTRTS line is driven to the active state only after the receive FIFO buffer has the configured number of free entries.

If CTS-based flow control is enabled, data transmission is performed only after the nUARTCTS line has gone to the active state.

The hardware flow control mode is configured by setting the RTSEn and CTSEn bits in the control register CR. Table 428 shows the required settings for the different flow control modes.

Table 415 – Flow control modes

CTSEn	RTSEn	Description
1	1	Flow control enabled on CTS and RTS
1	0	Flow control performed using CTS
0	1	Flow control performed using RTS
0	0	Flow control disabled

Note: If a mode with RTS-based flow control is selected, software cannot use the RTS bit in the CR register to check the state of the RTS line.

24.8.2. **RTS-based hardware flow control**

RTS flow control logic uses the receive FIFO buffer fill level threshold data. If the RTS-based flow control mode is selected, the nUARTRTS signal is driven active only after the receive FIFO buffer has the configured number of free entries. When the receive buffer reaches the threshold fill level, the nUARTRTS signal is deasserted (driven to the inactive state), indicating that there is no free space left to store received data. In this case transmission of further data must be stopped after the current character has been fully transmitted.

The nUARTRTS signal returns to the active state after data has been read from the receive FIFO buffer in an amount sufficient to bring the fill level below the threshold.

If RTS-based flow control is disabled while the UART transceiver is enabled, reception continues until the receive FIFO buffer is completely full or the data transmission is finished.

24.8.3. **CTS-based hardware flow control**

If a mode with CTS-based flow control is selected, the transmitter checks the state of the nUARTCTS line before starting transmission of each next data byte. Transmission is performed only if this line is active, and continues as long as the line remains active and the transmit buffer is not empty.

If the nUARTCTS line goes inactive, the module completes transmission of the current character and then stops sending data.

If CTS-based flow control is disabled while the UART transceiver is enabled, data is transmitted until the transmit FIFO buffer is empty.

24.9. **Direct Memory Access interface**

The universal asynchronous receiver-transmitter module provides an interface for connection to a DMA controller. Operation in this mode is controlled by the DMACR DMA control register.

The DMA interface includes the following signals:
For reception:

- UARTRXDMASREQ – single-symbol transfer request, initiated by the UART controller. The symbol size in receive mode is up to 13 bits. The signal is driven active if the receive FIFO buffer contains at least one symbol.
- UARTRXDMABREQ – block transfer request, initiated by the transceiver module. The signal is driven active if the receive FIFO buffer fill level exceeds the configured threshold. The threshold is configured individually for each FIFO buffer by writing a value to the IFLS register.
- UARTRXDMACLR – DMA request clear, initiated by the transceiver module to clear a previously issued request. If a block transfer was requested, the clear signal is generated while transferring the last data symbol in the block.

For transmission:

- UARTTXDMASREQ – single-symbol transfer request, initiated by the transceiver module. The symbol size in transmit mode is up to 9 bits. The signal is driven active when the transmit FIFO buffer contains at least one free location.
- UARTTXDMABREQ – block transfer request, initiated by the transceiver module. The signal is driven active if the transmit FIFO buffer fill level is below the configured threshold. The threshold is configured individually for each FIFO buffer by writing a value to the IFLS register.
- UARTTXDMACLR – DMA request clear, initiated by the DMA controller to clear a previously issued request. If a block transfer was requested, the clear signal is generated while transferring the last data symbol in the block.

Block and single-word transfer requests are not mutually exclusive and can be asserted simultaneously. For example, if the receive FIFO buffer fill level exceeds the threshold, both the single-word and block transfer request signals are asserted. If the amount of data in the receive buffer is below the threshold, only the single-word transfer request is asserted. This is useful when the total data volume is smaller than the block size. Assume, for instance, that 19 symbols must be received and the FIFO buffer threshold is configured to four. In this case, the DMA controller performs four block transfers of four characters each, and the remaining three characters are transferred in three single-word transfers.

Note: For these remaining three symbols the UART controller cannot initiate a block transfer.

Each DMA request asserted by the transceiver remains active until it is cleared by the corresponding DMACLR signal.

After the clear signal is deasserted, the transceiver can again generate a DMA request when the conditions described above are met. All DMA requests are cleared when operation of the transceiver is disabled, as well as when the DMA control bits TXDMAE or RXDMAE in the DMA control register DMACR are cleared to zero.

If the FIFO buffers are disabled, the device can only transmit and receive single symbols, so the controller can initiate DMA only in single-transfer mode. In this case the module can generate only the DMA control signals UARTRXDMASREQ and UARTTXDMASREQ. For information on disabling the FIFO buffers see the description of the line control register LCR_H.

When the FIFO buffers are enabled, data transfer can be performed using both single transfers and block transfers, depending on the configured buffer threshold and the actual fill level. Table 416 lists parameters for triggering block data transfer requests UARTRXDMABREQ and UARTTXDMABREQ as a function of the FIFO fill level.

Table 416 – Parameters for triggering block data transfer requests in DMA mode

Threshold level	Data block length	
	Transmit buffer (number of empty locations)	Receive buffer (number of filled locations)
1/8	14	2
¼	12	4
½	8	8
¾	4	12
7/8	2	14

The DMA control register DMACR contains the DMAONERR bit, which disables receiver DMA if the error interrupt line UARTEINTR is active. In this case, the corresponding receiver DMA request lines: UARTRXDMASREQ and UARTRXDMABREQ are made inactive (masked) until UARTEINTR is cleared. The UARTEINTR state does not affect the DMA request lines that serve the transmitter.

Figure 91 shows timing diagrams for single-word and block DMA requests, including the effect of the DMACLR signal. All signals must be synchronized to CPU_CLK. For clarity, it is assumed that DMA request signals are not resynchronized inside the DMA controller.

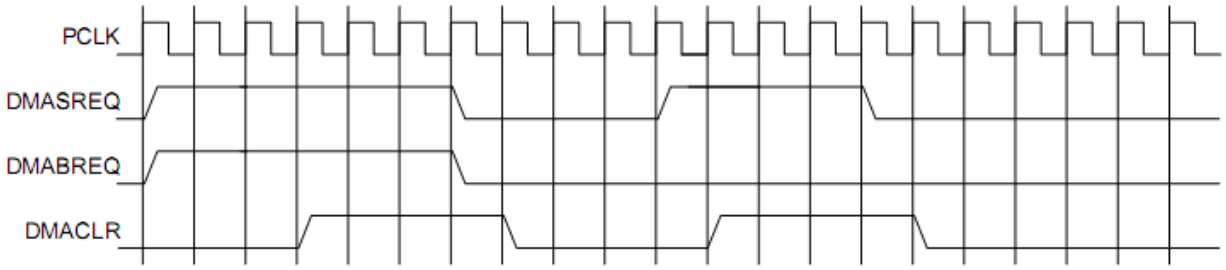


Figure 91 – Timing diagrams for single-word and block DMA requests

24.10. **Interrupts**

The module provides 11 maskable interrupt sources. These are combined into a single interrupt signal, which is a logical OR of the individual interrupt sources.

Interrupt request signals:

- UARTRXINTR – receiver interrupt;
- UARTTXINTR – transmitter interrupt;
- UARTRNEINTR – receive FIFO not empty;
- UARTTFEINTR – transmit FIFO empty;
- UARTTNBSYINTR – transmit shift register empty;
- UARTRTINTR – receiver timeout interrupt;
- UARTMSINTR – modem-status interrupt:
- UARTRIINTR, change of nUARTRI line state;
- UARTCTSINTR, change of nUARTCTS line state;
- UARTDCDINTR, change of nUARTDCD line state;
- UARTDSRINTR, change of nUARTDSR line state.
- UARTEINTR – error interrupt:
- UARTOEINTR, FIFO overrun;
- UARTBEINTR, break detected during reception;
- UARTPEINTR, parity error;
- UARTFEINTR, framing error.
- UARTINTR – logical OR of UARTRXINTR, UARTTXINTR, UARTRTINTR, UARTMSINTR, UARTEINTR, UARTRNEINTR, UARTTFEINTR, and UARTTNBSYINTR.

Each individual interrupt request signal can be masked by setting the corresponding bit in the UARTIMSC interrupt mask register. Setting a bit to 1 enables the corresponding interrupt; setting it to 0 disables it.

Availability of both individual interrupt lines and the combined interrupt line makes it possible to handle interrupts either through a system-wide handling procedure or using a modular device driver.

The receiver and transmitter interrupts UARTRXINTR and UARTTXINTR are output separately from interrupts related to changes in the device state. This allows request signals UARTRXINTR and UARTTXINTR to be used to coordinate data reads and writes with reaching the configured threshold levels of the receive and transmit FIFO buffers.

The error interrupt UARTEINTR is generated when any receive error condition occurs. Several conditions are defined for setting the error flag.

The modem-status interrupt is a combination of change indicators for the individual modem status lines.

The UARTRNEINTR interrupt is asserted when the receive FIFO accepts at least one data word and thus becomes non-empty.

The UARTTFEINTR interrupt is asserted when the transmit FIFO contains no data, that is, when it is currently empty.

UARTTNBSYINTR is asserted when the transmit shift register is empty.

The status of each interrupt condition can be read either from the raw interrupt status register UARTRIS or from the masked interrupt status register UARTMIS.

24.10.1. **UARTMSINTR**

The modem-status interrupt is generated when any modem status line (nUARTCTS, nUARTDCD, nUARTDSR, nUARTRI) changes state. The interrupt is cleared by writing 1 to the corresponding bit in the interrupt clear register ICR, selected according to which modem status line caused the interrupt.

24.10.2. **UARTRXINTR**

The receiver interrupt state can change when one of the following events occurs:

– The FIFO buffer is enabled and its fill level reaches the configured threshold. In this case, the interrupt line goes high. The interrupt signal goes low after data is read from the receive buffer until its fill level drops below the threshold, or after the interrupt is cleared.

– The FIFO buffer is disabled (its size is one character) and one character has been received. In this case, the interrupt line goes high. The interrupt signal goes low after one byte of data is read, or after the interrupt is cleared.

24.10.3. **UARTTXINTR**

The transmitter interrupt state can change when one of the following events occurs:

– The FIFO buffer is enabled and its fill level is less than or equal to the configured threshold. In this case, the interrupt line goes high. The interrupt signal goes low after data is written into the transmit buffer until its fill level rises above the threshold, or after the interrupt is cleared.
– The FIFO buffer is disabled (its size is one character) and there is no data in the transmit buffer register. In this case, the interrupt line goes high. The interrupt signal goes low after one byte of data is written, or after the interrupt is cleared.

To load data into the transmit FIFO, data must be written either before enabling the transceiver and interrupts, or after the transceiver and interrupts are enabled.

Note: The transmitter interrupt is generated on a threshold crossing, not on the transmit FIFO buffer fill level relative to the threshold. If the module and its interrupts are enabled before any data is written into the transmit FIFO buffer, the interrupt is not generated. The interrupt is generated only when the transmit FIFO buffer becomes empty.

24.10.4. **UARTRTINTR**

The receiver timeout interrupt is asserted when the receive FIFO buffer is not empty and no new data has arrived at the receiver input for a time interval sufficient to transmit 32 bits. The timeout interrupt is cleared either after data is read from the receive buffer until it becomes empty (or after one byte is read if the FIFO buffer is disabled), or by writing 1 to the corresponding bit in the interrupt clear register ICR.

24.10.5. **UARTEINTR**

The error-detection interrupt is asserted when a receive error occurs. It can be caused by several factors:

– a framing error;
– a parity error;
– a line break;
– a buffer overrun.

The cause of the interrupt is determined by reading the raw interrupt status register RIS or the masked interrupt status register MIS.

The interrupt is cleared by writing the corresponding bits in the interrupt clear register ICR. Bits 7 through 10 are assigned to the error-detection interrupts.

24.10.6. **UARTINTR**

All interrupt request signals described above are combined into a single line by logically OR-ing UARTRXINTR, UARTTXINTR, UARTRTINTR, UARTMSINTR, UARTEINTR, UARTRNEINTR, UARTTFEINTR, and UARTTNBSYINTR, taking masking into account. The combined output can be connected to the system interrupt controller, which allows additional masking of requests at the peripheral-device level.

24.11. **Software control of the module**

The following information applies to all controller registers:

- The controller base address is fixed. The offset of each register relative to the base address is constant.
- Reserved or unused addresses must not be accessed. Such accesses can lead to unpredictable module behavior.
- Except where explicitly stated otherwise in this specification:
- Undefined register bits are not to be modified;
- Undefined register bits are not to be used;
- All register bits (unless explicitly stated otherwise) are set to 0 after a power-on reset or system reset.
- The “Type” column in Table 417 defines the register access type with the following notation:
- RW – read/write;
- RO – read-only;
- WO – write-only.

24.12. **Description of UART controller registers**

Information on the UART controller registers is given in Table 417.

Table 417 – Summary of device registers

Base Address	Name				Description
0x50008000	UART1				UART1 interface controller registers
0x50010000	UART2				UART2 interface controller registers
0x50038000	UART3				UART3 interface controller registers
0x500E8000	UART4				UART4 interface controller registers
Offset	Name	Type	Reset value	Size, bits	Description
0x000	DR	RW	0x---	13/9	Data register
0x004	RSR_ECR	RW	0x0	4/0	Receive status/error clear register
0x008-0x014					Reserved
0x018	FR	RO	0b-10010---	9	Flag register
0x01C					Reserved
0x020	ILPR	RW	0x00	8	Low-power IrDA control register
0x024	IBRD	RW	0x0000	16	Integer part of the baud rate divider
0x028	FBRD	RW	0x00	6	Fractional part of the baud rate divider
0x02C	LCR_H	RW	0x00	9	Line control register
0x030	CR	RW	0x0300	16	Control register
0x034	IFLS	RW	0x12	6	Interrupt FIFO level select register
0x038	IMSC	RW	0x000	14	Interrupt mask register
0x03C	RIS	RO	0x300-	14	Interrupt status register
0x040	MIS	RO	0x00-	14	Masked interrupt status register
0x044	ICR	WO	-	11	Interrupt clear register
0x048	DMACR	RW	0x00	3	DMA control register
0x080	TCR	RW	0x00	3	Test control register

24.12.1. **Data register DR**

During data transmission:

If the transmit FIFO is enabled, the data word written to this register is placed in the transmit FIFO.

Otherwise, the written data word is latched in the transmit buffer register (the last element of the FIFO).

A write to the register starts data transmission. The data word is preceded by a start bit, followed by a parity bit (if parity checking is enabled), and a stop bit. The resulting word is sent to the transmit data line

During data reception:

If the receive FIFO is enabled, the data byte and four status bits (break, framing error, parity error, overrun) are stored in a 12-bit FIFO entry.

Otherwise, the data byte and status bits are written to the buffer register (the last element of the FIFO).

Received data bytes from the communication line are read by accessing the UART_DR register to get the received data together with the associated status bits. Status information can also be obtained by reading the RSR_ECR register (Table 418).

Table 418 – UARTDR register format

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
15…13	-	Reserved
12	OE	Receive buffer overrun. Set to 1 if data arrives at the receiver input while the buffer is full. Cleared to 0 after space becomes available in the buffer.
11	BE	Line break. Set to 1 when a break condition is detected, that is, when a low logic level is present at the receiver input for longer than the time required to transmit a complete data word (including start, stop, and parity bits). If the FIFO is enabled, this error is associated with the last character received by the FIFO. If a line break is detected, only a single zero character is loaded into the FIFO, data reception resumes only after the line returns to logic 1 and a valid start bit is subsequently detected.
10	PE	Parity error. Set to 1 if the parity of the received data character does not match the configuration of the EPS and SPS bits in the line control register LCR_H. If the FIFO is enabled, this error is associated with the last character received by the FIFO.
9	FE	Framing error. Set to 1 if a valid stop bit (a valid stop bit is 1) is not detected in the received character. If the FIFO is enabled, this error is associated with the last character received by the FIFO.
8…0	DATA	Received data (read) Transmit data (write)

Note: The transceiver must be disabled before any reprogramming of its control registers. If the transceiver is disabled while transmitting or receiving a character, it completes the current operation before stopping.

24.12.2. **Receive status/error clear register RSR_ECR**

The receiver status can also be read from the RSR register. In this case, the status indications for line break, parity error, and framing error relate to the last character read from the DR data register. By contrast, the buffer overrun flag is set as soon as this condition occurs (and is not tied to the last data byte read from the DR register).

A write to the ECR register clears the overrun, parity, framing, and line break error flags. In addition, all of these flags are cleared to 0 after a device reset.

The bit assignments of the RSR_ECR register are given in Table 419.

Table 419 – RSR_ECR Register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
7…4	-	Reserved. Read value is undefined.
3	OE	Receive buffer overrun. Set to 1 if data arrives at the receiver input while the buffer is full. Cleared to 0 after a write to the ECR register. The contents of the buffer remain valid, because only the shift register is overwritten. The CPU must read out data to free space in the FIFO buffer.
2	BE	Line break. Set to 1 when a break condition is detected, that is, when a low logic level is present at the receiver input for longer than the time required to transmit a complete data word (including start, stop, and parity bits). Cleared to 0 after a write to the ECR register. If the FIFO is enabled, this error is associated with the character currently at the top of the FIFO. If a line break is detected, only a single zero character is loaded into the FIFO, data reception resumes only after the line returns to logic 1 and a valid start bit is subsequently detected.
1	PE	Parity error. Set to 1 if the parity of the received data character does not match the configuration of the EPS and SPS bits in the line control register LCR_H. Cleared to 0 after a write to the ECR register. If the FIFO is enabled, this error is associated with the character currently at the top of the FIFO.
0	FE	Framing error. Set to 1 if a valid stop bit (a valid stop bit is 1) is not detected in the received character. Cleared to 0 after a write to the ECR register. If the FIFO is enabled, this error is associated with the character currently at the top of the FIFO.

Note: Before reading the RSR status register, the data received from the line must be read by accessing the DR data register. The reverse order is not allowed, because the RSR register updates its status only after the DR register is read. Receiver status information can also be obtained directly from the DR data register.

24.12.3. **Flag register FR**

After reset, the TXFF, RXFF, and BUSY bits in the flag register are cleared to 0, and the TXFE and RXFE bits are set to 1. The bit assignments in this register are given in Table 420.

Table 420 – FR Register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
15…9		Reserve. Do not modify. Read as 0
8	RI	Inversion of the nUARTRI line
7	TXFE	Transmit FIFO buffer empty. The bit value depends on the state of the FEN bit in the LCR_H line control register. If the FIFO buffer is disabled, the bit is set to 1 when the transmitter holding register is empty. Otherwise, it is 1 when the transmit FIFO buffer is empty. This bit does not provide any information about data in the transmitter shift register.
6	RXFF	Receive FIFO buffer full. The bit value depends on the state of the FEN bit in the LCR_H line control register. If the FIFO buffer is disabled, the bit is set to 1 when the receiver holding register is occupied. Otherwise, it is 1 when the receive FIFO buffer is full.
5	TXFF	Transmit FIFO buffer full. The bit value depends on the state of the FEN bit in the LCR_H line control register. If the FIFO buffer is disabled, the bit is 1 when the transmitter buffer register is occupied. Otherwise, it is 1 when the transmit FIFO buffer is full.
4	RXFE	Receive FIFO buffer empty. The bit value depends on the state of the FEN bit in the LCR_H line control register. If the FIFO buffer is disabled, the bit is set to 1 when the receiver holding register is empty. Otherwise, it is 1 when the receive FIFO buffer is empty.
3	BUSY	UART busy. Set to 1 when the controller is transmitting data on the line. The bit remains set until all data, including stop bits, have been completely transmitted. In addition, the busy bit is set to 1 whenever the transmit FIFO buffer contains data, regardless of the transceiver enable state (even if it is disabled).
2	DCD	Inversion of the nUARTDCD line
1	DSR	Inversion of the nUARTDSR line
0	CTS	Inversion of the nUARTCTS line

24.12.4. **Low-power IR transfer control register ILPR**

This 8-bit read/write register holds the value of the UART_CLK division factor used to generate the IrLPBaud16 clock signal. The bit assignments in the register are shown in Table 421.

The required value of the division factor for generating the IrLPBaud16 signal is calculated as

$$\text{ILPDVSR} = \frac{F_{UART\_CLK}}{F_{IrLPBaud16}}, \quad (30)$$

where the nominal IrLPBaud16 frequency $F_{IrLPBaud16}$ is 1.8432 MHz.

The division factor must be chosen so that the following condition is satisfied:

$$1.48 \text{ MHz} < F_{IrLPBaud16} < 2.12 \text{ MHz}, \quad (31)$$

which in turn ensures that the codec generates data pulses with a duration of (1.41–2.11) μs (three times the period of the IrLPBaud16 signal).

Table 421 – ILPR Register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
7…0	ILPDVSR	Clock division factor for UART_CLK used to generate the IrLPBaud16 clock signal. After reset, it is set to 0. Note: Factor of 0 is an invalid value. If it is set to 0, IrLPBaud16 pulses are not generated.

Note: To suppress noise, in low-power IrDA mode the codec ignores pulses at the SIRIN input whose duration is shorter than three IrLPBaud16 clock periods.

24.12.5. **IBRD integer baud rate division register**

The bit assignment of this register is given in Table 422.

Table 422 – IBRD Register

Bit number		Bit functional name	Explanation of the bit functional name, brief description o and possible values
15…0		BAUDDIV_INT	Integer part of the clock division factor used to generate t baud rate clock. After reset, it is set to 0

24.12.6. **FBRD fractional baud rate division register**

The bit assignment of this register is given in the table below.

Table 423 – BFRD Register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
5…0	BAUDDIV_FRAC	Fractional part of the clock division factor used to generate the transmit baud rate clock. After reset, it is set to 0

The division factor is calculated using the formula

$$\text{BAUDDIV} = \frac{F_{UART\_CLK}}{16 \cdot Baud\_rate}, \quad (32)$$

where FUART_CLK – UART controller clock frequency;
Baud_rate – required baud rate (in bit/s).

The BAUDDIV division factor consists of an integer and a fractional part – BAUDDIV_INT and BAUDDIV_FRAC, respectively.

Note: Changes to the contents of the IBRD and FBRD registers take effect only after transmission and reception of the current data character have completed.

The minimum allowed division factor is 1, and the maximum is 65535 (2^{16} – 1). Thus, an IBRD value of 0 is invalid, and in this case the FBRD register value is ignored.

Similarly, if IBRD equals 65535 (0xFFFF), the FBRD value cannot be greater than zero. Violating this condition causes receive or transmit operations to stop.

Example. Division factor calculation

Let the required baud rate be 230,400 bit/s and the UART clock frequency F_{UART_CLK} = 4 MHz. Then

$$\text{Division factor} = \frac{4 \cdot 10^6}{16 \cdot 230{,}400} = 1.085.$$

Thus, BRDI = 1, BRDF = 0.085.

Therefore, the value written to the BFRD register is

$$\text{m} = \text{integer}((0.085 \cdot 64) + 0.5) = 5.$$

Actual division factor = 1 + 5/64 = 1.078.

Actual baud rate = $(4 \cdot 10^6)/(16 \cdot 1.078)$ = 231,911 bit/s.

Data-rate setting error = (231,911 – 230,400)/230,400 • 100% = 0.656%.

The maximum baud rate setting error when using the 6-bit BFRD register is 1/64 • 100% = 1.56%. This error occurs when m = 1, with the difference accumulating over 64 bit periods.

Table 424 lists division-factor values for typical data rates when F_{UART_CLK} = 7.3728 MHz. With these parameters, the fractional part of the division factor is not used, so the FBRD register must be written with zero.

Table 424 – Division factors for F_{UART_CLK} = 7.3728 MHz

Division factor	Data rate
0x0001	460800
0x0002	230400
0x0004	115200
0x0006	76800
0x0008	57600
0x000C	38400
0x0018	19200
0x0020	14400
0x0030	9600
0x00C0	2400
0x0180	1200
0x105D	110

Table 425 gives division factor values for typical data rates when F_{UART_CLK} = 4 MHz.

Table 425 – Division factors for F_{UART_CLK} = 4 MHz

Integer part	Fractional part	Required data rate	Actual data rate	Error, %
0x001	0x05	230400	231911	0.656
0x002	0x0B	115200	115101	0.086
0x003	0x10	76800	76923	0.160
0x006	0x21	38400	38369	0.081
0x011	0x17	14400	14401	0.007
0x01A	0x03	9600	9598	0.021
0x068	0x0B	2400	2400	~0

24.12.7. **LCR_H line control register**

This register provides access to bits 29 through 22 of the LCR register. After reset, all bits in the LCR_H register are cleared to 0.

The bit assignment of the register is described in Table 426.

Table 426 – LCR_H Register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
15…9		Reserved. Do not modify. Reading returns zeros
8	WLEN[2]	Word length – number of data bits transmitted or received in a frame: 0 – word length is defined by WLEN[1:0]; 1 – 9 bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
7	SPS	Transmission of a parity bit with a fixed value: 0 – disabled; 1 – the parity bit position carries the inverted value of the EPS bit, and that value is also checked on receive. (If EPS = 0, a 1 is transmitted in the parity bit position; if EPS = 1, a 0 is transmitted). The SPS bit value has no effect if parity generation and checking is disabled by the PEN bit (Table 440).
6…5	WLEN[1:0]	Word length – number of data bits transmitted or received in a frame: 0b11 – 8 bits, 0b10 – 7 bits, 0b01 – 6 bits, 0b00 – 5 bits.
4	FEN	Enable receive and transmit FIFO buffers: 0 – disabled; 1 – enabled.
3	STP2	Two-stop-bit mode: 0 – one stop bit; 1 – two stop bits. The receiver does not verify the presence of an additional stop bit in the frame
2	EPS	Odd/even: 0 – the parity bit makes the number of 1s in the data portion of the frame odd; 1 – it makes the number of 1s even. The EPS bit value has no effect if parity generation and checking is disabled by the PEN bit (Table 440).
1	PEN	Parity enable: 0 – the frame does not contain a parity bit; 1 – a parity bit is transmitted in the frame and checked on receive (Table 440).
0	BRK	Line break. If this bit is set to 1, then after transmission of the current character completes, the UART_TXD output is driven low. To perform this operation correctly, software must hold the break signal on the line for at least the time required to transmit two data frames. In normal operation this bit must be set to 0.

The contents of the LCR_H, IBRD, and FBRD registers together form the combined 31-bit LCR register, which is updated on a strobe generated by a write to LCR_H. Thus, for changes to the baud rate division factor to take effect, after updating the IBRD and/or FBRD registers, a write to the LCR_H register must be performed.

Table 427 provides the truth data for the parity-control bits SPS, EPS, and PEN of the Line Control Register LCR_H.

Table 427 – Parity control mode

PEN	EPS	SPS	Parity control bit
0	X	X	Not transmitted, not checked
1	1	0	Even parity check of the data word
1	0	0	Odd parity check of the data word
1	0	1	Parity bit is always 1
1	1	1	Parity bit is always 0

Notes

- 1 The LCR_H, IBRD, and FBRD registers must not be modified:
  - while the transceiver is enabled;
  - while reception or transmission is being completed during transceiver shutdown (when switching it to the disabled state).
- 2 Data integrity in the FIFO buffers is not guaranteed in the following cases:
  - after the break bit BRK is set;
  - if the software has stopped the transceiver while data is present in the FIFO buffers and then enabled it again.

24.12.8. **Control Register CR**

After reset, all control register bits, except for bits 9 and 8, are set to logic 0. Bits 9 and 8 are set to logic 1.

The control register bit assignments are shown in Table 428.

Table 428 – Control register CR

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
15	CTSEn	Enable CTS-based flow control: 1 – enabled, data is transmitted on the line only when the nUARTCTS signal is active; 0 – disabled
14	RTSEn	Enable RTS-based flow control: 1 – enabled, a request for data from the external device is issued only when there is free space in the receive FIFO buffer; 0 – disabled
13	Out2	Inversion of the signal on the modem status line nUARTOut2. In Data Terminal Equipment (DTE) mode, this line can be used as the “ring indicator” (RI). 1 – signal enabled; 0 – signal disabled
12	Out1	Inversion of the signal on the modem-status line nUARTOut1. In Data Terminal Equipment (DTE) mode, this line can be used as the “data carrier detect” (DCD) line. 1 – signal enabled; 0 – signal disabled

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
11	RTS	Inversion of the signal on the modem-status line nUARTRTS: 1 – signal enabled; 0 – signal disabled
10	DTR	Inversion of the signal on the modem-status line nUARTDTR: 1 – signal enabled; 0 – signal disabled
9	RXE	Receive enable. 1 – receiver operation enabled. Reception is performed either via the asynchronous serial interface or via the SIR infrared interface, depending on the SIREN bit value. If the transceiver is disabled while data is being received, it completes reception of the current character before stopping. 0 – receiver operation disabled
8	TXE	Transmit enable. 1 – transmitter operation enabled. Transmission is performed either via the asynchronous serial interface or via the SIR infrared interface, depending on the SIREN bit value. If the transceiver is disabled while data is being transmitted, it completes transmission of the current character before stopping. 0 – transmitter operation disabled
7	LBE	Loopback test mode: 1 – loopback enabled; 0 – disabled. When loopback is enabled: If SIREN = 1 and the SIRTEST bit of the Test Control Register TCR is set to 1, the signal from the nSIROUT output of the codec is inverted and applied to the SIRIN input of the codec. The SIRTEST bit is set to 1 in order to force the device out of the half-duplex mode inherent to the SIR interface. After loopback testing is finished, the SIRTEST bit must be cleared to 0. If SIRTEST = 0, the UART_TXDx transmitter output line is switched to the UART_RXDx receiver input line. In both SIR and UART modes, the modem status output lines are switched to their corresponding input lines. After reset, this bit is set to 0.
6…3		Reserve. Do not modify. Reading returns zeros
2	SIRLP	Selection of low-power IrDA mode: 1 – data pulse width is equal to three cycles of the IrLPBaud16 signal, regardless of the selected data rate. Selecting this mode reduces power consumption, but may shorten the communication range; 0 – data pulse width is equal to 3/16 of the bit time.

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
1	SIREN	Enable operation of the IrDA SIR infrared codec: 1 – infrared codec operation enabled. Data is transmitted on the nSIROUT output and received on the SIRIN input. The UART_TXDx line is high. Data on the UART_RXDx input and on the modem status lines is ignored. If UARTEN = 0, the value of this bit has no effect. 0 – disabled. The nSIROUT signal is held low, and data on the SIRIN input is ignored.
0	UARTEN	Enabling the transceiver: 0 – operation disabled; Before stopping, reception and/or transmission of the character currently in progress is completed; 1 – operation enabled. Data is exchanged either over the asynchronous interface lines or over the SIR infrared interface lines, depending on the SIREN bit state.

Note: To enable data transmission, bits TXE and UARTEN must be set to 1. Similarly, to enable data reception, bits RXE and UARTEN must be set to 1.

The recommended sequence for programming the control registers is as follows:

1. Stop the transceiver;
2. Wait for reception and/or transmission of the current character to complete;
3. Clear the transmitter buffer by clearing bit FEN of the LCR_H register to 0;
4. Modify the CR register settings;
5. Restart the transceiver.

24.12.9. **Interrupt FIFO Level Select Register IFLS**

This register is used to set the FIFO fill-level thresholds for the transmitter and receiver at which the UARTTXINTR and UARTRXINTR interrupt signals, respectively, are generated. The interrupt is generated when the FIFO fill level crosses the configured threshold.

After reset, the register is set to a threshold corresponding to one half of the bugger depth. The register format and bit values are shown in Table 429.

Table 429 – IFLS Register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
15…6		Reserve. Do not modify. Reading returns zeros

5…3	RXIFLSEL	Receive FIFO interrupt level threshold:
		b000 = buffer 1/8 full;
		b001 = Buffer 1/4 full;
		b010 = Buffer 1/2 full;
		b011 = Buffer 3/4 full;
		b100 = Buffer 7/8 full;
		b101–b111 = reserved

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values

2…0	TXIFLSEL	Transmit buffer interrupt level threshold:
		b000 = buffer 1/8 full;
		b001 = Buffer 1/4 full;
		b010 = Buffer 1/2 full;
		b011 = Buffer 3/4 full;
		b100 = Buffer 7/8 full;
		b101-b111 = reserved.
		Note that when the transmitter shift register is empty, the word written to the FIFO is immediately copied into the shift register. Therefore, to generate a transmitter interrupt event from the UART block, enough words must be written into the FIFO so that the fill level exceeds the configured threshold by at least one word, taking this behavior into account.

24.12.10. **Interrupt Mask Set/Clear Register IMSC**

When read, this register returns the current mask value. When written, it sets or clears the mask for the corresponding interrupt.

After reset, all bits of the mask register are cleared to 0. The bit assignment of the IMSC register is described in Table 430.

Table 430 – IMSC register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…14		Reserve. Do not modify. Reading returns zeros
13	TNBSYIM	Mask for transmitter shift register empty interrupt (UARTTNBSYINTR): 1 – mask set; 0 – mask cleared
12	TFEIM	Mask for the transmit FIFO empty interrupt (UARTTFEINTR): 1 – mask set; 0 – mask cleared
11	RNEIM	Mask for the receive FIFO not empty interrupt (UARTRNEINTR): 1 – mask set; 0 – mask cleared
10	OEIM	Mask for the receiver overrun interrupt (UARTOEINTR): 1 – mask set; 0 – mask cleared
9	BEIM	Mask for the break condition on the line interrupt (UARTBEINTR): 1 – mask set; 0 – mask cleared
8	PEIM	Mask for the parity error interrupt (UARTPEINTR): 1 – mask set; 0 – mask cleared

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
7	FEIM	Mask for the framing error interrupt (UARTFEINTR): 1 – mask set; 0 – mask cleared
6	RTIM	Mask for the receive timeout interrupt (UARTRTINTR): 1 – mask set; 0 – mask cleared
5	TXIM	Mask for the transmitter interrupt (UARTTXINTR): 1 – mask set; 0 – mask cleared
4	RXIM	Mask for the receiver interrupt (UARTRXINTR): 1 – mask set; 0 – mask cleared
3	DSRMIM	Mask for the UARTDSRINTR interrupt on a change of the nUARTDSR line: 1 – mask set; 0 – mask cleared
2	DCDMIM	Mask for the UARTDCDINTR interrupt on a change of the nUARTDCD line: 1 – mask set; 0 – mask cleared
1	CTSMIM	Mask for the UARTCTSINTR interrupt on a change of the nUARTCTS line: 1 – mask set; 0 – mask cleared
0	RIMIM	Mask for the UARTRIINTR interrupt on a change of the nUARTRI line: 1 – mask set; 0 – mask cleared

24.12.11. **Raw Interrupt Status Register RIS**

This register is read-only and contains the current interrupt status before masking is applied. Any data written to this register is ignored.

Warning. After reset, all register bits, except for the modem-status interrupt bits (bits 3 through 0), are cleared to 0. The value of the modem status interrupt bits after reset is undefined.

The RIS register bit assignments are given in Table 431.

Table 431 – RIS Register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…14		Reserve. Do not modify. Reading returns zeros
13	TNBSYRIS	Status of the transmitter shift register empty interrupt (UARTTNBSYINTR)
12	TFERIS	Status of the transmit FIFO empty interrupt (UARTTFEINTR)
11	RNERIS	Status of the receive FIFO not empty interrupt (UARTRNEINTR)

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
10	OERIS	Status of the overrun error interrupt (UARTOEINTR): 1 – receive buffer overrun; 0 – no receive FIFO overrun
9	BERIS	Status of the break condition interrupt (UARTBEINTR): 1 – a receive-line break has occurred; 0 – no receive-line break has occurred
8	PERIS	Status of the parity error interrupt (UARTPEINTR): 1 – a parity error has occurred; 0 – no parity error has occurred
7	FERIS	Status of the framing error interrupt (UARTFEINTR): 1 – a framing error has occurred; 0 – no framing error has occurred
6	RTRIS	Status of the receive timeout interrupt (UARTRTINTR)*: 1 – receive timeout has expired; 0 – receive timeout has not expired
5	TXRIS	Status of the transmitter interrupt (UARTTXINTR): 1 – a transmitter interrupt has occurred; 0 – no transmitter interrupt has occurred
4	RXRIS	Status of the receiver interrupt (UARTRXINTR): 1 – a receiver interrupt has occurred; 0 – no receiver interrupt has occurred
3	DSRRMIS	Status of the UARTDSRINTR interrupt on a change of the nUARTDSR line: 1 – an interrupt has occurred; 0 – no interrupt has occurred
2	DCDRMIS	Status of the UARTDCDINTR interrupt on a change of the nUARTDCD line: 1 – an interrupt has occurred; 0 – no interrupt has occurred
1	CTSRMIS	Status of the UARTDSRINTR interrupt on a change of the nUARTCTS line: 1 – an interrupt has occurred; 0 – no interrupt has occurred
0	RIRMIS	Status of the UARTRIINTR interrupt on a change of the nUARTRI line: 1 – an interrupt has occurred; 0 – no interrupt has occurred
* The timeout interrupt mask signal is used as an enable for entry into low-power mode. Therefore, reading the timeout interrupt status from the MIS and RIS registers returns the same result.		

24.12.12. **Masked Interrupt Status register MIS**

This register is read-only and contains the current interrupt status after masking is applied. Any data written to this register is ignored.

After reset, all register bits, except for the modem-status interrupt bits (bits 3 through 0), are cleared to 0. The value of the modem status interrupt bits after reset is undefined.

The MIS register bit assignments are given in Table 432.

Table 432 – MIS Register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…14		Reserve. Do not modify. Reading returns zeros
13	TNBSYMIS	Masked status of the transmitter shift register empty interrupt (UARTTNBSYINTR)
12	TFEMIS	Masked status of the transmit FIFO empty interrupt (UARTTFEINTR)
11	RNEMIS	Masked status of the receive FIFO not empty interrupt (UARTRNEINTR)
10	OEMIS	Masked status of the receive buffer overrun interrupt (UARTOEINTR): 1 – receive buffer overrun; 0 – no receive FIFO overrun
9	BEMIS	Masked status of the break on the receive line interrupt (UARTBEINTR): 1 – a receive-line break has occurred; 0 – no receive-line break has occurred
8	PEMIS	Masked status of the parity error interrupt (UARTPEINTR): 1 – a parity error has occurred; 0 – no parity error has occurred
7	FEMIS	Masked status of the framing error interrupt (UARTFEINTR): 1 – a framing error has occurred; 0 – no framing error has occurred
6	RTMIS	Masked status of the receive timeout interrupt (UARTRTINTR): 1 – receive timeout has expired; 0 – receive timeout has not expired
5	TXMIS	Masked status of the transmitter interrupt (UARTTXINTR): 1 – a transmitter interrupt has occurred; 0 – no transmitter interrupt has occurred
4	RXMIS	Masked status of the receiver interrupt (UARTRXINTR): 1 – a receiver interrupt has occurred; 0 – no receiver interrupt has occurred

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
3	DSRMMIS	Masked status of the UARTDSRINTR interrupt for a change on the nUARTDSR line: 1 – an interrupt has occurred; 0 – no interrupt has occurred
2	DCDMMIS	Masked status of the UARTDCDINTR interrupt for a change on the nUARTDCD line: 1 – an interrupt has occurred; 0 – no interrupt has occurred
1	CTSMMIS	Masked status of the UARTCTSINTR interrupt for a change on the nUARTCTS line: 1 – an interrupt has occurred; 0 – no interrupt has occurred
0	RIMMIS	Masked status of the UARTRIINTR interrupt for a change on the nUARTRI line: 1 – an interrupt has occurred; 0 – no interrupt has occurred

24.12.13. **Interrupt Clear register ICR**

This register is write-only and is used to clear an interrupt flag for a given event by writing 1 to the corresponding bit. Writing 0 to any bit in this register is ignored.

The bit assignment of the ICR register is described in Table 433.

Table 433 – ICR Register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…11		Reserve. Do not modify. Reading returns zeros
10	OEIC	Clear the receive buffer overrun interrupt (UARTOEINTR): 1 – clear the interrupt; 0 – no effect on the register state.
9	BEIC	Clear the break on the receive line interrupt (UARTBEINTR): 1 – clear the interrupt; 0 – no effect on the register state.
8	PEIC	Clear the parity error interrupt (UARTPEINTR): 1 – clear the interrupt; 0 – no effect on the register state.
7	FEIC	Clear the framing error interrupt (UARTFEINTR): 1 – clear the interrupt; 0 – no effect on the register state.
6	RTIC	Clear the receive timeout interrupt (UARTRTINTR): 1 – clear the interrupt; 0 – no effect on the register state.
5	TXIC	Clear the transmitter interrupt (UARTTXINTR): 1 – clear the interrupt; 0 – no effect on the register state.

4	RXIC	Clear the receiver interrupt (UARTRXINTR): 1 – clear the interrupt; 0 – no effect on the register state.
3	DSRMIC	Clear UARTDSRINTR interrupt for a change on the nUARTDSR line: 1 – clear the interrupt; 0 – no effect on the register state.
2	DCDMIC	Clear UARTDCDINTR interrupt for a change on the nUARTDCD line: 1 – clear the interrupt; 0 – no effect on the register state.
1	CTSMIC	Clear UARTCTSINTR interrupt for a change on the nUARTCTS line: 1 – clear the interrupt; 0 – no effect on the register state.
0	RIMIC	Clear UARTCTSINTR interrupt for a change on the nUARTCTS line: 1 – clear the interrupt; 0 – no effect on the register state.

24.12.14. **Direct Memory Access Control register DMACR**

The register is readable and writable. After reset, all bits in this register are cleared to 0.

The bit assignment of the DMACR register is described in Table 434.

Table 434 – DMACR Register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…3		Reserve. Do not modify. Reading returns zeros
2	DMAONERR	Blocking of DMA requests on reception when line errors occur: 1 – when an error detection interrupt occurs, DMA requests from the receiver UARTRXDMASREQ and UARTRXDMABREQ are blocked; 0 – DMA requests are not blocked.
1	TXDMAE	DMA usage for transmission: 1 – DMA request generation for servicing the transmit FIFO buffer is enabled; 0 – generation of DMA requests is disabled.
0	RXDMAE	DMA usage for reception: 1 – DMA request generation for servicing the receive FIFO buffer is enabled; 0 – generation of DMA requests is disabled.

24.12.15. **Test control register TCR**

The register is readable and writable. After reset, all bits in this register are cleared to 0.

The bit assignment of the ICR register is described in Table 435.

Table 435 – TCR Register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
15…13		Reserve. The read value is unpredicted.
12…3	-	Reserved
2	SIRTEST	Enables reception of data in ring mode from the IrDA transmitter output: 1 – enabled; 0 – disabled. Used together with setting the LBE bit in the CR register.
1	TESTFIFO	Enables reading data from the transmit FIFO and writing data to the receive FIFO: 1 – enabled; 0 – disabled
0	ITEN	Puts the UART controller into test mode: 1 – test mode enabled; 0 – test mode disabled.

25 Direct Memory Access (DMA) controller

25.1. Key features of the DMA controller

Key features and distinguishing characteristics:

- 32 DMA channels;
- each DMA channel has its own data transfer control signals;
- each DMA channel has a programmable priority level;
- each priority level is serviced according to the priority level defined by the DMA channel number;
- support for various data transfer types:
- memory-to-memory;
- memory-to-peripheral;
- peripheral-to-memory;
- support for different DMA cycle types;
- support for various data widths;
- each DMA channel has a primary and an alternate channel control data structure;
- all channel control data is stored in system memory;
- receiver data width is equal to transmitter data width;
- the number of transfers in one DMA cycle can be programmed from 1 to 1024;
- the address increment can be greater than the data width.

25.2. Terms and definitions

Table 436 – Terms and definitions

Alternate	Alternate channel control data structure. The corresponding register can be configured to select the data structure type (see section “Channel control data structure”).
C	Channel identifier for the direct memory access controller. For example: C=1 – DMA channel 1 C=23 – DMA channel 23
Channel	Controller configurations are possible with up to 32 channels. Each channel has independent data transfer control signals capable of initiating data transfer on the DMA channel.
Channel control data	The data structure is located in system memory. This data structure can be programmed so that the controller performs data transfers on the DMA channel in the required mode. The controller must have access to the region of system memory where this information resides. Note: Any mention of a data structure in this specification refers to channel control data.
DMA cycle	All DMA transfers that the controller must perform to move N data items.

DMA transfer	A single operation that moves one byte, halfword, or word.
N	Total number of DMA transfers that the controller performs for a channel.
Ping-pong	Operating mode for a selected channel in which the controller receives an initial request and then performs a DMA cycle using either the primary or the alternate data structure. After this DMA cycle is completed, the controller starts a new DMA cycle using the other (primary or alternate) data structure. The controller signals completion of each DMA cycle, which allows the main processor to reconfigure the inactive data structure. The controller continues to switch between the primary and alternate data structures until an invalid data structure is read or a cycle completes without switching to the other structure.
Primary	Primary channel control data structure. The controller uses this data structure if the corresponding bit in the chnl_pri_alt_set register is set to 0.
R	Power of 2 that defines the number of DMA transfers that may occur before arbitration changes. The number of DMA transfers is programmed from 1 to 1024 in binary steps from 2 to the power of 0 to 2 to the power of 10.
Execution with configuration change	Operating mode for a selected channel in which the controller receives a request from a peripheral and performs 4 DMA transfers using the primary control data structure, which configures the alternate control data structure. After that, the controller starts a DMA cycle using the alternate data structure. When this cycle is completed and the peripheral asserts a new service request, the controller again performs 4 DMA transfers using the primary control data structure, which once more reconfigures the alternate control data structure. After that, the controller starts a DMA cycle using the alternate data structure. The controller continues to operate in this manner until an invalid data structure is read or the processor sets the alternate data structure for a normal cycle. The controller sets the dma_done flag if this operating mode terminates after completion of a normal cycle.

25.3. Functional description

Figure 93 shows a simplified block diagram of the controller.

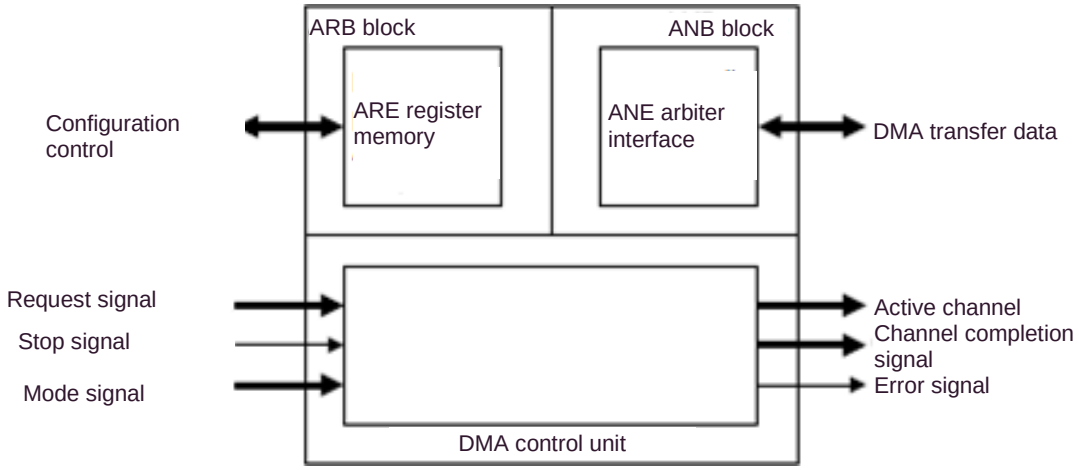


Figure 93 – Controller block diagram

The controller consists of the following main functional blocks:

– block connected to the APB bus;
– block connected to the AHB bus;
– DMA control block.

25.3.1. DMA channel assignment

Table 437 – DMA channel assignment

Request number	dma_req	dma_sreq	Description
0	UART1TXDMABREQ	UART1TXDMASREQ	Request from UART1 transmitter
1	UART1RXDMABREQ	UART1RXDMASREQ	Request from UART1 receiver
2	UART2TXDMABREQ	UART2TXDMASREQ	Request from UART2 transmitter
3	UART2RXDMABREQ	UART2RXDMASREQ	Request from UART2 receiver
4	SSP1TXDMABREQ	SSP1TXDMASREQ	Request from SSP1 transmitter
5	SSP1RXDMABREQ	SSP1RXDMASREQ	Request from SSP1 receiver
6	CRCDMABREQ	CRCDMASREQ	Request from hardware CRC computation block
7	UART3TXDMABREQ	UART3TXDMASREQ	Request from UART3 transmitter
8	UART3RXDMABREQ	UART3RXDMASREQ	Request from UART3 receiver
9	TMR3DMAREQ	TMR3DMAREQ	Request from general-purpose timer TIMER3
10	TMR1DMAREQ	TMR1DMAREQ	Request from general-purpose timer TIMER1
11	TMR2DMAREQ	TMR2DMAREQ	Request from general-purpose timer TIMER2
12	ADCUIDMABREQ1	ADCUIDMASREQ1	Requests from the ADC block for measuring voltages and currents in the power network
13	ADCUIDMABREQ2	ADCUIDMASREQ2	
14	ADCUIDMABREQ3	ADCUIDMASREQ3	
15	ADCUIDMABREQ4	ADCUIDMASREQ4	
16	ADCUIDMABREQ5	ADCUIDMASREQ5	
17	ADCUIDMABREQ6	ADCUIDMASREQ6	
18	ADCUIDMABREQ7	ADCUIDMASREQ7	

19	SSP2TXDMABREQ	SSP2TXDMASREQ	Request from SSP2 transmitter
20	SSP2RXDMABREQ	SSP2RXDMASREQ	Request from SSP2 receiver
21	SSP3TXDMABREQ	SSP3TXDMASREQ	Request from SSP3 transmitter
22	SSP3RXDMABREQ	SSP3RXDMASREQ	Request from SSP3 receiver
23	TMR4DMAREQ	TMR4DMAREQ	Request from general-purpose timer TIMER4
24	UART4TXDMABREQ	UART4TXDMASREQ	Request from UART4 transmitter
25	UART4RXDMABREQ	UART4RXDMASREQ	Request from UART4 receiver
26	-	-	-
27	-	-	-
28	-	-	-
29	-	-	-
30	-	ADCDMASREQ	Request from successive-approximation ADC
31	-	-	-

25.3.2. **Block connected to the APB bus**

This block contains a set of registers that configure the controller via the APB slave interface. The registers occupy a 4 KB address space.

25.3.3. **Block connected to the AHB bus**

The controller contains a single DMA bus master block that uses a 32-bit bus to transfer data from the source to the destination. The source and the destination are AHB bus slaves.

25.3.4. **DMA control block**

This block contains the control logic that implements the following functions:

- perform arbitration of incoming requests;
- indicate the active channel;
- indicate completion of a channel transfer;
- indicate an error condition on the DMA bus;
- allow slow devices to suspend execution of a DMA cycle;
- wait for a clear request until a DMA cycle completes;
- perform single or multiple DMA transfers for each request;
- perform the following types of DMA transfers:
- memory-to-memory;

– memory-to-peripheral;
– peripheral-to-memory.

25.3.5. **Transfer types**

The interface controller does not support burst transfers. The controller performs only single transfers. The absence of burst transfers has minimal impact on system performance, because burst transfers are more efficient in single-layer AHB systems where blocks must seize the bus or access external memory. At the same time, the DMA controller is intended for use in multilayer AHB systems that include on-chip memory.

25.3.6. **Data width of transfers**

The interface controller supports transfers with data widths of 8, 16, and 32 bits. The values of HSIZE bus combinations are given in Table 438.

Table 438 – HSIZE bus combinations

HSIZE[2]*	HSIZE[1]	HSIZE[0]	Data width (bits)
0	0	0	8
0	0	1	16
0	1	0	32
0	1	1	**
* The signal is held at logic 0. ** Forbidden combination.			

The controller always uses 32-bit transfers when accessing channel control data. The source data width must be set equal to the destination data width.

25.3.7. **Data protection control**

The controller supports configuration of AHB-Lite protocol data protection modes defined by the HPROT[3:1] bus. The following protection modes can be selected:

– cacheable;
– bufferable;
– privileged.

Table 439 gives the HPROT bus combinations

Table 439 – Data protection modes

HPROT[3] cacheable	HPROT[2] bufferable	HPROT[1] privileged	HPROT[0] data/command	Description
-	-	-	1*	Data access
-	-	0	-	User access
-	-	1	-	Privileged access
-	0	-	-	Non-bufferable
-	1	-	-	Bufferable
0	-	-	-	Non-cacheable
1	-	-	-	Cacheable

* The controller holds HPROT[0] at logic “1” to indicate data access.

For each DMA cycle, protection modes for source and destination transfers can be selected. For more details see subsection “Channel control data structure”.

For each DMA channel a data protection mode can also be selected. For more details see subsection “DMA control”.

25.3.8. **Address increment**

The controller allows configuration of the address increment when reading data from the source and when writing data to the destination. The address increment depends on the data width of the transfer. Possible combinations are given in Table 440.

Table 440 – Address Increment

Data width	Increment value
8	Byte, halfword, word
16	Halfword, word
32	Word

The minimum address increment always matches the data width of the transfer. The maximum address increment supported by the controller is one word. Further details on configuring the address increment are provided in subsection 25.5 “Channel control data structure”. This subsection describes the control bits for the address increment value in the channel control data.

Note: If the address must remain unchanged during read or write operations, for example when working with a FIFO, the controller can be configured for a fixed address (see section “Channel control data structure”).

25.4. DMA control

25.4.1. Data transfer rules

The controller applies the data transfer rules (see Table 454) when the following conditions are met:

- the DMA channel is enabled, which is achieved by setting the control bits chnl_enable_set[C] and master_enable to logic 1;
- the dma_req[C] and dma_sreq[C] request flags are not masked, which is achieved by setting the control bit chnl_req_mask_set[C] to logic 0;

Table 441 – Rules under which channel data transfers are allowed and requests are not masked

Rule	Description
1	If dma_active[C] is 0, then setting dma_req[C] or dma_sreq[C] to 1 for one or more HCLK cycles, consecutive or not, starts a transfer on channel C.
2	The controller sets only one dma_active[C] bit to 1.
3	The controller sets dma_active[C] to 1 at the start of a transfer on channel C.
4	For DMA cycle types other than the peripheral “Execution with configuration change” mode, dma_active[C] remains at 1 until the controller completes transfers with numbers less than 2^R or less than the transfer count specified in the n_minus_1 register. In the peripheral “Execution with configuration change” mode, dma_active[C] remains at 1 for each pair of DMA transfers using the primary and alternate channel control data structures. In this mode the controller performs 2^R transfers using the primary control data structure and then, without arbitration, performs transfers with numbers less than 2^R (or less than the transfer count specified in the n_minus_1 register) using the alternate control data structure. After the last transfer is completed, dma_active[C] is cleared to 0.
5	The controller keeps dma_active[C] at 0 for at least one HCLK cycle before setting dma_active[C] or any dma_active[] bit to 1 again.
6	For channels on which transfer is allowed, the controller sets only one dma_done[] bit to 1.
7	If dma_req[C] is set to 1 when dma_active[C] or dma_stall is also 1, this indicates that the controller has detected the request.
8	If the cycle_ctrl bits for the channel are set to 3’b100, 3’b101, 3’b110, or 3’b111, dma_done[C] is never set to 1.
9	If all transfers for the channel are complete and the cycle_ctrl bits allow dma_done[C] to be asserted, then on the falling edge of dma_active[] the following events occur: – if dma_stall is 0, the controller sets dma_done[] to 1 for one HCLK cycle; – if dma_stall is 1, controller operation is suspended. After dma_stall is cleared to 0, the controller sets dma_done[] to 1 for one HCLK cycle.
10	The dma_waitonreq[C] state can be changed only when the channel is disabled.
11	If dma_waitonreq[C] is 1, then dma_active[C] does not return to 0 until: the controller completes 2^R transfers (or the transfer count specified in the n_minus_1 register); dma_req[C] is cleared to 0; dma_sreq[C] is cleared to 0.

Rule	Description
12	If, during the HCLK cycle immediately before dma_active[C] would be cleared to 0, dma_stall is set to 1, then the controller clears dma_active[C] to 0 on the next HCLK cycle; the transfer on channel C does not complete until dma_stall is cleared to 0.
13	The controller ignores dma_sreq[C] if dma_waitonreq[C] is 0.
14	The controller ignores dma_sreq[C] if chnl_useburst_set[C] is 1*).
15	For DMA cycle types other than the peripheral "Execution with configuration change" mode, after completing 2^R transfers the controller sets chnl_useburst_set[C] to 0 if the number of remaining transfers is less than 2^R. In the peripheral "Execution with configuration change" mode the controller sets chnl_useburst_set[C] to 0 only if the number of remaining transfers using the alternate channel control data structure is less than 2^R.
16	For DMA cycle types other than the peripheral "Execution with configuration change" mode, if during the HCLK cycle before dma_active[C] is set to 1 the signals dma_sreq[C] and dma_waitonreq[C] are 1 and dma_req[C] is 0, the controller performs a single DMA transfer. In the peripheral "Execution with configuration change" mode, if during the HCLK cycle before dma_active[C] is set to 1 the signals dma_sreq[C] and dma_waitonreq[C] are 1 and dma_req[C] is 0, the controller performs 2^R transfers using the primary channel control data structure. It then performs one transfer using the alternate channel control data structure without arbitration.
17	For DMA cycle types other than the peripheral "Execution with configuration change" mode, if during the HCLK cycle immediately before dma_active[C] is set to 1 both dma_sreq[C] and dma_req[C] are 1, priority is given to dma_req[C], and the controller performs 2^R DMA transfers (or the number of transfers specified in the n_minus_1 register). In the peripheral "Execution with configuration change" mode, if during the HCLK cycle immediately before dma_active[C] is set to 1 both dma_sreq[C] and dma_req[C] are 1, priority is given to dma_req[C], and the controller performs 2^R transfers using the primary channel control data structure, then, without arbitration, performs transfers with numbers less than 2^R (or less than the number of transfers specified in the n_minus_1 register) using the alternate channel control data structure.
18	When chnl_req_mask_set[C] is 1, the controller ignores requests on dma_sreq[C] and dma_req[C].

*) These bits must be set with care. If the value specified in the n_minus_1 register is less than 2^R, the controller does not clear the chnl_useburst_set bits and, as a result, requests on dma_sreq[C] remain masked. If the peripheral does not set dma_req[C] to 1, the controller never performs the required transfers.

When a channel is disabled, the controller performs DMA transfers according to the rules given in Table 442.

Table 442 – Rules for performing DMA transfers on “disabled” channels

Rule	Description
19	If dma_req[C] is 1, the controller sets dma_done[C] to 1. This allows the controller to signal a ready request to the central processor even when the channel is disabled.
20	If dma_sreq[C] is 1, the controller sets dma_done[C] to 1 provided that dma_waitonreq[C] is 1 and chnl_useburst_set[C] is 0. This allows the controller to signal a ready request to the central processor even when the channel is disabled.
21	dma_active[C] is always held at 0.

25.4.2. **DMA controller operation diagrams**

This section describes the following examples of controller operation using the data transfer rules (see Table 441):

– pulse-triggered service request;
– level-triggered service request;
– completion flags;
– wait-for-service request flags.

Note: All diagrams shown in Figures 94-97 assume the following:

– ready is 1;
– the AHB slave always returns an “OKAY” response.

25.4.2.1. **Pulse-triggered service request**

Figure 94 shows the DMA controller timing diagram for a pulse-triggered request from a peripheral block.

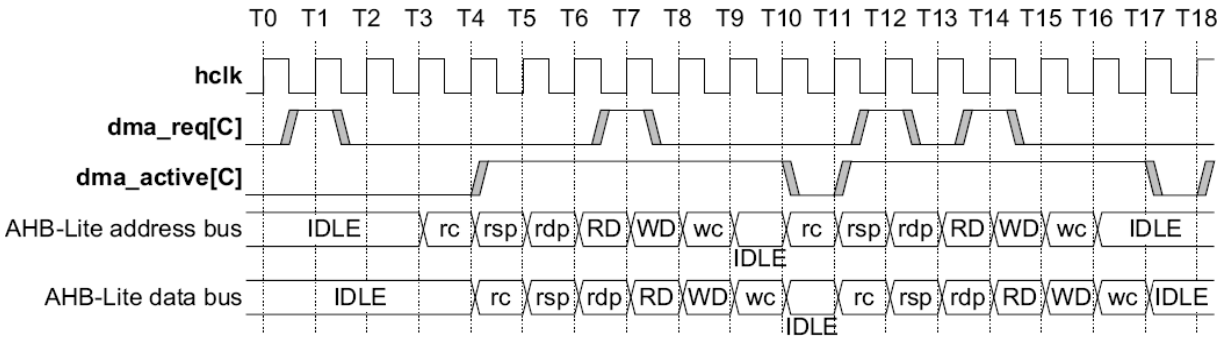


Figure 94 – Timing diagram for a pulse-triggered request from a peripheral block

Table 443 – Notes to the timing diagram for a pulse-triggered request

T1	The controller detects a service request on channel C (see rule 1) provided that chnl_req_mask_set[C] is 0 (see rule 18).
T4	The controller asserts dma_active[C] (see rules 2 and 3) and starts DMA transfers on channel C.
T4-T7	The controller reads the channel control data, where: rc – read of the channel configuration, channel_cfg; rsp – read of the source data end pointer, src_data_end_ptr; rdp – read of the destination data end pointer, dst_data_end_ptr.
T7	With dma_active[C] at 1 and chnl_req_mask_set[C] at 0, the controller detects a pulse-triggered service request on channel C (see rule 7). The controller services this request during the next arbitration.
T7-T9	The controller performs a DMA transfer on channel C, where: RD – data read; WD – data write.
T9-T10	The controller writes the channel configuration, channel_cfg, where wc – write of the channel configuration, channel_cfg.
T10	The controller resets dma_active[C], indicating completion of the DMA transfer (see rule 4).
T10-T11	The controller holds dma_active[C] at 0 for at least one HCLK cycle (see rule 5).
T11	If channel C has higher priority, the controller asserts dma_active[C], because a service request was received earlier at cycle T7 (see rules 2 and 3).
T12	With dma_active[C] at 1 and chnl_req_mask_set[C] at 0, the controller detects a pulse-triggered service request on channel C (see rule 7). The controller services this request during the next arbitration.
T14	The controller ignores the request on channel C because of the pending request received at cycle T12.
T17	The controller resets dma_active[C], indicating completion of the DMA transfer (see rule 4).
T17-T18	The controller holds dma_active[C] for at least one HCLK cycle (see rule 5).
T18	If channel C has higher priority, the controller asserts dma_active[C], because a service request was received earlier at cycle T12 (see rules 2 and 3).

25.4.2.2. **Level-triggered service request.**

Figure 95 shows the DMA controller timing diagram for a level-triggered request from a peripheral block.

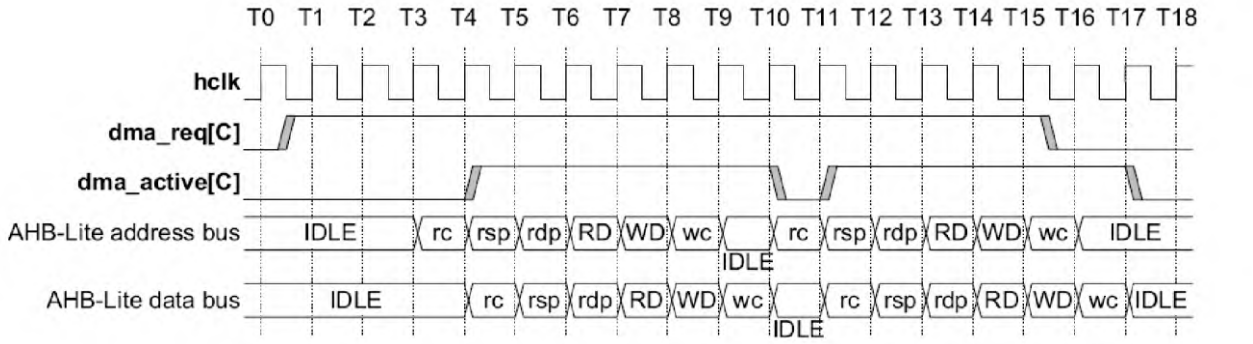


Figure 95 – Timing diagram for a level-triggered request from a peripheral block

Table 444 – Notes to the timing diagram for a level-triggered request

T1	The controller detects a service request on channel C (Table 454, rule 1) provided that chnl_req_mask_set[C] is 0 (see rule 18).
T4	The controller asserts dma_active[C] (see rules 2 and 3) and starts DMA transfers on channel C.
T4-T7	The controller reads the channel control data, where: rc – read of the channel configuration, channel_cfg; rsp – read of the source data end pointer, src_data_end_ptr; rdp – read of the destination data end pointer, dst_data_end_ptr.
T7-T9	The controller performs a DMA transfer on channel C, where: RD – data read; WD – data write.
T9-T10	The controller writes the channel configuration, channel_cfg, where wc – write of the channel configuration, channel_cfg.
T10	The controller resets dma_active[C], indicating completion of the DMA transfer (see rule 4). The controller detects a service request on channel C (see rule 1) provided that chnl_req_mask_set[C] is 0 (see rule 18).
T10-T11	The controller holds dma_active[C] at 0 for at least one HCLK cycle (see rule 5).
T11	If channel C has higher priority, the controller asserts dma_active[C] and starts a second DMA transfer on channel C.
T11-T14	The controller reads the channel control data for channel C.
T14-T16	The controller performs a DMA transfer on channel C
T15-T16	The peripheral block detects that a DMA transfer has started and clears dma_req[C].
T16-T17	The controller writes the channel configuration, channel_cfg.
T17	The controller resets dma_active[C], indicating completion of the DMA transfer (see rule 4).

When a level-triggered request is used, the peripheral block may not be fast enough to clear the request signal in time, in this case it must assert dma_stall. Asserting dma_stall prevents a repeated transfer from being performed.

25.4.2.3. **Completion flags**

Figure 96 shows a timing diagram of the dma_done[] signal (flag) operation for the following conditions:

- dma_stall and dma_waitonreq[] are 0;
- dma_stall is 1;
- dma_waitonreq[] is 1.

Assertion of the dma_done signal is propagated into the DMA_DONE_STICK register of the clock controller block (see 16.6.9 “DMA_DONE_STICK”).

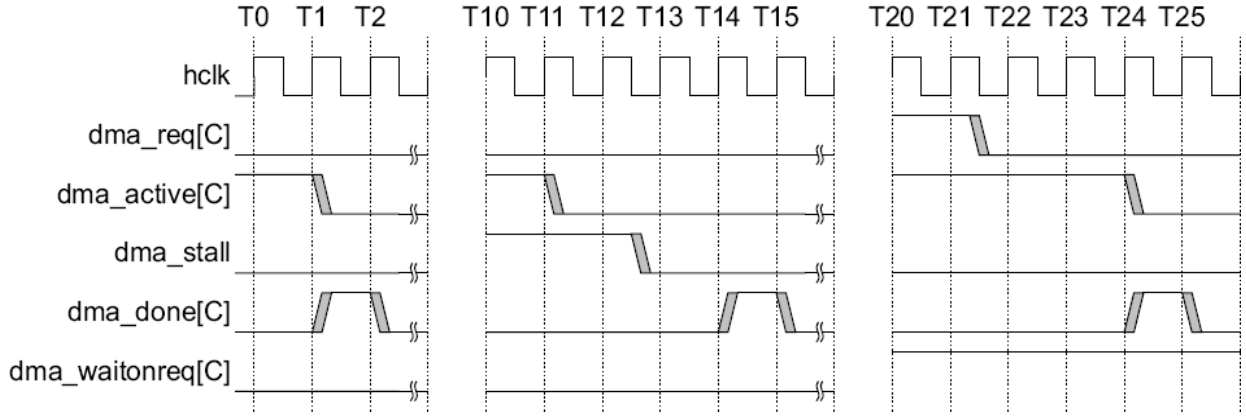


Figure 96 – dma_done operation diagrams

Table 445 – Explanations for dma_done operation, cycles T0 through T2

T1	The controller resets dma_active[C], indicating completion of the DMA transfer (see rule 4).
T1-T2	The controller completes the DMA cycle and, if cycle_ctrl[2] is 0, sets dma_done[C] to 1 for one HCLK cycle (see rules 8 and 9). For all other enabled channels the dma_done[C] signal remains 0 (see rule 6).

Table 446 – Explanations for dma_done operation, cycles T10 through T15

T11*	The controller resets dma_active[C], indicating completion of the DMA transfer (see rule 4).
T12-T13	The peripheral block resets the dma_stall signal.
T14-T15	The controller completes the DMA cycle and, if cycle_ctrl[2] is 0, sets dma_done[C] to 1 for one HCLK cycle (see rules 8 and 9). For all other enabled channels the dma_done[C] signal remains 0 (see rule 6).
* The controller does not assert dma_done[C], because dma_stall is 1 in the preceding HCLK cycle (see rules 9 and 12).	

Table 447 – Explanations for dma_done operation, cycles T20 through T25

T20	The controller has completed a DMA transfer, but because dma_waitonreq[C] is 1 it must wait until dma_req[C] is set to 0 before deasserting dma_active[C] (see rule 11) and asserting dma_done[C] (see rule 9).
T21-T25	The peripheral block deasserts dma_req[C].
T24	The controller resets dma_active[C], indicating completion of the DMA transfer (see rule 4).
T24-T25	The controller completes the DMA cycle and, if cycle_ctrl[2] is 0, sets dma_done[C] to 1 for one HCLK cycle (see rules 8 and 9). For all other enabled channels the dma_done[C] signal remains 0 (see rule 6).

25.4.2.4. **Wait-for-service request flags**

Figures 97 and 98 show examples of using wait-for-service request flags when performing 2^R transfers and single transfers:

- DMA controller timing diagram when the peripheral uses dma_waitonreq;
- DMA controller timing diagram when the peripheral uses dma_waitonreq together with dma_sreq.

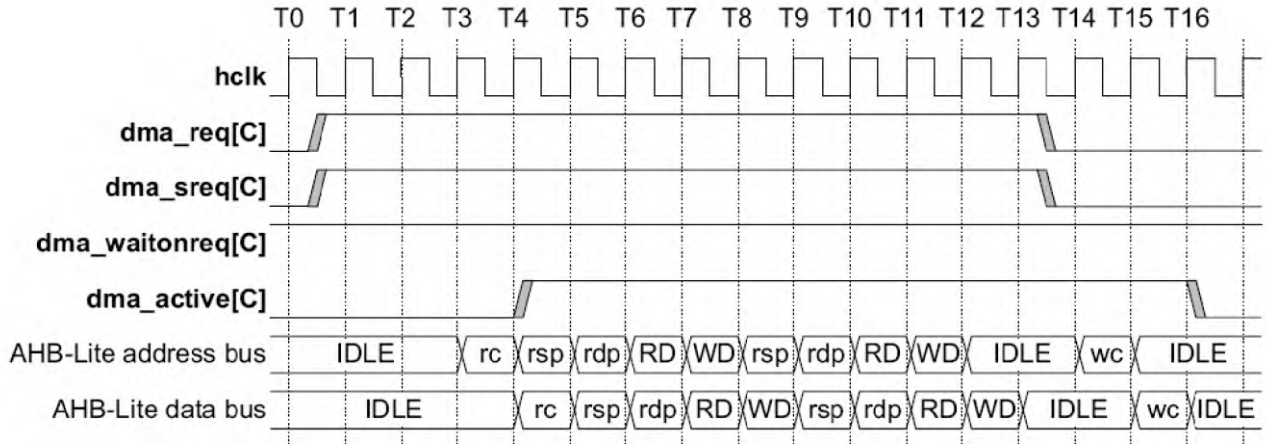


Figure 97 – DMA controller timing diagram when dma_waitonreq is used

Table 448 – Explanations for DMA controller operation when dma_waitonreq is used

T0-T16	The peripheral block must keep dma_waitonreq[C] status continuously (see rule 10)
T0-T1	The controller detects a service request on channel C (see rule 1) provided that chnl_req_mask_set[C] is 0 (see rule 18).
T3-T4	The peripheral block holds dma_req[C] and dma_sreq[C] at 1. The controller ignores the dma_sreq[C] request and services the dma_req[C] request (see rules 16 and 17).
T4	The controller asserts dma_active[C] (see rules 2 and 3) and starts DMA transfers on channel C.
T4-T7	The controller reads the channel control data, where: rc – read of the channel configuration, channel_cfg; rsp – read of the source data end pointer, src_data_end_ptr; rdp – read of the destination data end pointer, dst_data_end_ptr.

T7-T9	The controller performs a DMA transfer on channel C, where: RD – data read; WD – data write.
T9-T11	The controller reads the 2 data end pointers rsp and rdp.
T11-T13	The peripheral block resets dma_req[C] and dma_sreq[C].
T15-T16	The controller writes the channel configuration, channel_cfg, where wc – write of the channel configuration, channel_cfg.
T16	The controller resets dma_active[C], indicating completion of the DMA transfer (see rule 11). The controller sets the read value of chnl_useburst_set[C] to 0 if the number of remaining transfers is less than 2^R (see rule 15).

Figure 98 shows the operation of the DMA controller when dma_waitonreq is 1 and a single DMA transfer is performed.

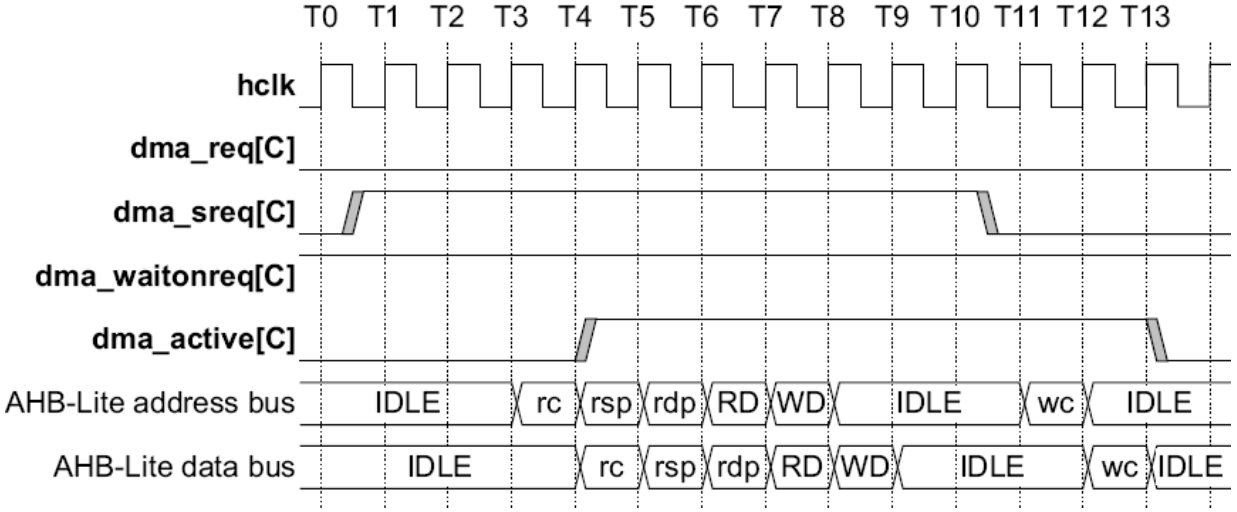


Figure 98 – DMA operation when dma_waitonreq is used together with dma_sreq

Table 449 – Explanations for DMA operation when dma_waitonreq is used together with dma_sreq

T0-T13	The peripheral block must keep dma_waitonreq[C] status continuously (see rule 10)
T0-T1	The controller detects a service request on channel C (see rule 1) provided that chnl_useburst_set[C] is 0 (see rules 13 and 14).
T3-T4	The controller services the dma_sreq[C] request (see rule 16).
T4	The controller asserts dma_active[C] (see rules 2 and 3) and starts DMA transfers on channel C.
T4-T7	The controller reads the channel control data, where: rc – read of the channel configuration, channel_cfg; rsp – read of the source data end pointer, src_data_end_ptr; rdp – read of the destination data end pointer, dst_data_end_ptr.
T7-T9	The controller performs a DMA transfer on channel C, where: RD – data read; WD – data write. This request is handled in response to dma_sreq[], therefore R=0 and, consequently, the controller performs 1 DMA transfer.
T10-T11	The peripheral block resets dma_sreq[C].
T12_T13	The controller writes the channel configuration, channel_cfg, where wc – write of the channel configuration, channel_cfg.
T13	The controller resets dma_active[C], indicating completion of the DMA transfer (see rule 11)

25.4.3. **DMA arbitration rules**

The controller supports configuration of the arbitration interval for DMA transfers. This capability makes it possible to reduce response time when servicing high-priority channels.

The controller has four bits that determine the number of AHB bus transactions before arbitration is repeated. These bits define the exponent R of 2; changing R directly sets the arbitration period to 2 to the power of R. For example, if R is 4, arbitration is performed every 16 DMA transfers.

Table 450 shows the possible arbitration periods.

Table 450 – Arbitration period in units of AHB transfers

R value	Arbitration period, every x DMA transfers
4’b0000	1
4’b0001	2
4’b0010	4
4’b0011	8
4’b0100	16
4’b0101	32
4’b0110	64
4’b0111	128
4’b1000	256
4’b1001	512
4’b1010 – 4’b1111	1024

Note: Large R values for low-priority channels must be used with care, as this can prevent servicing of requests on high-priority channels.

For $N > 2^R$ (N is the transfer number) and when the result of dividing 2^R by N is not an integer, the controller always performs a sequence of 2^R transfers until the condition $N<2^R$ becomes true. The controller then performs the remaining N transfers at the end of the DMA cycle.

The bits representing the exponent R of 2 are located in the channel control data structure. The location of these bits is described in subsection 25.5 “Channel control data structure”.

25.4.4. **Priority**

Arbitration selects which channel will be serviced in the next DMA cycle. The selection of the next channel is determined by:

- the channel number;
- the priority level assigned to the channel.

Each channel can be assigned either the default (low) priority level or a high priority level. Assignment of the priority level is performed by setting or clearing the chnl_priority_set bit.

Channel 0 has the highest priority; the priority level decreases as the channel number increases. Table 464 shows the DMA channel priority levels in descending order.

Table 451 – DMA channel priority level

Channel number	Priority level setting	Priority level, in descending order
0	High	Highest priority level
1	High	-
2	High	-
-	High	-
-	High	-
-	High	-
30	High	-
31	High	-
0	Default (low)	-
1	Default (low)	-
2	Default (low)	-
-	Default (low)	-
-	Default (low)	-
-	Default (low)	-
30	Default (low)	-
31	Default (low)	Lowest priority level

After completion of a DMA cycle, the controller selects the next channel to be serviced from all enabled DMA channels. Figure 99 shows the algorithm for selecting the next channel to be serviced.

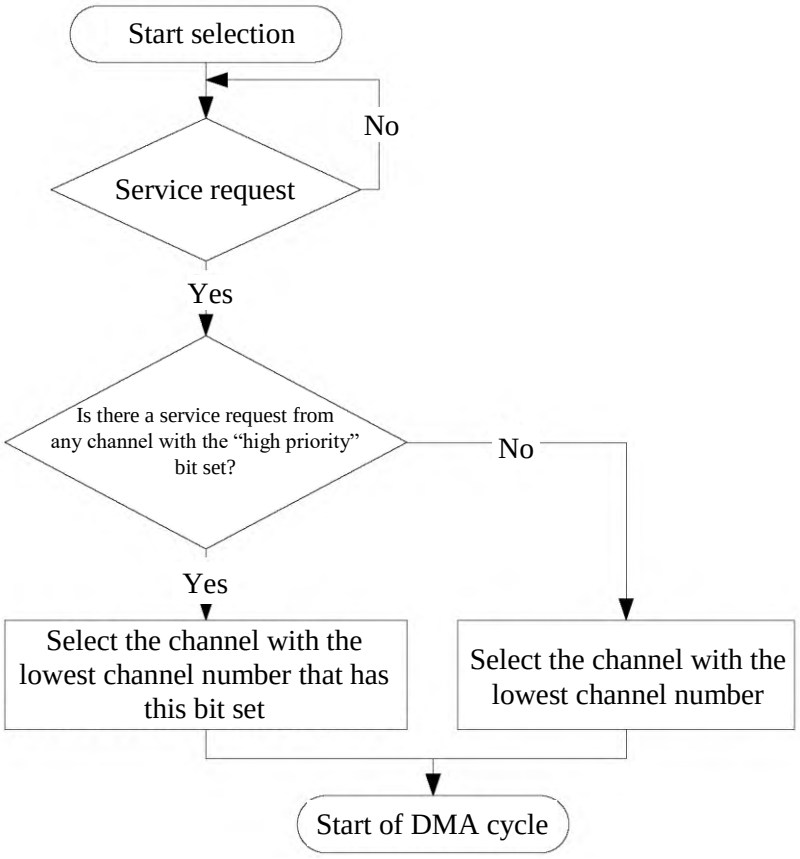


Figure 99 – Algorithm for selecting the next channel to be serviced

25.4.5. **DMA cycle types**

The cycle_ctrl bits define how the controller executes DMA cycles. The meaning of these bits is described below.

Table 452 – DMA cycle types

cycle_ctrl	Description
3’b000	Channel control data structure in the disabled state
3’b001	Basic DMA cycle
3’b010	Auto-request
3’b011	Ping-pong mode
3’b100	Memory “Execution with configuration change” mode using the primary channel control data
3’b101	Memory “Execution with configuration change” mode using the alternate channel control data
3’b110	Peripheral “Execution with configuration change” mode using the primary channel control data
3’b111	Peripheral “Execution with configuration change” using the alternate channel control data
Note: The cycle_ctrl bits are located in the memory area allocated for channel_cfg; see subsection “Channel control data structure”.	

For all DMA cycle types, re-arbitration takes place after 2^R DMA transfers. If a long arbitration period is configured on a low-priority channel, this can block all service requests from other channels until 2^R DMA transfers have been completed on that channel. Therefore, when setting the value of R, it is necessary to consider that this may increase the response time for service requests on high-priority channels.

This section describes the following DMA cycle types:

– invalid;
– basic;
– auto-request;
– ping-pong;
– memory “Execution with configuration change” mode;
– peripheral “Execution with configuration change” mode.

25.4.5.1. **Invalid**

After a DMA cycle completes, the controller sets the cycle type to “invalid” to prevent repetition of the completed DMA cycle.

25.4.5.2. **Basic**

In this mode, the controller operates using only the primary or the alternate channel control data. After the channel is enabled and the controller receives a service request, the DMA cycle proceeds as follows:

1. The controller performs 2^R transfers. If the number of remaining transfers is 0, the controller proceeds to step 3.
2. Arbitration is performed:
   – if a high-priority channel issues a service request, the controller starts servicing that channel;
   – if the peripheral block or software issues a service request (a repeated service request on this channel), the controller returns to step 1.
3. The controller sets dma_done[C] to 1 for one HCLK cycle. This indicates to the central processor that the DMA cycle has completed.

25.4.5.3. **Auto-request**

In this mode, the controller waits for a single service request to enable operation and execute a DMA cycle. This mode makes it possible to transfer large data packages without significantly increasing the response time to high-priority requests and without requiring multiple service requests from the processor or peripheral blocks.

In this mode, the controller can use either the primary or the alternate channel control data structure. After the channel is enabled and the controller receives a service request, the DMA cycle proceeds as follows:

1 The controller performs 2^R transfers on channel C. If the number of remaining transfers is 0, the controller proceeds to step 3.

2 The controller performs arbitration. When channel C becomes the highest-priority channel, the controller returns to step 1.

3 The controller sets dma_done[C] to 1 for one HCLK cycle. This indicates to the central processor that the DMA cycle has completed.

25.4.5.4. **Ping-pong**

In this mode, the controller executes a DMA cycle using one channel control data structure and then executes another DMA cycle using the other control data structure. The controller executes DMA cycles, alternating between the control data structures, until it reads an "invalid" control data structure or the processor disables the channel.

Figure 100 shows an example of controller operation in ping-pong mode.

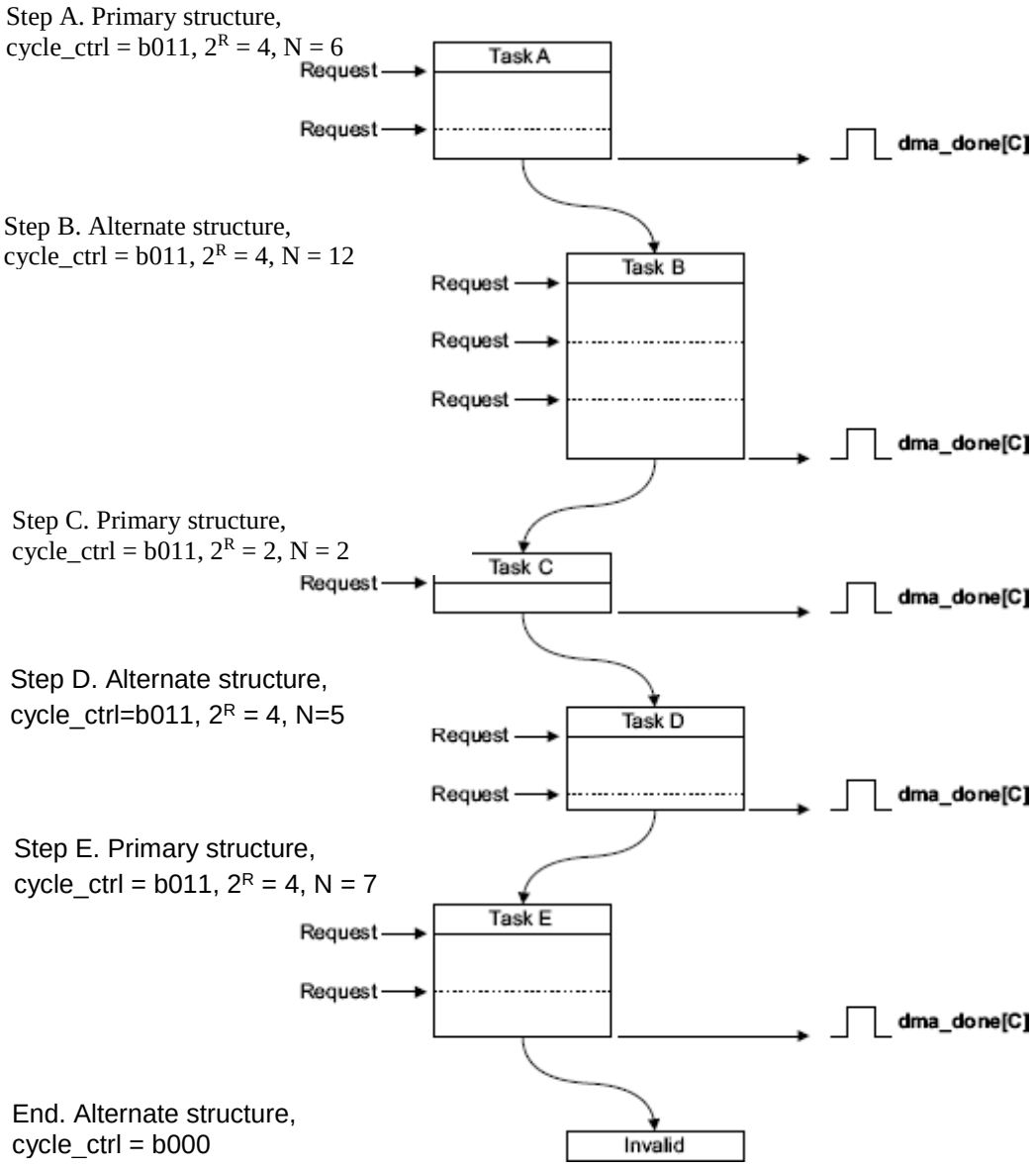


Figure 100 – Example of controller operation in ping-pong mode

Explanations for Figure 100

Step A
The processor programs the primary channel control data for step A.
The processor programs the alternate channel control data for step B. This allows the controller to switch to step B immediately after completing step A, provided that the controller does not receive a service request from a higher-priority channel.
The controller receives a request and performs 4 DMA transfers.
The controller performs arbitration. After receiving a service request on the same channel, and in the absence of high-priority requests, the controller continues the cycle.
The controller performs the remaining 2 DMA transfers.
The controller sets dma_done[C] to 1 for one HCLK cycle and enters arbitration.

After completing step A, the processor can program the primary channel control data for step C. This allows the controller to switch to step C immediately after completing step B, provided that the controller does not receive a service request from a higher-priority channel.

After receiving a new service request on the channel and assuming this request has the highest priority, step B is executed:

Step B
The controller performs 4 DMA transfers.
The controller performs arbitration. After receiving a service request on the same channel, and in the absence of high-priority requests, the controller continues the cycle.
The controller performs 4 DMA transfers.
The controller performs arbitration. After receiving a service request on the same channel, and in the absence of high-priority requests, the controller continues the cycle.
The controller performs the remaining 4 DMA transfers.
The controller sets dma_done[C] to 1 for one HCLK cycle and enters arbitration.

After completing step B, the processor can program the alternate channel control data for step D.

After receiving a new service request on the channel and assuming this request has the highest priority, step C is executed:

Step C
The controller performs 2 DMA transfers.
The controller sets dma_done[C] to 1 for one HCLK cycle and enters arbitration.

After completing step C, the processor can program the primary channel control data for step E.

After receiving a new service request on the channel and assuming this request has the highest priority, step D is executed:

Step D The controller performs 4 DMA transfers.
The controller performs arbitration. After receiving a service request on the same channel, and in the absence of high-priority requests, the controller continues the cycle.
The controller performs the remaining DMA transfer.
The controller sets dma_done[C] to 1 for one HCLK cycle and enters arbitration.

After receiving a new service request on the channel and assuming this request has the highest priority, step E is executed:

Step E The controller performs 4 DMA transfers.
The controller performs arbitration. After receiving a service request on the same channel, and in the absence of high-priority requests, the controller continues the cycle.
The controller performs the remaining 3 DMA transfers.
The controller sets dma_done[C] to 1 for one HCLK cycle and enters arbitration.

If the controller receives a new service request on this channel and that request has the highest priority, it will attempt to execute the next step. However, because the processor has not programmed alternate control data and, after completion of step D, the controller set cycle_ctrl to b000, DMA transfers are stopped.

Note: To terminate a DMA cycle executing in ping-pong mode, the controller can also be switched at step E to “Basic DMA cycle” mode by setting cycle_ctrl to 3’b001.

25.4.5.5. **Memory “Execution with configuration change” mode**

In this mode, after an initial service request is received, the controller performs 4 DMA transfers using the primary control data. When these transfers are completed, the controller starts a DMA cycle using the alternate control data. Then the controller performs another 4 DMA transfers using the primary control data. The controller continues to execute DMA cycles, switching between the control data structures, until one of the following conditions is met:

- the processor switches the controller to “Basic” mode during a cycle that uses the alternate structure;
- the controller reads an “invalid” control data structure.

Note: After the controller executes N transfers using the primary control data, it makes this control data “invalid” by setting cycle_ctrl to 3’b000.

In this operating mode, the controller sets the dma_done[C] flag only when the DMA transfer is completed using the basic cycle.

In this mode, the controller uses the primary control data to program the alternate control data. Table 453 shows the channel_cfg memory fields that must be defined as constants, as well as the fields whose values are defined by the user.

Table 453 – Channel_cfg for the primary control data structure in memory “Execution with configuration change” mode

Bits	Symbol	Value	Description
Fields with constant values			
31, 30	dst_inc	2’b10	The controller increments the address in word steps.
29, 28	dst_size	2’b10	The controller transfers data in word units.
27, 26	src_inc	2’b10	The controller increments the address in word steps.
25, 24	src_size	2’b10	The controller transfers data in word units.
17…14	R_power	4’b0010	The controller performs 4 DMA transfers.
3	next_useburst	1’b0	For this mode, this bit must be 0.
2…0	cycle_ctrl	3’b100	The controller operates in peripheral “Execution with configuration change” mode.
Fields with user-defined values			
23…21	dst_prot_ctrl	-	Defines the HPROT state when writing data to the destination.
20…18	src_prot_ctrl	-	Defines the HPROT state when reading data from the source.
13…4	n_minus_1	N*	Configures the controller to perform N DMA transfers, where N is a multiple of 4.
* Since the R_power bits are set to 2, the value of N must be a multiple of 4. The value equal to N/4 is the number of times the alternate control data must be configured.			

Figure 101 shows an example of operation in memory “Execution with configuration change” mode.

Initialization:

1. Configuration of the primary control data to allow copying of A, B, C, and D: cycle_ctrl = 3’b100, $2^R = 4$, N=16.

2. Writing of the primary data to memory using the structure shown in the table below.

	src_data_end_ptr	dst_data_end_ptr	channel_cfg	Unused
Data for Task A	0x0A000000	0x0AE00000	cycle_ctrl = b101, 2^R = 4, N = 3	0xXXXXXXXX
Data for Task B	0x0B000000	0x0BE00000	cycle_ctrl = b101, 2^R = 2, N = 8	0xXXXXXXXX
Data for Task C	0x0C000000	0x0CE00000	cycle_ctrl = b101, 2^R = 8, N = 5	0xXXXXXXXX
Data for Task D	0x0D000000	0x0DE00000	cycle_ctrl = b001, 2^R = 4, N = 4	0xXXXXXXXX

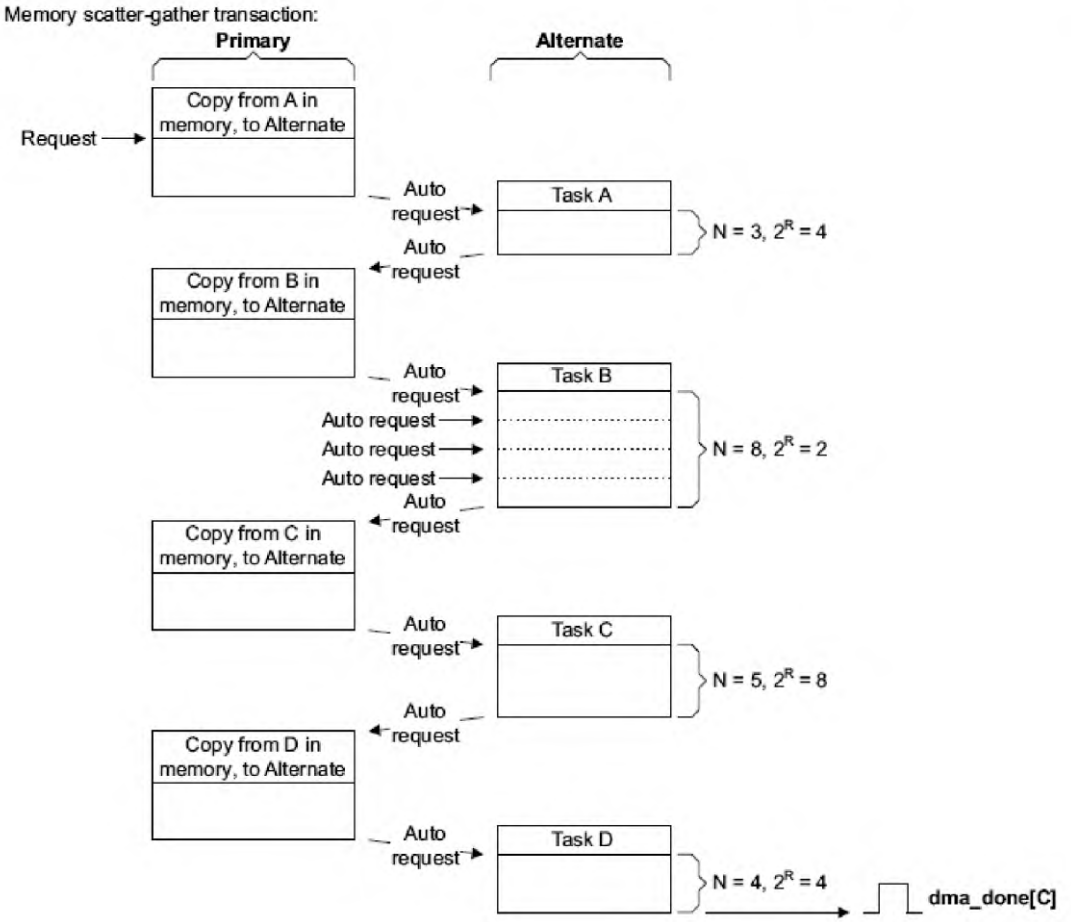


Figure 101 – Example of DMA operation in “Execution with configuration change” mode

Initialization:

1. The processor configures the primary control data structure for operation in memory mode “Execution with configuration change” by setting cycle_ctrl to 3’b100. Since the channel control data consists of 4 words, 2^R must be set to 4. In this example, the number of transfers is 4 and therefore N is set to 16.
2. The processor writes the control data for steps A, B, C, and D to the memory area at the address specified in src_data_end_ptr.
3. The processor enables the DMA channel.

Transfers in this mode start executing when the controller receives a service request via dma_req[] or a request from the processor. The execution order is as follows:

Primary, copy A

Upon receiving a service request, the controller performs 4 DMA transfers. These transfers write the alternate control data structure for step A.

The controller generates an auto-request for the channel and then performs arbitration.

Step A

The controller executes step A. When this step is completed, the controller generates an auto-request for the channel and performs arbitration.

Primary, Copy B

The controller performs 4 DMA transfers. These transfers write the alternate control data structure for step B.

The controller generates an auto-request for the channel and then performs arbitration.

Step B

The controller executes step B. When this step is completed, the controller generates an auto-request for the channel and performs arbitration.

Primary, copy C

The controller performs 4 DMA transfers. These transfers write the alternate control data structure for step C.

The controller generates an auto-request for the channel and then performs arbitration.

Step C

The controller executes step C. When this step is completed, the controller generates an auto-request for the channel and performs arbitration.

Primary, copy D

The controller performs 4 DMA transfers. These transfers write the alternate control data structure for step D.

The controller sets cycle_ctrl in the primary control data to 3’b000 to indicate that this control data structure is “invalid”.

The controller generates an auto-request for the channel and then performs arbitration.

Step D

The controller executes step D using the basic DMA cycle.

The controller sets the dma_done[C] flag to 1 for one HCLK cycle and enters arbitration.

25.4.5.6. **Peripheral “Execution with configuration change” mode**

In this mode, after an initial service request is received, the controller performs 4 DMA transfers using the primary control data. When these transfers are completed, the controller starts a DMA cycle using the alternate control data without entering arbitration or setting dma_active[C] to 0.

Note: This is the only case where the controller skips arbitration after completing a DMA transfer based on the primary control data.

After this cycle is finished, the controller performs arbitration and, once it receives a service request from the highest-priority peripheral, it carries out another 4 DMA transfers using the primary control data. When these transfers are completed, the controller starts a DMA cycle using the alternate control data without entering arbitration or setting dma_active[C] to 0.

The controller continues to execute DMA cycles, switching between the control data structures, until one of the following conditions is met:

– the processor switches the controller to “Basic” mode during a cycle that uses the alternate structure;

– the controller reads an “invalid” control data structure.

Note: After the controller executes N transfers using the primary control data, it makes this control data “invalid” by setting cycle_ctrl to 3’b000.

In this operating mode, the controller sets the dma_done[C] flag only when the DMA transfer is completed using the basic cycle.

In this mode, the controller uses the primary control data to program the alternate control data. Table 454 shows the channel_cfg memory fields that must be defined as constants, as well as the fields with values defined by the user.

Table 454 – Channel_cfg for the primary control data structure in peripheral “Execution with configuration change” mode

Bits	Symbol	Value	Description
Fields with constant values			
31, 30	dst_inc	2’b10	The controller increments the address in word steps.
29, 28	dst_size	2’b10	The controller transfers data in word units.
27, 26	src_inc	2’b10	The controller increments the address in word steps.
25, 24	src_size	2’b10	The controller transfers data in word units.
17…14	R_power	4’b0010	The controller performs 4 DMA transfers.
2…0	cycle_ctrl	3’b110	The controller operates in peripheral “Execution with configuration change” mode.
Fields with user-defined values			
23…21	dst_prot_ctrl	-	Defines the HPROT state when writing data to the destination.
20…18	src_prot_ctrl	-	Defines the HPROT state when reading data from the source.
13…4	n_minus_1	N*	Configures the controller to perform N DMA transfers, where N is a multiple of 4.
3	next_useburst	-	When set to 1, the controller will set chnl_useburst_set[C] to 1 after performing a transfer with an alternate structure.
* Since the R_power bits are set to 2, the value of N must be a multiple of 4. The value equal to N/4 is the number of times the alternate control data must be configured.			

Figure 102 shows an example of operation in peripheral “Execution with configuration change” mode.

Initialization:

1. Configuration of the primary control data to allow copying of A, B, C, and D: cycle_ctrl = 3’b110, $2^R = 4$, N=16.
2. Writing of the primary data to memory using the structure shown in the table below.

	src_data_end_ptr	dst_data_end_ptr	channel_cfg	Unused
Data for Task A	0x0A000000	0x0AE00000	cycle_ctrl = b111, $2^R = 4$, N = 3	0xXXXXXXXX
Data for Task B	0x0B000000	0x0BE00000	cycle_ctrl = b111, $2^R = 2$, N = 8	0xXXXXXXXX
Data for Task C	0x0C000000	0x0CE00000	cycle_ctrl = b111, $2^R = 8$, N = 5	0xXXXXXXXX
Data for Task D	0x0D000000	0x0DE00000	cycle_ctrl = b001, $2^R = 4$, N = 4	0xXXXXXXXX

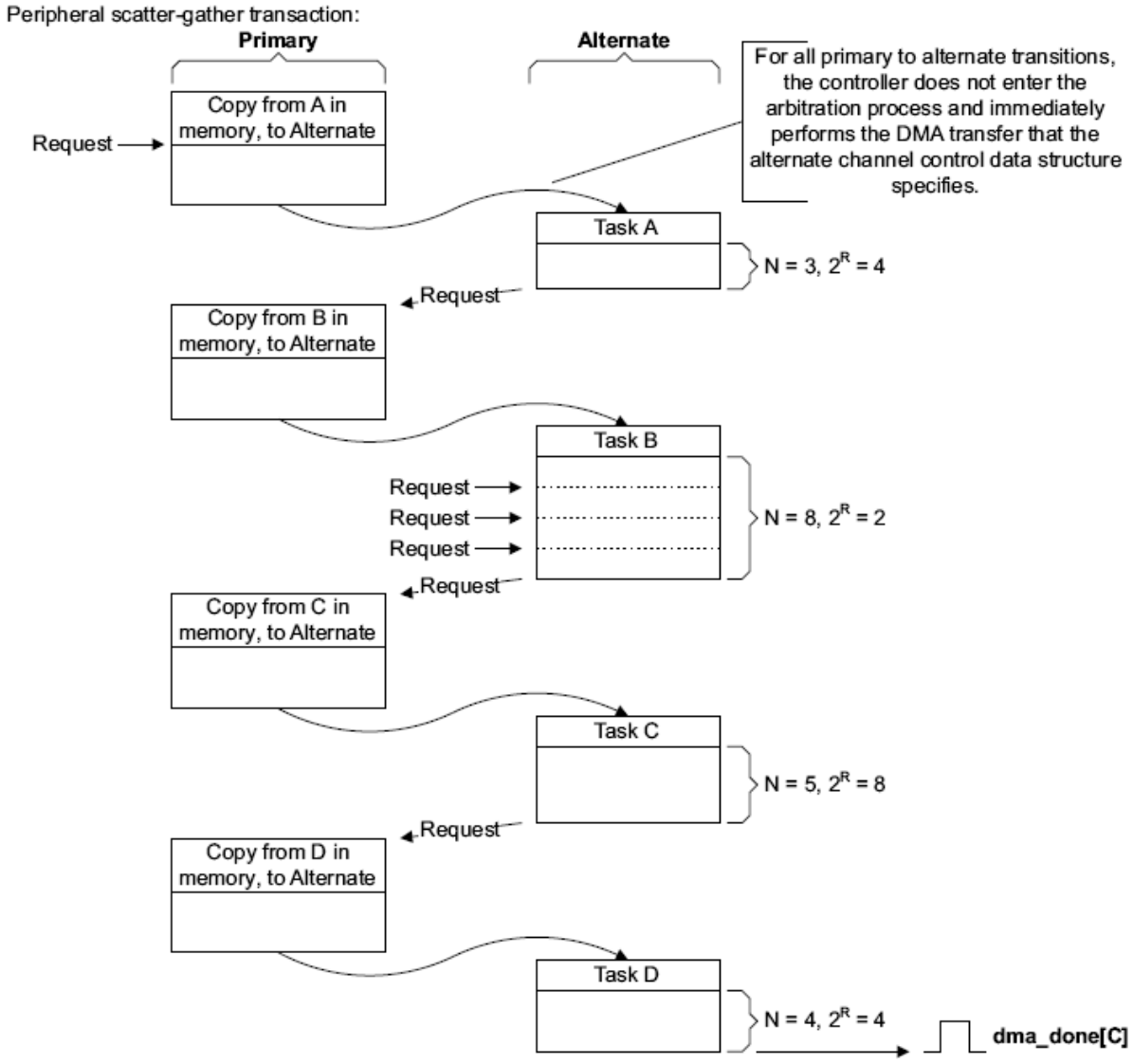


Figure 102 – Example of DMA operation in “Execution with configuration change” mode

Initialization:

1. The processor configures the primary control data structure for operation in peripheral mode “Execution with configuration change” by setting cycle_ctrl to 3’b110. Since the channel control data consists of 4 words, 2^R must be set to 4. In this example, the number of tasks is 4 and therefore N is set to 16.
2. The processor writes the control data for steps A, B, C, and D to the memory area at the address specified in src_data_end_ptr.
3. The processor enables the DMA channel.

Transfers in this mode start executing when the controller receives a service request via dma_req[] The tasks are performed as follows:

Primary, copying from memory area A

Upon receiving a service request, the controller performs 4 DMA transfers. These transfers write the alternate control data structure for step A.

Step A

The controller performs Step A.

When completed, the controller performs arbitration. Primary, copying from memory area B

The controller performs 4 DMA transfers. These transfers write the alternate control data structure for step B.

Step B

The controller executes step B. To complete the task, the peripheral must set 3 requests in sequence.

When completed, the controller performs arbitration. Primary, copying from memory area C

The controller performs 4 DMA transfers. These transfers write the alternate control data structure for step C.

Step C

The controller performs Step C.

When completed, the controller performs arbitration.

Once a new service request is submitted by the peripheral, provided that this request is the highest priority, the process continues as follows:

Primary, copying from memory area D

The controller performs 4 DMA transfers. These transfers write the alternate control data structure for step D.

The controller sets cycle_ctrl in the primary control data to 3’b000 to indicate that this control data structure is “invalid”.

Step D

The controller executes step D using the basic DMA cycle.

The controller sets the dma_done[C] flag to 1 for one HCLK cycle and enters arbitration.

25.4.5.7. **Error indication**

When the controller receives an error response on the AHB bus, it performs the following actions:

- disables the channel associated with the error;
- sets the dma_err flag to 1.

After detecting the dma_err flag, the processor determines the number of the channel that was active at the moment the error occurred. To do this, it performs the following:

1 It reads the chnl_enable_set register in order to create a list of disabled channels;

2 If a channel has set the dma_done[] flag, the controller disables that channel. The program executed by the processor must always maintain information about the channels that have recently set dma_done[] flags. Assertion of the dma_done signal is propagated into the DMA_DONE_STICK register of the clock controller block (see 16.6.9 “DMA_DONE_STICK”).

3 The processor must compare the list of disabled channels obtained in step 1 with the information about the channels that have recently set dma_done[] flags. The channel for which there is no record of a dma_done[] flag being set is the channel associated with the error.

25.5. **Channel control data structure**

A region must be allocated in system memory for storing the channel control data. System memory must:

- provide a contiguous system memory region that is accessible to both the controller and the processor;
- have a base address that is an integer multiple of the total size of the channel control-data structure.

Figure 103 shows the memory region required by the controller for the channel control-data structures when 32 channels and an optional alternate control-data structure are used.

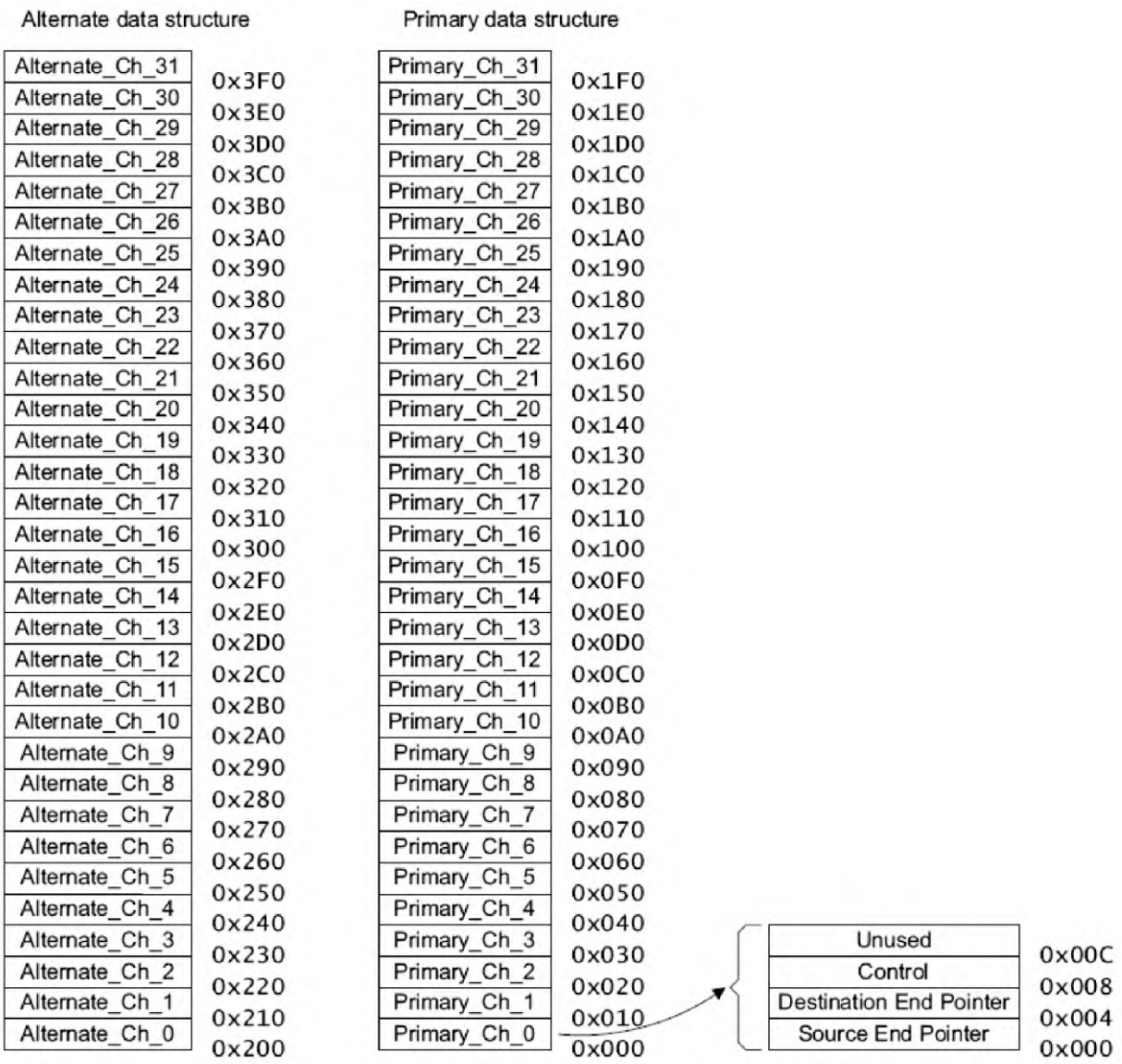


Figure 103 – Memory map for 32 channels including alternate control-data structure

The example uses 1 Kbyte of system memory. In this example, the controller uses the least significant 10 address bits to access all elements of the control-data structure, and therefore the base address of the structure must be 0xXXXXX000, then 0xXXXXX400, then 0xXXXXX800, then 0xXXXXXC00.

The base address of the primary control-data structure is set by writing the corresponding value to the ctrl_base_ptr register.

The required size of the system-memory region depends on:

– the number of channels implemented in the controller;
– whether an alternate control-data structure is used.

Table 455 lists the address bits used by the controller to access the various elements of the control data structure, depending on the number of channels implemented in the controller.

Table 455 – Address bits corresponding to the elements of the control data structure

Number of channels implemented in the controller	[9]	[8]	[7]	[6]	[5]	[4]	[3:0]
1						A	0x0 0x4 0x8
2					A	C[0]	
3-4				A	C[1]	C[0]	
5-8			A	C[2]	C[1]	C[0]	
9-16		A	C[3]	C[2]	C[1]	C[0]	
17-32	A	C[4]	C[3]	C[2]	C[1]	C[0]	

where A selects one of the channel control data structures:
A = 0 selects the primary control data structure;
A = 1 selects the alternate control data structure.

C[x:0] selects the DMA channel.

Address[3:0] selects one of the control elements:
0x0 – selects the source end pointer;
0x4 – selects the destination end pointer;
0x8 – selects the control data configuration;
0xC – the controller has no access to this address.

If necessary, these addresses can be made available to the processor as system memory.

Note: There is no need to compute the base address of the alternate control data structure, because the alt_ctrl_base_ptr register contains this information.

Figure 104 illustrates an example implementation of the controller using three DMA channels with an alternate control data structure.

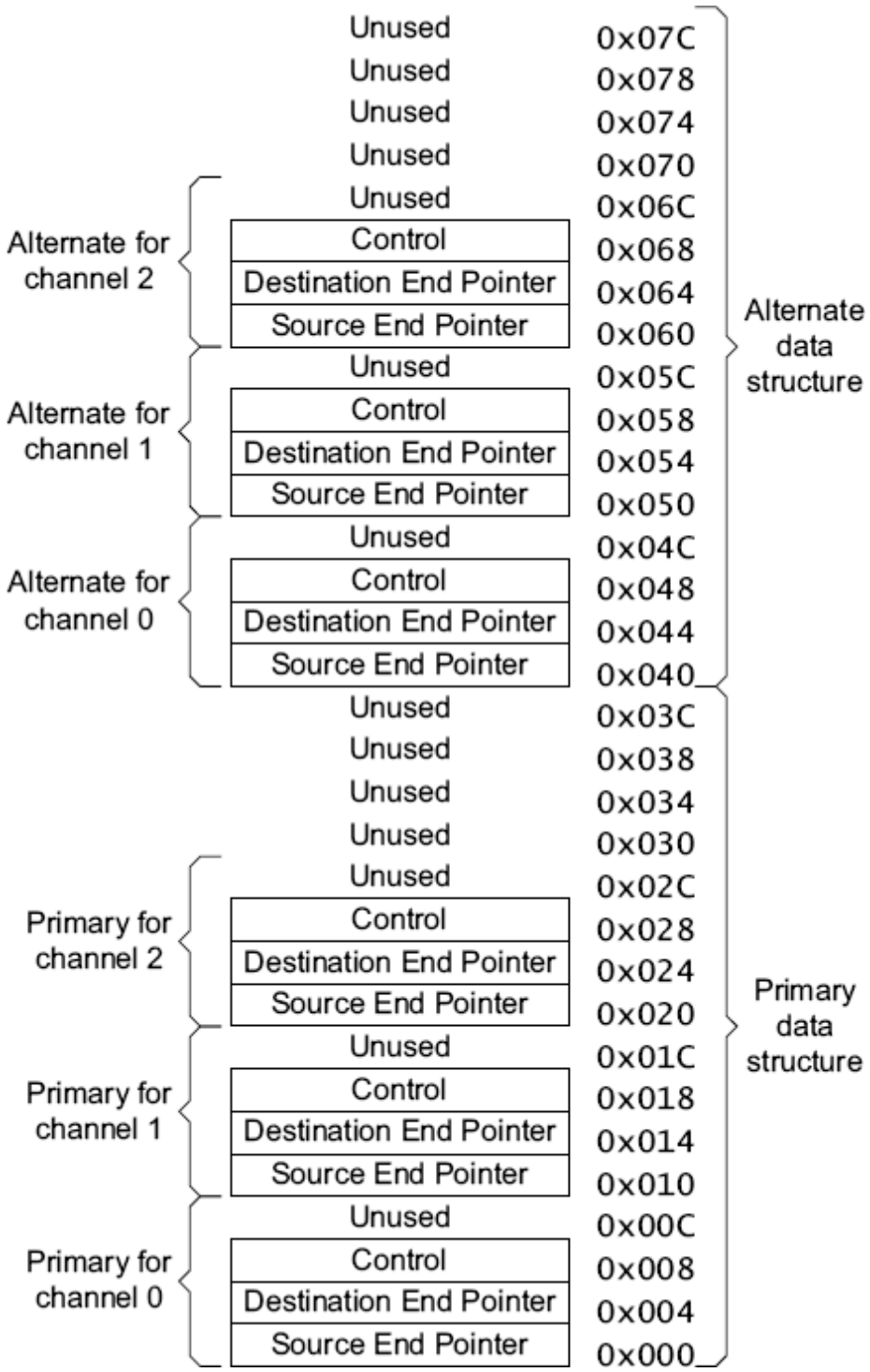


Figure 104 – Memory map for three DMA channels including an alternate control data structure (where “Destination end pointer” is the destination data end pointer; “Source end pointer” is the source data end pointer; “Control” is the control word)

The example control data layout uses 128 bytes of system memory. In this example, the controller uses the least significant 6 address bits to access all elements of the control data structure, and therefore the base address of the structure must be 0xXXXXXX00, then 0xXXXXXX80.

Table 456 shows the allowed base address values for the primary control data structure, depending on the number of DMA channels implemented in the controller.

Table 456 – Allowed base addresses

Number of DMA channels	Allowed base address values for the primary control data structure
1	0xXXXXXX00, 0xXXXXXX20, 0xXXXXXX40, 0xXXXXXX60, 0xXXXXXX80, 0xXXXXXXA0, 0xXXXXXXC0, 0xXXXXXXE0
2	0xXXXXXX00, 0xXXXXXX40, 0xXXXXXX80, 0xXXXXXXC0
3-4	0xXXXXXX00, 0xXXXXXX80
5-8	0xXXXXX000, 0xXXXXX100, 0xXXXXX200, 0xXXXXX300, 0xXXXXX400, 0xXXXXX500, 0xXXXXX600, 0xXXXXX700, 0xXXXXX800, 0xXXXXX900, 0xXXXXXA00, 0xXXXXXB00, 0xXXXXXC00, 0xXXXXXD00, 0xXXXXXE00, 0xXXXXXF00
9-16	0xXXXXX000, 0xXXXXX200, 0xXXXXX400, 0xXXXXX600, 0xXXXXX800, 0xXXXXXA00, 0xXXXXXC00, 0xXXXXXE00
17-32	0xXXXXX000, 0xXXXXX400, 0xXXXXX800, 0xXXXXXC00

The controller uses system memory to access the two end address pointers and the control bits for each channel. The following subsections describe these 32-bit memory locations and the procedure the controller uses to calculate the DMA transfer address:

- source end pointer;
- destination end pointer;
- control bits;
- address calculation.

25.5.1. **Source end pointer**

The memory region named src_data_end_ptr contains the pointer to the last address of the source data location. The bit assignments for this location are shown in Table 457.

Table 457 – Bit assignments of src_data_end_ptr

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…0	src_data_end_ptr	Pointer to the last address of the source data

Before the controller performs a DMA transfer, this memory location must be defined. The controller reads the value of this location before starting the 2^R DMA transfers.

Note: The controller has no write access to this memory region.

25.5.2. **Destination end pointer**

The memory region named dst_data_end_ptr contains the pointer to the last address of the destination data location. The bit assignments for this location are shown in Table 458.

Table 458 – Bit assignments of dst_data_end_ptr

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…0	dst_data_end_ptr	Pointer to the last address of the destination data

Before the controller performs a DMA transfer, this memory location must be defined. The controller reads the value of this location before starting the 2^R DMA transfers.

Note: The controller has no write access to this memory region.

25.5.3. **Control bits**

A memory region called channel_cfg provides control over each DMA transfer. The bit assignments for this location are shown in Table 459.

Table 459 – Bit names of the channel_cfg memory region

Number	31...30	29...28	27...26	25...24	23...21
Access	R/W	R/W	R/W	R/W	R/W
Reset	10	10	10	10	-
	dst_inc	dst_size	src_inc	src size	dst_prot_crtl

Number	20...18	17...14	13...4	3	2...0
Access	R/W	R/W	R/W	R/W	R/W
Reset	-	0010	-	0	100
	Src_prot_ctrl	R_power	n_minus_1	next_useburst	cycle_ctrl

Table 460 – Bit assignments of channel_cfg

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31, 30	dst_inc	Destination address increment step. The increment step depends on the source data bit. Source data bit depth = bytes: 2’b00 = byte; 2’b01 = half word (16-bit word); 2’b10 = word (32-bit word); 2’b11 = no increment. The address remains equal to the value of the dst_data_end_ptr memory region. Source data width = halfword: 2’b00 = reserved; 2’b01 = half-word; 2’b10 = word; 2’b11 = no increment. The address remains equal to the value of the dst_data_end_ptr memory region. Source data width = word: 2’b00 = reserved; 2’b01 = reserved; 2’b10 = word; 2’b11 = no increment. The address remains equal to the value of the dst_data_end_ptr memory region

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
29, 28	dst_size	Destination data width. Note – The value of this field must be equal to the value of the src_size field.
27, 26	src_inc	Source address increment step. The increment step depends on the source data bit. Source data bit depth = bytes: 2'b00 = byte; 2'b01 = half word (16-bit word); 2'b10 = word (32-bit word); 2'b11 = no increment. The address remains equal to the value of the src_data_end_ptr memory region Source data width = halfword: 2'b00 = reserved; 2'b01 = half word; 2'b10 = word; 2'b11 = no increment. The address remains equal to the value of the src_data_end_ptr memory region Source data width = word: 2'b00 = reserved; 2'b01 = reserved; 2'b10 = word; 2'b11 = no increment. The address remains equal to the value of the src_data_end_ptr memory region
25, 24	src_size	Defines the source data width: 2'b00 = byte; 2'b01 = half word (16-bit word); 2'b10 = word (32-bit word); 2'b11 = reserved
23…21	dst_prot_ctrl	Sets the HPROT[3:1] value when the controller writes data to the destination. Bit [23] controls HPROT[3]: 0 = HPROT[3] is 0 and the access is non-cacheable; 1 = HPROT[3] is 1 and the access is cacheable. Bit [22] controls HPROT[2]: 0 = HPROT[2] is 0 and the access is non-bufferable; 1 = HPROT[2] is 1 and the access is bufferable. Bit [21] controls HPROT[1]: 0 = HPROT[1] is 0 and the access is non-privileged; 1 = HPROT[1] is 1 and the access is privileged.

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
20…18	src_prot_ctrl	Sets the HPROT[3:1] value when the controller reads data from the source. Bit [20] controls HPROT[3]: 0 = HPROT[3] is 0 and the access is non-cacheable; 1 = HPROT[3] is 1 and the access is cacheable. Bit [19] controls HPROT[2]: 0 = HPROT[2] is 0 and the access is non-bufferable; 1 = HPROT[2] is 1 and the access is bufferable. Bit [18] controls HPROT[1]: 0 = HPROT[1] is 0 and the access is non-privileged; 1 = HPROT[1] is 1 and the access is privileged.
17…14	R_power	Sets the number of DMA transfers before the controller performs arbitration. Possible values: 4'b0000 – arbitration after each DMA transfer; 4'b0001 – arbitration after 2 DMA transfers; 4'b0010 – arbitration after 4 DMA transfers; 4'b0011 – arbitration after 8 DMA transfers; 4'b0100 – arbitration after 16 DMA transfers; 4'b0101 – arbitration after 32 DMA transfers; 4'b0110 – arbitration after 64 DMA transfers; 4'b0111 – arbitration after 128 DMA transfers; 4'b1000 – arbitration after 256 DMA transfers; 4'b1001 – arbitration after 512 DMA transfers; 4'b1010 – 4'b1111 – arbitration after 1024 DMA transfers. This means that arbitration is not performed, because the maximum number of DMA transfers is 1024.
13…4	n_minus_1	Before a DMA cycle is started, these bits specify the total number of DMA transfers that make up the DMA cycle. These bits must be set to a value corresponding to the desired DMA cycle length. A 10-bit number plus 1 specifies the number of DMA transfers. Possible values: 10'b0000000000 = 1 DMA transfer; 10'b0000000001 = 2 DMA transfers; 10'b0000000010 = 3 DMA transfers; 10'b0000000011 = 4 DMA transfers; 10'b0000000100 = 5 DMA transfers; 10'b0000000101 = 6 DMA transfers; …. 10'b1111111111 = 1024 DMA transfers. The controller updates this field before it performs arbitration. This allows the controller to keep track of the remaining number of DMA transfers until the DMA cycle is complete.
3	next_useburst	Determines whether chnl_useburst_set[C] is set to 1 when the controller operates in the peripheral "Execution with configuration change" mode and when the controller completes a DMA cycle using the alternate control data. Note: Before completing a DMA cycle that uses the alternate control data, the controller sets chnl_useburst_set[C] to 0 if the number of remaining DMA transfers is less than 2^R. Setting the next_useburst bit determines whether the controller additionally overrides the chnl_useburst_set[C] bit.

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
		If the controller executes a DMA cycle in the peripheral “Execution with configuration change” mode, then after a cycle that uses the alternate control data has completed, the following occurs depending on the state of next_useburst: 0 – the controller does not change the value of chnl_useburst_set[C]. If chnl_useburst_set[C] is 0, then for all remaining DMA cycles in the peripheral “Execution with configuration change” mode the controller responds to requests on dma_req[] and dma_sreq[], and uses the alternate control data while executing the DMA cycles. 1 – the controller changes the value of chnl_useburst_set[C] to 1. Therefore, for the remaining DMA cycles in the peripheral “Execution with configuration change” mode the controller responds only to requests on dma_req[], and uses the alternate control data while executing the DMA cycles.
2…0	cycle_ctrl	Operating mode when executing a DMA cycle: 3’b000 Stop. Indicates that the control data structure is “invalid”; 3’b001 Basic. The controller must receive a new request to complete the DMA cycle; before this, it must perform arbitration; 3’b010 Auto-request. During arbitration the controller automatically issues a service request on the corresponding channel. This means that an initial service request is sufficient to execute the DMA cycle; 3’b011 Ping-pong. The controller performs a DMA cycle using one of the control data structures. After the DMA cycle has completed, the controller performs the next DMA cycle using the other control data structure. The controller signals completion of each DMA cycle, which allows the processor to reconfigure the inactive data structure. The controller continues executing DMA cycles until it reads an “invalid” data structure or until the processor changes the cycle_ctrl field to 3’b001 or 3’b010; 3’b100 Memory “Execution with configuration change” mode. See the corresponding section. When the controller operates in this mode, the value of this field in the primary control data structure must be 3’b100; 3’b101 Memory “Execution with configuration change” mode. See the corresponding section. When the controller operates in this mode, the value of this field in the alternate control data structure must be 3’b101; 3’b110 Peripheral “Execution with configuration change” mode See the corresponding section. When the controller operates in this mode, the value of this field in the primary control data structure must be 3’b110; 3’b111 Peripheral “Execution with configuration change” mode See the corresponding section. When the controller operates in this mode, the value of this field in the alternate control data structure must be 3’b111;

At the start of a DMA cycle, or at the start of a block of 2^R DMA transfers, the controller reads the channel_cfg value from system memory. After 2^R or N transfers have been completed, it stores the updated channel_cfg value back into system memory.

The controller does not support dst_size values that differ from src_size. If the controller detects that these fields are not equal, it uses the src_size value as the data width for both the destination and the source, and at the next update of the n_minus_1 field it also sets the dst_size field equal to src_size.

After the controller has performed N transfers, it sets the cycle_ctrl field to 3'b000, thereby marking the channel_cfg data as "invalid". This prevents repetition of a DMA cycle that has already been executed.

25.5.3.1. *Address calculation*

To calculate the source address for a DMA transfer, the controller shifts the value of n_minus_1 left by the number of bits determined by the src_inc field and then subtracts the resulting value from the source end address pointer value. In a similar way, to calculate the destination address for a DMA transfer, the controller shifts the value of n_minus_1 left by the number of bits determined by the dst_inc field and then subtracts the resulting value from the destination end address pointer value.

Depending on the src_inc and dst_inc field values, the source and destination addresses are computed according to the following equations:

src_inc=b00 and dst_inc=b00

- `source address = src_data_end_ptr – n_minus_1`
- `destination address = dst_data_end_ptr – n_minus_1.`

Src_inc=b01 and dst_inc=b01

- `source address = src_data_end_ptr – (n_minus_1<<1)`
- `destination address = dst_data_end_ptr – (n_minus_1<<1).`

Src_inc=b01 and dst_inc=b10

- `source address = src_data_end_ptr – (n_minus_1<<2)`
- `destination address = dst_data_end_ptr – (n_minus_1<<2).`

src_inc = b11 and dst_inc = b11

- source address = src_data_end_ptr
- destination address = dst_data_end_ptr.

Table 461 shows the destination addresses in a DMA cycle for 6 words.

Table 461 – DMA cycle for 6 words with word increment

Initial channel_cfg values before the DMA cycle				
src_size = 2'b10, dst_inc = 2'b10, n_minus_1 = 3'b101, cycle_ctrl = 1				
DMA transfers	End address pointer	Counter	Offset*	Address
	0x2AC	5	0x14	0x298
	0x2AC	4	0x10	0x29C
	0x2AC	3	0xC	0x2A0
	0x2AC	2	0x8	0x2A4
	0x2AC	1	0x4	0x2A8
	0x2AC	0	0x0	0x2AC
Final channel_cfg values after the DMA cycle				
src_size = 2'b10, dst_inc = 2'b10, n_minus_1 = 0, cycle_ctrl = 0				
* Value obtained by left-shifting the counter value by the number of bits corresponding to dst_inc.				

Table 462 gives the destination addresses for DMA transfers of 12 bytes using a halfword increment.

Table 462 – DMA cycle for 12 bytes with halfword increment

Initial channel_cfg values before the DMA cycle				
src_size = 2'b00, dst_inc = 2'b01, n_minus_1 = 4'b1011, cycle_ctrl = 1, R_power = 2'b11				
DMA transfers	End address pointer	Counter	Offset*	Address
	0x5E7	11	0x16	0x5D1
	0x5E7	10	0x14	0x5D3
	0x5E7	9	0x12	0x5D5
	0x 5E7	8	0x10	0x5D7
	0x 5E7	7	0xE	0x5D9
	0x5E7	6	0xC	0x5DB
	0x5E7	5	0xA	0x5DD
	0x5E7	4	0x8	0x5DF
channel_cfg values after 2^R DMA transfers				
src_size = 2'b00, dst_inc = 2'b01, n_minus_1 = 3'b011, cycle_ctrl = 1, R_power = 2'b11				
DMA transfers	0x 5E7	3	0x6	0x5E1
	0x 5E7	2	0x4	0x5E3
	0x5E7	1	0x2	0x5E5
	0x5E7	0	0x0	0x5E7
Final channel_cfg values after the DMA cycle				
src_size = 2'b00, dst_inc = 2'b01, n_minus_1 = 0, cycle_ctrl = 0**, R_power = 2'b11				

* Value obtained by left-shifting the counter value by the number of bits corresponding to dst_inc.

** After the DMA cycle completes, the controller makes channel_cfg “invalid” by clearing the cycle_ctrl field to 0.

25.6. Description of DMA controller register

This section describes the controller registers and control of the controller through these registers.

The section contains the following information:

- about the controller register model;
- register descriptions.

Basic provisions of the controller register model:

- access to reserved or unused addresses is prohibited, since this can lead to unpredictable controller behavior;
- unused or reserved register bits must be written as zeros and their values ignored on read, except where explicitly stated in this section;
- a system reset or power-on reset sets all registers to 0, except where explicitly stated in this section;
- all registers support both read and write access, except where explicitly stated in this section. Write access updates the contents of a register, and read access returns the contents of a register.

Table 463 – List of controller registers

Name	Offset from base address	Type	Reset value	Description
status	0x000	RO	0x-nn0000*	DMA status register
cfg	0x004	WO	-	DMA configuration register
ctrl_base_ptr	0x008	R/W	0x00000000	Channel control data base address register
alt_ctrl_base_ptr	0x00C	RO	0x000000nn**	Alternate channel control data base address register
waitonreq_status	0x010	RO	0x00000000	Channel wait-on-request status register
chnl_sw_request	0x014	WO	-	Channel software request register
chnl_useburst_set	0x018	R/W	0x00000000	Channel burst request set register
chnl_useburst_clr	0x01C	WO	-	Channel burst request clear register
chnl_req_mask_set	0x020	R/W	0x00000000	Channel service request mask set register
chnl_req_mask_clr	0x024	WO	-	Channel service request mask clear register

Name	Offset from base address	Type	Reset value	Description
chnl_enable_set	0x028	R/W	0x00000000	Channel enable set register
chnl_enable_clr	0x02C	WO	-	Channel enable clear register
chnl_pri_alt_set	0x030	R/W	0x00000000	Channel primary/alternate control data structure set register
chnl_pri_alt_clr	0x034	WO	-	Channel primary/alternate control data structure clear register
chnl_priority_set	0x038	R/W	0x00000000	Channel priority set register
chnl_priority_clr	0x03C	WO	-	Channel priority clear register
-	0x040-0x048		-	reserved
err_clr	0x04C	R/W	0x00000000	Error flag clear register
-	0x050-0xDFC	-		reserved
* The reset value depends on the number of DMA channels implemented in the controller and on the presence of integrated test logic. ** The reset value depends on the number of DMA channels implemented in the controller.				

25.6.1. **DMA status register STATUS**

This register is read-only. On a read, the register returns the controller status. If the controller is in reset, reads from the register are not allowed.

Table 464 – DMA status register

Number	31...28	27...21	20...16	15...8	7...4	3...1	0
Access	RO	U	RO	U	RO	U	RO
Reset	0	0	0	0	0	0	0
	test_status	-	chnls_minus1	-	state	-	master_enable

Table 465 – dma_status register bit assignments

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…28	test_status	Value when read: 4'b0000 = the controller does not have integrated test logic; 4'b0001 = the controller has integrated test logic; 4'b0010 – 4'b1111 = undefined
27…21	-	Undefined
20…16	chnls_minus1	Number of available DMA channels minus 1. For example: 5'b00000 = the controller has 1 DMA channel; 5'b00001 = the controller has 2 DMA channels; 5'b00010 = the controller has 3 DMA channels; … 5'b11111 = the controller has 32 DMA channels
15…8	-	Undefined
7…4	state	Current state of the controller control state machine. The state can be one of the following: 4'b0000 = idle; 4'b0001 = reading channel control data; 4'b0010 = reading source end pointer; 4'b0011 = reading destination end pointer; 4'b0100 = reading source data; 4'b0101 = writing data to destination; 4'b0110 = waiting for a DMA request; 4'b0111 = writing channel control data; 4'b1000 = stalled; 4'b1001 = done; 4'b1010 = peripheral "Execution with configuration change" mode; 4'b1011 – 4'b1111 = undefined
3…1	-	Undefined
0	master_enable	Controller state: 0 = controller operation disabled; 1 = controller operation enabled

25.6.2. **DMA configuration register CFG**

This register is write-only. The register defines the controller state.

Table 466 – DMA configuration register

Number	31...8	7...5	4...1	0
Access	U	WO	U	WO
Reset	0	0	0	0
	-	chnl_prot_ctrl	-	master_enable

Table 467 – dma_cfg register bit assignments

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…8	-	Undefined; 0 must be written.
7…5	chnl_prot_ctrl	Defines indication levels for the AHB-Lite bus protection signals HPROT[3:1]: Bit 7 controls HPROT[3] to indicate cached access; Bit 6 controls HPROT[2] to indicate buffered access; Bit 5 controls HPROT[1] to indicate privileged access. Note: If bit [n] = 1, the corresponding HPROT signal is 1. If bit [n] = 0, the corresponding HPROT signal is 0.
4…1	-	Undefined. 0 must be written.
0	master_enable	Defines the controller state: 0 – disables controller operation; 1 – enables controller operation.

25.6.3. **Channel control data base address register CTRL_BASE_PTR**

This register supports both read and write access. The register defines the base address in system memory where the channel control data is located.

Note: The controller does not contain internal memory for storing channel control data.

The size of the system memory region allocated to the controller depends on the number of DMA channels implemented in the controller and on whether alternate channel control data is used. Therefore, the number of register bits required to specify the base address varies and depends on the system configuration.

If the controller is in reset, reads from the register are not allowed.

Table 468 – Channel control data base address register

Number	31...10	9...0
Access	R/W	U
Reset	0	0
	ctrl_base_ptr	-

Table 469 – ctrl_base_ptr register bit assignments

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…10	ctrl_base_ptr	Pointer to the base address of the primary channel control data structure. See the corresponding section.
9…0	-	Undefined. 0 must be written.

25.6.4. **Alternate channel control data base address register ALT_CTRL_BASE_PTR**

This register is read-only. On a read, the register returns the base address pointer of the alternate channel control data. If the controller is in reset, reads from the register are not allowed. This register eliminates the need to calculate the base address of the alternate channel control data.

Table 470 – Alternate channel control data base address register

Number	31... 0
Access	RO
Reset	0
	alt_ctrl_base_ptr

Table 471 – alt_ctrl_base_ptr register bit assignments

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…0	alt_ctrl_base_ptr	Base address pointer of the alternate control data structure

25.6.5. **Channel wait-on-request status register WAITONREQ_STATUS**

This register is read-only. On a read, the register returns the state of the dma_waitonreq[] signals. If the controller is in reset, reads from the register are not allowed.

Table 472 – Channel wait-on-request status register

Number	31	.....	2	1	0
Access	RO	.....	RO	RO	RO
Reset	0	.....	0	0	0
	dma_waitonreg_ status for dma_waitnreg [31]	.....	dma_waitonreg_ status for dma_waitnreg [2]	dma_waitonreg_ status for dma_waitnreg [1]	dma_waitonreg_status for dma_waitnreg [0]

Table 473 – dma_waitonreq_status register bit assignments

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…0	dma_waitonreq_status	State of the DMA channel wait-on-request signals. On a read: Bit [C] = 1 means that dma_waitonreq[C] is at logic 1; Bit [C] = 0 means that dma_waitonreq[C] is at logic 0.

25.6.6. **Channel software request register CHNL_SW_REQUEST**

This register is write-only. The register makes it possible to generate a software request to perform a DMA cycle.

Table 474 – Channel software request register

Number	31	.....	2	1	0
Access	WO	.....	WO	WO	WO
Reset	0	.....	0	0	0
	chnl_sw_request for channel [31]	.....	chnl_sw_request for channel [2]	chnl_sw_request for channel [1]	chnl_sw_request for channel [0]

Table 475 – chnl_sw_request register bit assignments

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…0	chnl_sw_request	Sets the corresponding bit to generate a software request to perform a DMA cycle on the corresponding DMA channel. When recording: Bit [C] = 0 means that the DMA cycle request for channel C is not set; Bit [C] = 1 means that the DMA cycle request for channel C is set; Writing a bit corresponding to an unimplemented channel means that no DMA cycle request is set.

25.6.7. **Channel use-burst set register CHNL_USEBURST_SET**

This register supports both read and write access. The register disables servicing of single requests asserted via dma_sreq[], and therefore only requests via dma_req[] are processed and executed. On a read, the register returns the current use-burst configuration.

Table 476 – Channel use-burst set register

Number	31	...	2	1	0
Access	R/W	...	R/W	R/W	R/W
Reset	0	...	0	0	0
	chnl_useburst_ set for channel [31]	...	chnl_useburst_ set for channel [2]	chnl_useburst_set for channel [1]	chnl_useburst_set for channel [0]

Table 477 – chnl_useburst_set register bit assignments

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…0	chnl_useburst_set	Disables servicing of DMA cycle requests from dma_sreq[] and returns on a read the state of these settings. On a read: Bit [C] = 0 means that DMA channel C performs DMA cycles in response to requests received from both dma_sreq[] and dma_req[]. The controller performs either single transfers or 2^R transfers. Bit [C] = 1 means that DMA channel C performs DMA cycles only in response to requests received from dma_req[]. The controller performs 2^R transfers. When recording: Bit [C] = 0 has no effect. The chnl_useburst_clr register must be used and the corresponding bit C must be cleared to 0. Bit [C] = 1 disables servicing of DMA cycle requests received from dma_sreq[]. The controller performs 2^R transfers. Writing a bit corresponding to an unimplemented channel has no effect.

After the penultimate transfer out of 2^R transfers, if the number of remaining transfers (N) is less than 2^R, the controller clears the chnl_useburst_set bit to 0. This makes it possible to complete the remaining transfers using dma_sreq[] and dma_req[].

Note: When channel_cfg is programmed with a value of N less than 2^R, it is not allowed to set the corresponding chnl_useburst_set bit if the peripheral block does not support the dma_req[] signal.

In the peripheral “Execution with configuration change” mode, if the next_useburst bit is set in channel_cfg, the controller sets chnl_useburst_set[C] to 1 after completion of a DMA cycle that uses the alternate control data.

25.6.8. **Channel use-burst clear register CHNL_USEBURST_CLR**

This register is write-only. The register enables servicing of single requests asserted via dma_sreq[].

Table 478 – Channel use-burst clear register

Number	31	.....	2	1	0
Access	WO	.....	WO	WO	WO
Reset	0	.....	0	0	0
	chnl_useburst_clr for channel [31]	.....	chnl_useburst_clr for channel [2]	chnl_useburst_clr for channel [1]	chnl_useburst_clr for channel [0]

Table 479 – chnl_useburst_clr register bit assignments

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…0	chnl_useburst_clr	Setting the corresponding bit enables servicing of DMA cycle requests from dma_sreq[]. When recording: Bit [C] = 0 has no effect. The chnl_useburst_set register must be used to disable servicing of requests from dma_sreq[]. Bit [C] = 1 enables servicing of DMA cycle requests received from dma_sreq[]. Writing a bit corresponding to an unimplemented channel has no effect.

25.6.9. **Channel request mask set register CHNL_REQ_MASK_SET**

This register supports both read and write access. The register disables assertion of DMA cycle requests on dma_sreq[] and dma_req[]. On a read, the register returns the mask state for requests from dma_sreq[] and dma_req[] for channel servicing.

Table 480 – Channel request mask set register

Number	31	.....	2	1	0
Access	R/W	.....	R/W	R/W	R/W
Reset	0	.....	0	0	0
	chnl_reg_mask_set for dma_req [31] and dma_sreq [31]	.....	chnl_reg_mask_set for dma_req [2] and dma_sreq [2]	chnl_reg_mask_set for dma_req [1] and dma_sreq [1]	chnl_reg_mask_set for dma_req [0] and dma_sreq [0]

Table 481 – chnl_req_mask_set register bit assignments

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…0	chnl_req_mask_set	Disables servicing of requests from dma_sreq[] and dma_req[] to perform DMA cycles for the channels, and returns on a read the state of these settings. On a read: Bit [C] = 0 means that DMA channel C performs DMA cycles in response to incoming requests; Bit [C] = 1 means that DMA channel C does not perform DMA cycles in response to incoming requests. On a write: Bit [C] = 0 has no effect. The chnl_req_mask_clr register must be used to enable assertion of requests. Bit [C] = 1 disables assertion of DMA cycle requests on dma_sreq[] and dma_req[]. Writing a bit corresponding to an unimplemented channel has no effect.

25.6.10. **Channel request mask clear register CHNL_REQ_MASK_CLR**

This register is write-only. The register enables assertion of DMA cycle requests on dma_sreq[] and dma_req[].

Table 482 – Channel request mask clear register

Number	31	.....	2	1	0
Access	WO	.....	WO	WO	WO
Reset	0	.....	0	0	0
	chnl_reg_mask_clr for dma_req [31] and dma_sreq [31]	.....	chnl_reg_mask_clr for dma_req [2] and dma_sreq [2]	chnl_reg_mask_clr for dma_req [1] and dma_sreq [1]	chnl_reg_mask_clr for dma_req [0] and dma_sreq [0]

Table 483 – chnl_req_mask_clr register bit assignments

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…0	chnl_req_mask_clr	Setting the corresponding bit enables assertion of DMA cycle requests from dma_sreq[] and dma_req[] for the channels. On a write: Bit [C] = 0 has no effect. The chnl_req_mask_set register must be used to disable assertion of requests. Bit [C] = 1 enables assertion of DMA cycle requests on dma_sreq[] and dma_req[]. Writing a bit corresponding to an unimplemented channel has no effect.

25.6.11. **Channel enable set register CHNL_ENABLE_SET**

This register supports both read and write access. This register allows read and write access. On a read, the register returns the enable state of the DMA channels.

Table 484 – Channel enable set register

Number	31	.....	2	1	0
Access	R/W	.....	R/W	R/W	R/W
Reset	0	.....	0	0	0
	chnl_enable_set for channel [31]	.....	chnl_enable_set for channel [2]	chnl_enable_set for channel [1]	chnl_enable_set for channel [0]

Table 485 – chnl_enable_set register bit assignments

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…0	chnl_enable_set	Enables DMA channels and, on a read, returns the state of these settings. On a read: Bit [C] = 0 means that DMA channel C is disabled; Bit [C] = 1 means that DMA channel C is enabled. On a write: Bit [C] = 0 has no effect. The chnl_enable_clr register must be used to disable a channel. Bit [C] = 1 enables DMA channel C. Writing a bit corresponding to an unimplemented channel has no effect.

25.6.12. **Channel enable clear register CHNL_ENABLE_CLR**

This register is write-only. The register disables DMA channels.

Table 486 – Channel enable clear register

Number	31	.....	2	1	0
Access	WO	.....	WO	WO	WO
Reset	0	.....	0	0	0
	chnl_enable_clr for channel 31	.....	chnl_enable_clr for channel 2	chnl_enable_clr for channel 1	chnl_enable_clr for channel 0

Table 487 – chnl_enable_clr register bit assignments

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…0	chnl_enable_clr	Setting the corresponding bit disables the corresponding DMA channel. On a write: Bit [C] = 0 has no effect. The chnl_enable_set register must be used to enable a channel. Bit [C] = 1 disables DMA channel C. Writing a bit corresponding to an unimplemented channel has no effect. Note: The controller can disable a DMA channel by setting the corresponding bit in the following cases: - when a DMA cycle is completed; - when channel_cfg is read with the cycle_ctrl field set to 3'b000; - when an error occurs on the AHB-Lite bus.

25.6.13. **Channel primary/alternate control data structure select register CHNL_PRI_ALT_SET**

This register supports both read and write access. The register enables operation of a DMA channel using the alternate control data structure. On a read, the register returns the state of the DMA channels (which control data structure each DMA channel uses).

Table 488 – Channel primary/alternate control data structure select register

Number	31	.....	2	1	0
Access	R/W	.....	R/W	R/W	R/W
Reset	0	.....	0	0	0
	chnl_pri_alt_set for channel [31]	.....	chnl_pri_alt_set for channel [2]	chnl_pri_alt_set for channel [1]	chnl_pri_alt_set for channel [0]

Table 489 – chnl_pri_alt_set register bit assignments

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…0	chnl_pri_alt_set	Setting the corresponding bit selects the alternate control data for the corresponding DMA channel; a read returns the state of these settings. On a read: Bit [C] = 0 means that DMA channel C uses the primary control data structure; Bit [C] = 1 means that DMA channel C uses the alternate control data structure. On a write: Bit [C] = 0 has no effect. The chnl_pri_alt_clr register must be used to clear bit [C] to 0. Bit [C] = 1 selects the alternate control data structure for DMA channel C. Writing a bit corresponding to an unimplemented channel has no effect. Note: The controller can toggle the value of bit chnl_pri_alt_set[C] in the following cases: - when 4 DMA transfers specified in the primary control data structure have completed while executing a DMA cycle in memory or peripheral “execution with configuration change” mode; - when all DMA transfers specified in the primary control data structure have completed while executing a DMA cycle in “ping-pong” mode; - when all DMA transfers specified in the alternate control data structure have completed while executing a DMA cycle in the following modes: - ping-pong; - memory “execution with configuration change”; - peripheral “Execution with configuration change”

25.6.14. **Channel primary/alternate control data structure clear register CHNL_PRI_ALT_CLR**

This register is write-only. The register enables operation of a DMA channel using the primary control data structure.

Table 490 – Channel primary/alternate control data structure clear register

Number	31	.....	2	1	0
Access	WO	.....	WO	WO	WO
Reset	0	.....	0	0	0
	chnl_pri_alt_clr for channel [31]	.....	chnl_pri_alt_clr for channel [2]	chnl_pri_alt_clr for channel [1]	chnl_pri_alt_clr for channel [0]

Table 491 – chnl_pri_alt_clr register bit assignments

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…0	chnl_pri_alt_clr	Setting the corresponding bit selects the primary control data for the corresponding DMA channel. On a write: Bit [C] = 0 has no effect. The chnl_pri_alt_set register must be used to select the alternate control data. Bit [C] = 1 selects the primary control data structure for DMA channel C. Writing a bit corresponding to an unimplemented channel has no effect. Note: The controller can toggle the value of bit chnl_pri_alt_clr[C] in the following cases: - when 4 DMA transfers specified in the primary control data structure have completed while executing a DMA cycle in memory or peripheral “execution with configuration change” mode; - when all DMA transfers specified in the primary control data structure have completed while executing a DMA cycle in “ping-pong” mode; - when all DMA transfers specified in the alternate control data structure have completed while executing a DMA cycle in the following modes: - ping-pong - memory “execution with configuration change” - peripheral “Execution with configuration change”

25.6.15. **Channel priority set register CHNL_PRIORITY_SET**

This register supports both read and write access. The register is used to assign a high priority level to a DMA channel. On a read, the register returns the current priority state of the DMA channels.

Table 492 – Channel priority set register

Number	31	...	2	1	0
Access	R/W	…	R/W	R/W	R/W
Reset	0	...	0	0	0
	chnl_priorit_set for channel [31]	...	chnl_priority_set for channel [2]	chnl_priority_set for channel [1]	chnl_priority_set for channel [0]

Table 493 – chnl_priority_set register bit assignments

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…0	chnl_priority_set	Assigns a high priority level to a DMA channel; on a read, returns the DMA channel priority state. On a read: Bit [C] = 0 means DMA channel C has the default priority level; Bit [C] = 1 means DMA channel C has a high priority level. On a write: Bit [C] = 0 has no effect. The chnl_priority_clr register must be used to set channel C to the default priority level. Bit [C] = 1 assigns a high priority level to DMA channel C. Writing a bit corresponding to an unimplemented channel has no effect.

25.6.16. **Channel priority clear register CHNL_PRIORITY_CLR**

This register is write-only. The register is used to assign the default priority level to a DMA channel.

Table 494 – Channel priority clear register

Number	31	...	2	1	0
Access	WO	...	WO	WO	WO
Reset	0	...	0	0	0
	chnl_priorit_clr for channel [31]	...	chnl_priority_clr for channel [2]	chnl_priority_clr for channel [1]	chnl_priority_clr for channel [0]

Table 495 – chnl_priority_clr register bit assignments

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…0	chnl_priority_clr	Setting a bit assigns the corresponding DMA channel the default priority level. On a write: Bit [C] = 0 has no effect. The chnl_priority_set register must be used to assign a high priority level to channel C. Bit [C] = 1 assigns DMA channel C the default priority level. Writing a bit corresponding to an unimplemented channel has no effect.

25.6.17. **Error flag clear register ERR_CLR**

This register supports both read and write access. The register is used to clear the dma_err signal to 0. On a read, the register returns the state of the dma_err signal.

Table 496 – Error flag clear register

Number	31…1	0
Access	U	R/W
Reset	0	0
	-	err_clr

Table 497 – err_clr register bit assignments

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…1	-	Undefined. 0 must be written.
0	err_clr	Clears the signal to 0; on a read, returns the state of the dma_err signal (flag). On a read: Bit [C] = 0 means dma_err is in state 0; Bit [C] = 1 means dma_err is in state 1. On a write: Bit [C] = 0 has no effect. The dma_err state remains unchanged. Bit [C] = 1 clears the dma_err signal (flag) to state 0. Note: If the dma_err signal is cleared at the same time as an error occurs on the AHB-Lite bus, the error has priority, and therefore the register value (and dma_err) remains unchanged (not cleared).

26 Interrupts

According to the privileged command set specification, the RISC-V core must support handling of machine software interrupts and machine timer interrupts. Generation of requests (signals) for these interrupts is performed by the Core Local Interruptor (CLINT). CLINT contains registers for controlling software interrupts and timer interrupts. The CLINT register map is given in Table 498

Table 498 – CLINT registers

Address	Name	Access
0x0200_0000	Software interrupt request/clear register	R/W
0x0200_4000	Timer compare register	R/W
0x0200_BFF8	Timer value register	R/W

Table 499 – Interrupt request/clear register

Bits	Name	Access	Description	Value
0	msip	R/W	Software interrupt pending	0

Table 500 – Timer value register

Bits	Name	Access	Description	Value
31…0	mtime_lo	R/W	Least significant part of timer value	0
64…32	mtime_hi	R/W	Upper part of timer value	0

The register holds a 64-bit timer value. The value is incremented by the timer_pulse signal (rising edge of HCLK divided by the value programmed in the DIV_SYS_TIM register) of the external processor-complex interface.

Table 501 – Timer compare register

Bits	Name	Access	Description	Value
31…0	mtimecmp_lo	R/W	Least significant part of the compare register value	0
64…32	mtimecmp_hi	R/W	Upper part of the compare register value	0

The register holds a 64-bit compare value for the timer. A timer interrupt request is asserted if the timer value is strictly greater than the compare value.

27 Core local interrupt controller CLIC

The core local interrupt controller (CLIC) receives interrupt signals and delivers the interrupt to the core for servicing. The CLIC controller supports 47 interrupt inputs to the core. Each interrupt input i has four 8-bit control registers mapped in memory: the interrupt pending bit clicintip[i], the interrupt enable bit clicintie[i], the interrupt attributes clicintattr[i] to specify privilege mode and trigger type, and the interrupt control bits clicintctl[i] to specify level and priority. The first 16 interrupt inputs are connected to basic local interrupt signals. The remaining interrupt inputs are connected to interrupt signals generated by peripheral blocks.

27.1. Interrupt preemption

The CLIC controller implements interrupt preemption based on partitioning interrupts into levels. Up to 16 interrupt levels are supported for each privilege mode, with higher-numbered interrupt levels allowed to preempt lower-numbered levels. Interrupt level 0 corresponds to normal execution outside an interrupt handler. Levels 1-15 correspond to interrupt-handler levels. Incoming interrupts with a higher level can preempt an active interrupt handler running at a lower level in the same privilege mode, provided that interrupts are globally enabled in that privilege mode. Similarly, incoming interrupts for a higher privilege mode can preempt an active interrupt handler running in a lower privilege mode, regardless of the global interrupt enable in the lower privilege mode.

Software handling of interrupt preemption is described in the document “Smclic Core-Local Interrupt Controller (CLIC) RISC-V Privileged Architecture Extension”, version 0.9-12/21/2021.

27.2. Interaction of the CLIC controller with other local interrupts

The CLIC controller includes the functionality of the basic local interrupts, so the basic local interrupt control bits are not mapped to the least significant 16 bits of the mip/mie (uip/uie) CSR registers. Timer interrupts (mtip/utip), software interrupts (msip/usip), and external interrupt inputs (meip/ueip) are treated as additional local interrupt sources, where privilege mode, interrupt level, and priority can be changed using the clicintattr[i] and clicintctl[i] registers. In CLIC mode, interrupt delegation for these signals is achieved by changing the interrupt privilege mode in the CLIC interrupt attribute register clicintattr, as for any other CLIC interrupt input.

27.3. Description of CLIC controller registers

Information on the CLIC controller registers is given in Table 502.

Table 502 – CLIC controller registers

Address	Name	Access	Description
0x0D00_0000	CLIC		CLIC controller
Offset			
0x0000	cliccfg	R/W	Control register
0x0004	clicinfo	R/O	Information register
0x0008 – 0x0FFC	–	–	Reserved
0x1000 + 4 · i*	clicintip[i]	R/W	Interrupt present register
0x1001 + 4 · i*	clicintie[i]	R/W	Interrupt enable register
0x1002 + 4 · i*	clicintattr[i]	R/W	Interrupt attribute register
0x1003 + 4 · i*	clicintctl[i]	R/W	Interrupt level control register
* i is the interrupt number from 0 to 46.			

Access to the cliccfg and clicinfo registers is only allowed from Machine privilege mode. Access to these registers from User privilege mode generates an exception.

From Machine privilege mode, all clicintip[i], clicintie[i], clicintattr[i], and clicintctl[i] registers are accessible. Access to the interrupt i registers from User privilege mode is allowed only if the corresponding interrupt i in the clicintattr[i] register, field mode[1:0], is configured for User mode. If interrupt i is configured for Machine mode, then from User privilege mode reads of the registers of interrupt i return zeros and writes are not performed.

27.3.1. **cliccfg configuration register**

The single 8-bit global configuration register cliccfg defines how many privilege modes are supported, how the clicintctl[i] registers are divided into level and priority fields, and whether selective hardware interrupt vectoring is supported. The cliccfg register has three fields:

- 2-bit field nmbits;
- 4-bit field nlbits;
- 1-bit field nvbits.

Table 503 – cliccfg register

Number	7	6	5	4..1	0
Access	U	U	R/W	R/W	R/O
Reset	0	0	0	XXXX	1
		nmbits[1]	nmbits[0]	nlbits[3:0]	nvbits

Table 504 – Description of cliccfg register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
7	–	Reserved. It is recommended to leave the value of this field unchanged.
6	nmbits[1]	Reserved
5	nmbits[0]	Number of hardware bits implemented in clicintattr[i].mode to represent the privilege mode of input i
4…1	nlbits[3:0]	Number of most significant bits in the clicintctl[i] register allocated to define the interrupt level. This field must be initialized by software
0	nvbits	Selective hardware interrupt vectoring function in the CLIC controller: 1 – function implemented; 0 – function not implemented

27.3.1.1. **Determining interrupt privilege mode**

The 2-bit field cliccfg.nmbits indicates how many bits are physically implemented in clicintattr[i].mode to represent the privilege mode of input i. Although the cliccfg.nmbits field always has a width of 2 bits, only one bit in this field is physically implemented (M/U interrupt privilege modes are supported).

In systems with M/U modes and support for user interrupts with the N extension, cliccfg.nmbits can be set to 0 or 1. If cliccfg.nmbits = 0, all interrupts are handled as M-mode interrupts. If cliccfg.nmbits = 1, value 1 in the most significant bit (MSB) of the clicintattr[i].mode register indicates that the interrupt input is handled in M mode, while value 0 indicates that the interrupt is handled in U mode.

Table 505 – Explanation of privilege levels (mstatus.mpp)

Level	Code	Name	Abbreviation
0	00	User/Application	U
1	01	Reserved	–
2	10	Reserved	–
3	11	Machine	M

Table 506 – Privilege modes

Privilege mode	nmbits	clicintattr[i].mode	Interpretation
M/U	0	xx	M-mode interrupt
M/U	1	0x	U-mode interrupt
M/U	1	1x	M-mode interrupt

27.3.1.2. **Determining interrupt level**

4 most significant bits are implemented in the clicintctl[i] register and are intended for configuring the interrupt level and priority. The least significant 4 bits of the clicintctl[i] register are not implemented and are hardwired to logical “1”. The number of bits actually implemented in the clicintctl[i] register can also be read from the clicinfo register, field CLICINTCTLBITS.

The 4-bit field cliccfg.nlbits indicates how many most significant bits in the clicintctl[i] register are allocated to define the interrupt level. This field must be initialized by software. In the current implementation this field can be set from 0 to 8. A value of cliccfg.nlbits > CLICINTCTLBITS is implemented identically to cliccfg.nlbits = CLICINTCTLBITS.

Table 507 shows how interrupt levels are encoded depending on the value set in cliccfg.nlbits.

Table 507 – Privilege Modes

Value in cliccfg.nlbits	Code	Interrupt levels
0	pppp....	255
1	lppp....	127, 255
2	llpp....	63, 127, 191, 255
3	lllp....	31, 63, 95, 127, 159, 191, 223, 255
4-8	llll....	15, 31, 47, 63, 79, 95, 111, 127, 143, 159, 175, 191, 207, 223, 239, 255
Note: Designations in the “Code” column: “l” – modifiable bits available for setting the level; “p” – modifiable bits available for setting the priority; “.” – non-existent bits for level encoding, assumed to be set to “1”.		

If nlbits = 0, all interrupts are treated as having level 255.

27.3.1.3. **Determining interrupt priority**

The least significant bits in clicintctl[i] that are not configured as part of the interrupt level are used for the interrupt priority. They are used to determine priority among pending and enabled interrupts in the same privilege mode and at the same interrupt level. The number of available interrupt-priority bits can be determined by subtracting the value of cliccfg.nlbits from clicinfo.CLICINTCTLBITS (a negative value obtained after subtraction is equivalent to zero priority bits).

The interrupt with the highest priority for a given privilege mode and interrupt level is serviced first. If there are several pending and enabled interrupts with the same highest priority, the interrupt with the highest number (Table 536) is serviced first.

Note that the interrupt level is used to determine preemption (for nested interrupts). By contrast, interrupt priority does not affect preemption and is used only as additional ranking when several pending interrupts have the same interrupt level.

27.3.1.4 **Hardware vectoring of selective interrupts**

The selective hardware vectoring function gives users flexibility to choose the behavior for each interrupt: either hardware-vectored or non-vectored. As a result, each interrupt can be optimized and can take advantage of both behaviors. Hardware vectoring provides faster interrupt response at the cost of a small increase in code size (for saving/restoring context). The non-vectored option, on the other hand, provides smaller code size (through sharing and reuse of a single copy of common save/restore code) at the cost of slower interrupt response.

In the cliccfg register, the nvbits bit is set to 1, which indicates the presence of the selective hardware interrupt vectoring function in the CLIC controller. The clicintattr[i].shv bit controls the vectoring behavior of interrupt i. If clicintattr[i].shv is 0, the interrupt is non-vectored and always branches to the common code at the address specified in the mtvec (utvec) CSR register. If clicintattr[i].shv is 1, the branch is made to the interrupt handler function address, which is read in hardware from the interrupt vector table. The base address of the interrupt vector table is set in the mtvt (utvt) CSR register. This allows some interrupts to branch to the common base address stored in mtvec (utvec), while others are handled in hardware through the table pointed to by the additional mtvt (utvt) CSR register.

27.3.2. **Information register clicinfo**

The clicinfo register is read-only and shows debug information.

Table 508 – clicinfo register

Number	31	30...25	24…21	20...13	12…0
Access	U	R/O	R/O	R/O	R/O
Reset	0	000000	0100	00000000	00101111
		num_trigger	CLICINTCTLBITS	version	num_interrupt

Table 509 – Description of clicinfo register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31	-	Reserved
30…25	num_trigger	Maximum number of interrupt triggers
24...21	CLICINTCTLBITS	Number of hardware bits implemented in the clicintctl[i] registers
20…13	version	CLIC implementation version
12…0	num_interrupt	Maximum number of interrupt input signals

The num_interrupt field indicates the actual maximum number of interrupt input signals supported in this implementation.

The version field indicates the CLIC implementation version. Four most significant bits specify the architecture version, and the four least significant bits specify the implementation version.

The CLICINTCTLBITS field indicates how many hardware bits are actually implemented in the clicintctl[i] registers. The implemented bits are stored left-aligned in the most significant bits of each 8-bit clicintctl[i] register, with the least significant non-implemented bits hardwired to logical "1".

The num_trigger field indicates the maximum number of interrupt triggers supported in this implementation.

27.3.3. **Interrupt pending register clicintip**

Each interrupt input signal has a dedicated interrupt pending bit clicintip[i]. The clicintip register occupies one byte in the memory map. The pending bit is located in bit 0 of the byte.

Table 510 – clicintip register

Number	7...1	0
Access	U	R/O
Reset	-	X
		clicintip

Table 511 – Description of bits in the clicintip register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
7...1	-	Reserved
0	clicintip	Interrupt pending bit: 0 – no interrupt pending; 1 – interrupt pending

Software must interpret clicintip[i] = 0 as no interrupt pending, and clicintip[i]! = 0 as an interrupt pending.

When an input is configured as level-triggered, the bit clicintip[i] reflects the value of the interrupt input to the controller after the conditional inversion specified in the clicintattr[i] field, and software writes to the bit are ignored. Software clears the interrupt at the source device.

When an input is configured as edge-triggered, clicintip[i] is a read/write register that can be updated by both hardware interrupts and software. The bit is set by hardware after an edge of the corresponding polarity is observed on the interrupt input, as defined by the clicintattr[i] field. The corresponding interrupt pending bit is cleared by hardware when the interrupt is serviced in vectored mode. Additional information on hardware clearing is given in Section 27.4.1 "mtvec, utvec registers". Software writes can set or clear edge-triggered interrupt pending bits directly by writing to the memory-mapped register. Edge-triggered interrupt pending bits can also be cleared by writing to the mnxti (unxti) CSR register.

For performance, when vectored interrupt mode is selected and the interrupt is serviced, the corresponding edge-triggered pending bit is cleared by hardware, so software does not need to clear the pending bit in the handler.

In contrast, when a non-vectored interrupt (common code) is selected, the edge-triggered pending bit is not cleared by hardware. Software is expected to use a write to the mnxti (unxti) CSR register to clear the edge-triggered pending bit in non-vectored mode. Additional information on this is given in the description of the mnxti (unxti) register.

The value in clicintip[i] is undefined when switching from level-triggered mode to edge-triggered mode in clicintattr[i]. Software cannot rely on the preserved state of the clicintip[i] bits for edge-triggered mode when switching to level-triggered mode.

27.3.4. **Interrupt enable register clicintie**

Each interrupt input has a dedicated interrupt enable bit clicintie[i]. The clicintie register occupies one byte in the memory map. The control bit clicintie[0] is readable and writable and is used to enable or disable the corresponding interrupt.

Table 512 – clicintie register

Number	7…1	0
Access	U	R/W
Reset	-	0
		clicintie

Table 513 – Description of bits in the clicintie register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
7…1	-	Reserved
0	clicintie	Interrupt enable bit: 0 – interrupt disabled; 1 – interrupt enabled

Software must interpret clicintie[i] = 0 as interrupt disabled, and clicintie[i]! = 0 as interrupt enabled.

The bit clicintie[i] is an individual enable bit, while mstatus.xie (ustatus.uie) is the global enable bit for the current privilege mode. Therefore, to enable interrupt i in the current privilege mode, both clicintie[i] and mstatus.xie (ustatus.uie) must be set.

Because mstatus.xie (ustatus.uie) is effective only in the current privilege mode according to the RISC-V convention, interrupt i from a more privileged mode is enabled as long as clicintie[i] is set (regardless of the mstatus.xie (ustatus.uie) setting in more privileged modes).

27.3.5. **Interrupt attribute register clicintattr**

The clicintattr register is an 8-bit read/write register that defines various attributes for each interrupt.

Table 514 – clicintattr register

Number	7...6	5…3	2...1	0
Access	R/W	U	R/W	R/W
Reset	11	000	00	X
	mode		trig	shv

Table 515 – Description of bits in the clicintattr register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
7…6	mode	Interrupt privilege mode: 00 – U-mode; 01, 10 – reserved; 11 – M-mode
5...3	-	Reserved
2...1	trig	Defines trigger type and polarity for each interrupt input signal: 00 – high-level interrupt trigger; 01 – rising-edge interrupt trigger; 10 – low-level interrupt trigger; 11 – falling-edge interrupt trigger
0	shv	Selection of vectored interrupt mode: 0 – interrupt is non-vectored; 1 – interrupt is vectored

The single-bit shv field is used to select vectored interrupt mode. If shv = 0, the interrupt is non-vectored and branches to the common code at mtvec (utvec). If shv = 1, vectored mode is selected for the interrupt, and control automatically branches to the interrupt handler function pointer specified in the mtvt (utvt) CSR. This option allows some interrupts to branch to the common base address stored in mtvec (utvec), while other interrupts are handled in hardware through the table pointed to by the additional mtvt (utvt) CSR.

If cliccfg.nvbits = 0, the selective interrupt vectorization function is not available, and the shv field is hardwired to logic “0”.

The two-bit trig field defines the trigger type and polarity for each interrupt input signal. Bit 0, trig[0], selects level or edge (0: level-triggered, 1: edge-triggered), while bit 1, trig[1], selects edge polarity (0: rising edge, 1: falling edge). There are four possible combinations: high-level trigger, low-level trigger, rising-edge trigger, and falling-edge trigger.

Notes

1 A level-triggered interrupt is taken for servicing when the execution subsystem pipeline of the processor core subsystem becomes free. The number of cycles until the execution subsystem pipeline becomes free depends on the type of instruction already accepted for execution. Number of cycles:

– for DIV, DIVU, REM, REMU instructions: from 2 to 16 cycles (depends on operands);
– for CLMUL, CLMULH and CLMULR instructions: 8 cycles;
– for load/store instructions accessing the TCM region: 1 cycle;
– for read/write instructions accessing the IO region: more than 1 cycle (depends on the peripheral being accessed (APB, AHB), see Figure 14);
– for all other instructions: 1 cycle.

2 For interrupts that clear the request in hardware (such as DMA, EXT_INTx and the system timer), to prevent loss of interrupts due to the request being de-asserted before it is taken for servicing, it is recommended to configure edge-triggered generation.

The two-bit mode field specifies the privilege mode in which this interrupt operates. This field uses the same encoding as mstatus.mpp (11: M-mode, 00: U-mode). The valid width of this field can be programmed using cliccfg.nmbits.

For safety reasons, the mode field can only be set to a privilege level that is equal to or lower than the current privilege level.

27.3.6. **Interrupt input control register clicintctl**

The clicintctl[i] register is an 8-bit read/write control register. It defines the interrupt level and interrupt priority.

Table 516 – clicintctl register

Number	7..4	3..0
Access	R/W	U
Reset	XXXX	1111
	clicintctl	

Table 517 – Description of the clicintctl register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
7…4	clicintctl	Interpreted as interrupt level and interrupt priority
3…0	–	Reserved

The number of bits actually implemented in this register is defined by the fixed parameter CLICINTCTLBITS (in clicinfo), which has a value from 0 to 8. The implemented bits are stored left-aligned in the most significant bits of each 8-bit clicintctl[i] register, while the least significant unimplemented bits are fixed to logic 1. These control bits are interpreted as interrupt level and interrupt priority according to the settings in the CLIC configuration register cliccfg.nlbits.

To select an interrupt for delivery to the core, the CLIC hardware combines the valid bits from clicintattr.mode and clicintctl into an unsigned integer and then selects the global maximum for all pending and enabled interrupts based on this value. Then the cliccfg setting defines how the clicintctl value is split into interrupt level and interrupt priority. Finally, the level of the selected interrupt is compared with the threshold value for the corresponding privilege mode to determine whether it is allowed or masked by the threshold (the interrupt is not serviced).

Selecting an interrupt in a more privileged mode masks any interrupt in a less privileged mode, because a more privileged mode indicates that the interrupt request is more urgent than any interrupt in a less privileged mode.

27.3.6.1. **Interrupt input identification number**

The 47 inputs are assigned unique identification numbers (Table 536), which are used in the mcause (ucause) CSR, in the exccode field. To maintain backward compatibility, the base interrupts keep their original values, while new interrupts are numbered starting from 16.

27.4. **Control and status registers (CSR)**

This subsection describes the control and status registers (CSR) involved in interrupt configuration.

The list of implemented control and status registers that are accessible from Machine privilege mode is given in Table 518.

Table 518 – Control and status registers CSR (Machine)

Address	Name	Access	Description
0x305	mtvec	R/W	Address of the common interrupt vector
0x307	mtvt	R/W	Address of the table containing the addresses of vectored interrupt handlers
0x308	mclicbase	R/O	Base address of the local interrupt controller
0x345	mnxti	R/W	Address of the next interrupt handler and interrupt enable control
0x346	mintstatus	R/O	Interrupt status
0x347	mintthresh	R/W	Interrupt level threshold control
0x348	mscratchcsw	R/W	Scratch register with privilege mode check
0x349	mscratchcswl	R/W	Scratch register with interrupt level check

The list of implemented control and status registers that are accessible from User privilege mode is given in Table 519.

Table 519 – Control and status registers CSR (User)

Address	Name	Access	Description
0x005	utvec	R/W	Address of the common interrupt vector
0x007	utvt	R/W	Address of the table containing the addresses of vectored interrupt handlers
0x045	unxti	R/W	Address of the next interrupt handler and interrupt enable control
0x046	uintstatus	R/O	Interrupt status
0x047	uintthresh	R/W	Interrupt level threshold control
0x048	uscratchcsw	R/W	Scratch register with privilege mode check
0x049	uscratchcswl	R/W	Scratch register with privilege level check

27.4.1. **mtvec, utvec registers**

The 32-bit read/write register holds the base address of the common interrupt vector, aligned to a 64-byte boundary, as well as the interrupt handling mode.

Table 520 – mtvec, utvec register

Number	31...6	5...2	1...0
Access	R/W	U	R
Reset	0	0000	11
	BASE		MODE

Table 521 – Description of the mtvec, utvec registers bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...6	BASE	Base address of the common interrupt vector
5…2	-	Reserved
1…0	MODE	Interrupt handling mode: 00 – non-vectored interrupt handling mode; 01 – vectored interrupt handling mode; 10 – reserved; 11 – interrupt handling mode using CLIC

Only the interrupt handling mode using CLIC is implemented. In this mode, all exceptions and all interrupts configured for non-vectored mode (clicintattr[i].shv = 0) branch to the common interrupt vector.

27.4.2. **mtvt, utvt registers**

The 32-bit read/write register holds the base address of the interrupt vector table, aligned to a 64-byte boundary. The actual alignment can be determined by writing ones to the least significant bits and then reading them back. Values with 0 in the least significant six bits are reserved.

Table 522 – mtvt, utvt register

Number	31...6	5...0
Access	R/W	U
Reset	X	000000
	mtvt, utvt	

Table 523 – Description of the mtvt, utvt registers bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...6	mtvt, utvt	Base address of the interrupt vector table
5…0	-	Reserved

27.4.3. **mnxti, unxti registers**

Table 524 – mnxti, unxti register

Number	31…0
Access	R/W
Reset	0
	mnxti, unxti

Table 525 – Bit description of the mnxti, unxti registers

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…0	mnxti,unxti	Pointer to an entry in the interrupt handler table

The register can be used by software to service the next interrupt for the same privilege mode, if it has a level higher than the level of the saved interrupt context (stored in mcause.pil (ucause.pil)) and a threshold level greater than the interrupt threshold of the corresponding privilege mode, without additional overhead for clearing the interrupt pipeline and saving/restoring context. The register is intended to be accessed using CSRRSI/CSRRCI instructions, where the value read is a pointer to an entry in the interrupt handler table and the write-back updates the interrupt enable status. In addition, a write to the register has side effects that update the interrupt context state.

Unlike a normal CSR instruction, the value returned differs from the value used in the read-modify-write operation.

All CSR instructions can be used with this register. The operation will be the same as for the corresponding CSR instruction used with the mstatus (ustatus) register.

Reading the register returns either zero, indicating that there is no suitable interrupt to service, or that the highest-ranked interrupt is SHV, or that the system is not in CLIC mode, or it returns a non-zero address of the interrupt handler table for software interrupt vectors.

The mtvt (utvt) registers can be set to memory addresses such that the table entry is at address zero, which is indistinguishable from the case where there is no interrupt.

If a CSR instruction that accesses mnxti (unxti) contains a write operation, then a read-modify-write operation is performed only for the mstatus (ustatus) register, while the exccode field of the mcause (ucase) register and the xil field of the mintstatus (uintstatus) register can also be updated with a new interrupt ID and level, and the pending bit of the taken interrupt is cleared accordingly.

If the CSR instruction does not contain a write operation (for example, csrr t0, mnxti), then no state update is performed for any CSR. This can be used to determine whether an interrupt can be taken without actually updating xil and exccode.

The mnxti (unxti) registers are intended to be used inside an interrupt handler after the initial interrupt has been taken and the mcause (ucase) and mepc (uepc) registers have been updated with the interrupt context and interrupt ID.

If the pending interrupt is edge-triggered, hardware automatically clears the corresponding pending bit when a CSR write instruction accesses the mnxti (unxti) register. However, if the CSR instruction does not contain a write (for example, csrr t0, mnxti), then no CSR state is updated and, therefore, the interrupt pending bit is not cleared. This behavior allows software to optimize selection and execution of interrupts using mnxti (unxti).

27.4.4. **mintstatus, uintstatus registers**

The mintstatus register stores the active interrupt level for each privilege level. The mintstatus register fields are read-only. The main reason for providing access to these fields is to support debugging.

Table 526 – mintstatus register

Number	31...24	23...8	7...0
Access	R/O	U	R/O
Reset	0	-	0
	mil		uil

Table 527 – Bit description of the mintstatus register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...24	mil	Active interrupt level for Machine privilege level
23...8	-	Reserved
7...0	uil	Active interrupt level for User privilege level

The uintstatus register provides limited access to mintstatus.

Table 528 – uintstatus register

Number	31...8	7...0
Access	U	R/O
Reset	-	0
		uil

Table 529 – Bit description of the uintstatus register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…8	-	Reserved
7…0	uil	Active interrupt level for User privilege level

27.4.5. **mintthresh, uintthresh registers**

The interrupt threshold level register is read/write and contains an 8-bit field (th) that specifies the threshold level for the corresponding privilege mode. The th field is stored in the least significant 8 bits; zero must be written to the most significant bits.

Table 530 – mintthresh, uintthresh register

Number	31...8	7...0
Access	U	R/W
Reset	0	0
		th

Table 531 – Bit description of the mintthresh, uintthresh registers

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…8	-	Reserved
7...0	th	Interrupt threshold level

Typically, the interrupt threshold level is used to implement critical sections. The current handler can temporarily raise its effective interrupt level to implement a critical section over a subset of levels, while still allowing higher interrupt levels to preempt the current interrupt.

The effective interrupt level is calculated as:

effective_level = max(mintstatus.xil, mintthresh.th),

effective_level = max(uintstatus.xil, uintthresh.th).

The max value is used to prevent the level from falling below its original value, which would violate privileges, and also to simplify removal of the threshold by software without knowing its own level, simply by writing zero.

The interrupt threshold value is valid only when running in the associated privilege mode, but not in other modes. This is because interrupts for lower-privilege modes are always disabled, while interrupts for higher-privilege modes are always enabled. For example, Machine-mode interrupts will not be masked by setting a threshold in User mode. This is similar to how mstatus.mie works: it does not mask Machine-mode interrupts when running in lower-privilege modes.

Such behavior significantly simplifies hardware implementation, since only one global maximum interrupt needs to be selected and compared with the threshold value for the corresponding privilege mode (ignoring threshold values in other modes). Otherwise, hardware would have to select several maximum interrupts (one per privilege mode), compare and match them with the corresponding threshold values, and then choose the maximum interrupt with the highest privilege mode.

27.4.6. **mclicbase register**

The mclicbase register is a 32-bit read-only register that provides the base address of the CLIC memory-mapped registers. Its value must be configured or set at the platform hardware level to indicate the start address of the CLIC memory-mapped registers.

Table 532 – mclicbase register

Number	31…0
Access	R/O
Reset	0xD000000
	mclicbase

Table 533 – Bit description of the mclicbase register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...0	mclicbase	Base address of the CLIC memory-mapped registers

Since the CLIC memory map must be aligned to a 4-KB boundary, the twelve least significant bits in the mclicbase register are hardwired to logic 0. Software uses the register to determine the location of the CLIC memory-mapped registers.

27.4.7. **mscratchcswl, uscratchcswl registers**

The register is provided for fast exchange of the stack pointer between interrupt code and non-interrupt code running in the same privilege mode.

Table 534 – mscratchcswl, uscratchcswl register

Number	31...0
Access	R/W
Reset	X
	mscratchcswl, uscratchcswl

Table 535 – Bit description of the mscratchcswl, uscratchcswl registers

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...0	mscratchcswl, uscratchcswl	Stack pointer value

Example:

```
csrrw rd, mscratchcswl, rs1
//Pseudocode operation.
if ((mcause.pil==0) != (mintstatus.mil==0)) then {
    t = rs1;
    rd = mscratch;
    mscratch = t;
} else {
    rd = rs1;//mscratch unchanged.
}
//Usual use: csrrw sp, mscratchcswl, sp
```

The mscratchcswl (uscratchcswl) register operates similarly to the mscratchcsw (uscratchcsw) register, except that the swap condition is true when the interrupter and the interrupted context are not both application tasks or are not both interrupt handlers.

As with mscratchcsw (uscratchcsw), only the read-modify-write (swap/CSRRW) operation is defined for mscratchcswl (uscratchcswl).

Implementations of non-CSRRW variants (when rd or rs1 is x0, or rs1 is an immediate operand) for mscratchcswl (uscratchcswl) are undefined/reserved.

27.5. **Local interrupt map**

Table 536 – Interrupts generated by peripheral blocks

Number	Interrupts	Block	Generation method
0	usip	Core	Software interrupt for User privilege mode
1	-	-	Reserved
2	-	-	Reserved
3	msip	Core	Software interrupt for Machine privilege mode
4	utip	System timer	Timer interrupt for User privilege mode
5	-	-	Reserved
6	-	-	Reserved
7	mtip	System timer	Timer interrupt for Machine privilege mode
8	-	-	Reserved
9	-	-	Reserved
10	-	-	Reserved
11	-	-	Reserved
12	csip	Core	CLIC software interrupt
13	-	-	Reserved
14	-	-	Reserved
15	-	-	Reserved
16	int_local0	DMA	DMA interrupts: DMA_ERR or DMA_DONE. DMA interrupt is handled in accordance with subclause 25.4.5.7 “Error indication”.
17	int_local1	UART1	UARTINTR signal
18	int_local2	UART2	UARTINTR signal
19	int_local3	SSP1	SSPINTR signal
20	int_local4	POWER	Interrupt signal from the POWER Detector
21	int_local5	WWDG	Interrupt signal from the WWDG
22	int_local6	Timer1	Interrupt signal from the timer via TIM_STATUS and TIM_IE
23	int_local7	Timer2	Interrupt signal from the timer via TIM_STATUS and TIM_IE
24	int_local8	ADC	ADC interrupt signals EOCIF_1 or AWOIF_1 or EOCIF_2 or AWOIF_2
25	int_local9	I2C	I2C_INT signal
26	int_local10	BACKUP	Interrupt from BKP and the real-time clock
27	int_local11	External Interrupt 1	EXT_INT1 signal Pins PA[10] and PD[15] in primary function mode
28	int_local12	External interrupt 2	EXT_INT2 signal Pin PB[6] in alternate function mode and pin PC[4] in primary function mode
29	int_local13	External Interrupt 3	EXT_INT3 signal Pin PB[7] in alternate function mode and pin PC[5] in primary function mode
30	int_local14	ADCUI	Interrupt from sigma-delta ADC block F0 for voltage and current measurement
31	int_local15	ADCUI	Interrupt from sigma-delta ADC block F1 for voltage and current measurement
32	int_local16	ADCUI	Interrupt from sigma-delta ADC block F2 for voltage and current measurement
33	int_local17	-	Software interrupt
34	int_local18	-	Software interrupt

Number	Interrupts	Block	Generation method
35	int_local19	-	Software interrupt
36	int_local20	-	Software interrupt
37	int_local21	RANDOM	Random number generator block interrupt
38	int_local22	ISO7816	UART ISO7816 block interrupt
39	int_local23	UART3	UARTINTR signal
40	int_local24	SSP2	SSPINTR signal
41	int_local25	SSP3	SSPINTR signal
42	int_local26	Timer3	Interrupt signal from the timer via TIM_STATUS and TIM_IE
43	int_local27	Timer4	Interrupt signal from the timer via TIM_STATUS and TIM_IE
44	int_local28	UART4	UARTINTR signal
45	int_local29	-	Software interrupt
46	int_local30	-	Software interrupt

28 Debug subsystem

The core debug subsystem complies with the RISC-V External Debug Support Version 0.13 specification. The following options are supported:

– reading/writing general-purpose registers using abstract commands;
– read/write access to a 64-byte program buffer;
– four abstract data registers;
– read-only memory (Debug ROM).

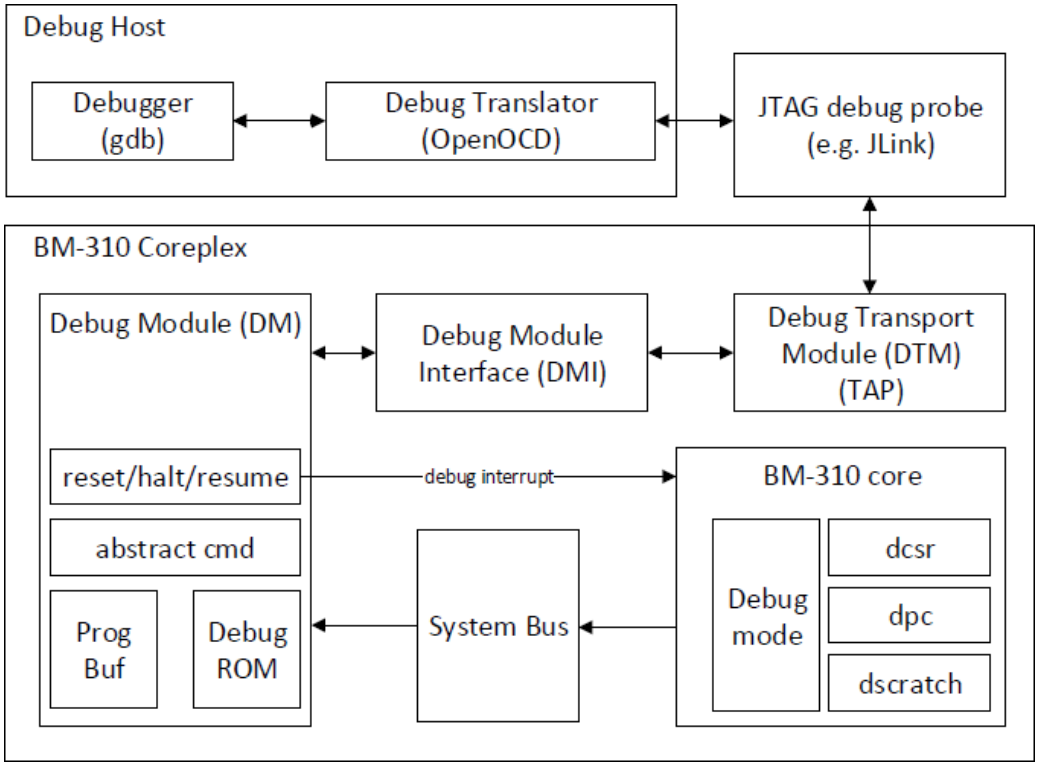


Figure 105 – Debug system architecture

The debug subsystem consists of the DTM (Debug Transport Module) block and the DM (Debug Module) block. The DTM is connected to the DM through the DMI (Debug Module Interface) block.

28.1. DTM (Debug Transport Module) block

The DTM contains a TAP (Test Access Port) that complies with the JTAG interface specification. The IR length is 5 bits. Access to the DM over JTAG is performed via the DMI register. The list of registers accessible over JTAG is given in the table below. The general structure of the core debug chain is shown in Figure 106. The clock-domain crossing from the JTAG clock domain to the core clock domain occurs in the DMI block.

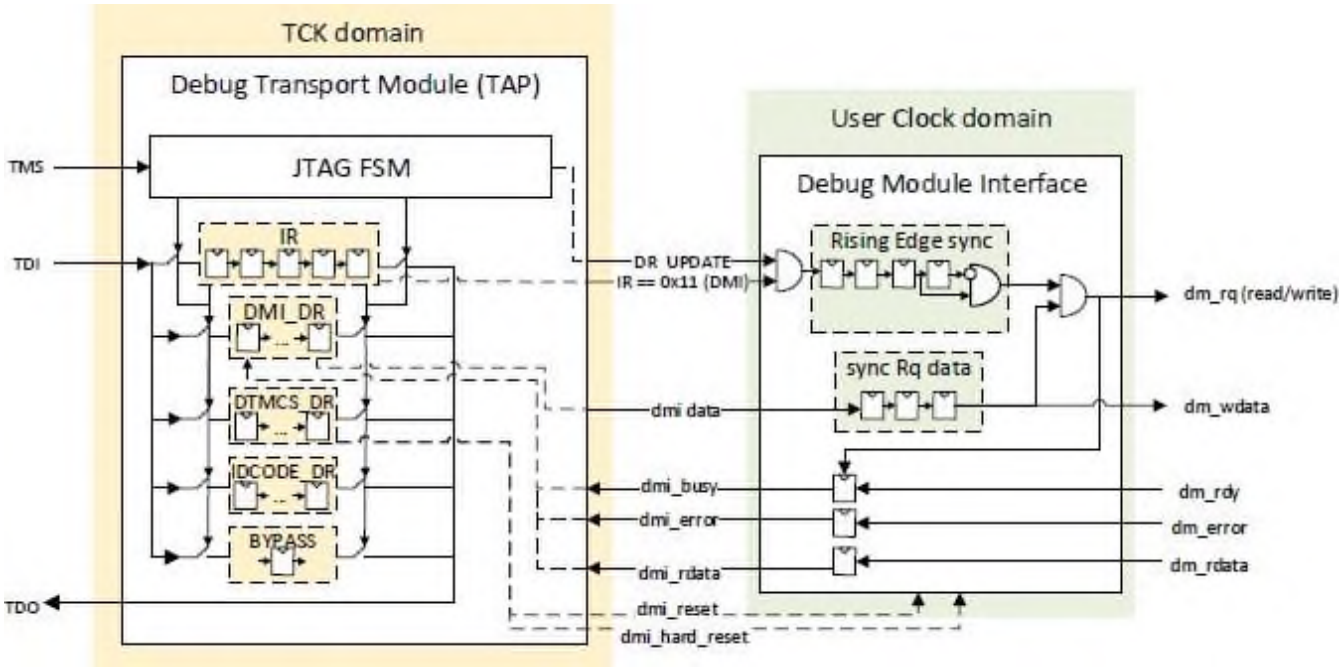


Figure 106 – DM interface

28.1.1. **DTM registers**

Table 537 – DTM registers

Address	Name	Access
0x00	BYPASS	R
0x01	Chip manufacturer ID register (IDCODE)	R
0x10	DTM status and control register (DTMCS)	R/W
0x11	DMI access register	R/W
0x12-0x1A	BYPASS	R/W

Table 538 – DTM status and control register (DTMCS)

Bits	Name	Access	Description	Value
17	dmihardreset	W1	Writing 1 resets the DMI.	
16	dmireset	W1	Writing 1 clears the DMI error.	
14…12	idle	R	Number of cycles that the debugger must hold the IDLE state between issuing a request and reading the result from the DMI in order to avoid the busy state.	5
11, 10	dmistat	R	MI state: 0 – no errors; 1 – see 2; 2 – operation execution error; 3 – operation not yet completed (busy state)	
9…4	abits	R	Width of the DMI register address field.	7
3…0	version	R	Implemented specification version: 0 – 0.11; 1 – 0.13	1

Table 539 – DMI access register

Bits	Name	Access	Description	Value
abits+33…34	address	R/W	DMI request address.	
33…2	data	R/W	Write data sent to the DM during the Update-DR state, or data read from the DM.	
1…0	op	R/W	Operation code. On a write: 0 – null request. Nothing is sent to the DM; 1 – read at address; 2 – write at address. On a read: 0 – the previous request completed successfully; 2 – the previous request returned an error; 3 – the previous request has not yet completed. A DMI reset is required to clear this state.	

Interaction between the debugger and the DM occurs via the DMI by reading and writing DM registers. For this purpose, the DMI register in the DTM is used. The abits field value is 7. The list of registers accessible via the DMI is given in Table 540.

Table 540 – DM debug registers

Address	Name	Access
0x04-0x07	Abstract Data 0-3	R/W
0x10	DM control	R/W
0x11	DM status	R/W
0x12	Hart info	R/W
0x16	Abstract Control and Status	R/W
0x17	Abstract Command	R/W
0x18	Abstract Command Autoexec	R/W
0x20-0x2F	Program Buffer 0-15	R/W

28.2. **DM (Debug Module) block**

Interaction between the processor core and the DM is performed by reading/writing the memory-mapped registers (DM MMRs). The DM memory map is shown in Table 541.

Table 541 – DM memory map

Address	Name	Access
0x0000_0000	Safe zero	R/W
0x0000_0100	Halted ack	W
0x0000_0104	Going ack	W
0x0000_0108	Resume ack	W
0x0000_010C	Exception	W
0x0000_0300	Where to	R
0x0000_0338	Abstract command 0	R
0x0000_033C	Abstract command 1	R
0x0000_0340-0x37C	Program buf	R/W
0x0000_0380-0x38C	Abstract data	R/W
0x0000_0400	CPU flags	R
0x0000_0800-0xFFF	Debug ROM	R

Note: Accesses to unused addresses are prohibited and may lead to unpredictable behavior.

Table 542 – CPU flags

Bits	Name	Access	Description	Value
0	Going	R	Request to execute an abstract command. When in debug mode, the core reads this field and branches to the address of the first abstract command (0x338).	
1	Resumereq	R	Request to exit debug mode. When in debug mode, the core reads this field and exits debug mode if its value is 1. Program execution then continues from the PC value saved in the dpc register.	

28.2.1. **DM registers**

To interact with an external debugger, a special execution mode is implemented in the processor core: debug mode. Table 556 provides the list of debug mode control registers. The processor is allowed to read and write the debug control registers only in debug mode. In any other mode, an attempt to access these registers causes an exception. Execution in debug mode has the following properties:

- accesses to control registers and memory are performed in the same way as in Machine privilege mode;
- any interrupts are not serviced;
- exceptions do not update the values of the control registers;
- all instructions that change the privilege level are ignored. The only exception is the *ebreak* instruction. When the *ebreak* instruction is executed in debug mode, the processor branches to the start of the Debug ROM without updating dpc and dcsr.

Table 543 – Debug mode control registers

Address	Name	Access
0x7B0	Debug mode control register dcsr	R/W
0x7B1	Program counter register dpc	R/W
0x7B2	Additional debug mode register dscratch	R/W

Table 544 – Debug mode control register dcsr

Bits	Name	Access	Description	Value
31…28	version	R/W	Version: 0 – debug mode is not supported 4 – debug mode is supported and conforms to specification version 0.13	4
15	ebreakm	R/W	A value of 1 means that when the ebreak instruction is executed in M mode, the processor enters debug mode.	0
12	ebreaku	R/W	A value of 1 means that when the ebreak instruction is executed in U mode, the processor enters debug mode.	0
8…6	cause	R	Reason for entering debug mode: 1 – ebreak instruction executed 3 – interrupt from the DM 4 – next instruction executed in single-step mode	0
2	step	R/W	A value of 1 means that single-step mode is enabled. In this mode, after exiting debug mode the processor executes the next instruction of the user application and then switches back to debug mode. Interrupts are disabled during single-step execution.	0
1, 0	prv	R/W	The privilege mode in which the core was running before switching to debug mode. The debugger can change the value of this field to change the processor mode after exiting debug mode.	0

Table 545 – Program counter register dpc

Bits	Name	Access	Description	Value
31…0	dpc	R/W	Saved program counter value. When entering debug mode, the current program counter value is saved in this register. Writing to this register is possible only in debug mode. When exiting debug mode, execution continues from the address stored in this register.	

Table 546 – Additional debug mode register dscratch

Bits	Name	Access	Description	Value
31…0	dscratch	R/W	Additional register used by the DM to store intermediate information.	

After entering debug mode, the processor core branches to the start address of the Debug ROM (address 0x800). The software code stored in the Debug ROM polls the DM flags. When a request to execute an abstract command is detected ("Going" flag in register 0x500), execution branches to the address of the first abstract command (register at address 0x338). If an exception occurs while executing an abstract command or code from the program buffer, the core branches to an address inside the Debug ROM where the exception handler code is located. This handler contains an instruction that writes to address 0x10C. After that, the debugger can read the Abstract Control and Status register and determine that an exception occurred during execution of the abstract command. If, during polling, a request to resume execution of the user application is read ("Resumereq" flag), code is executed that branches to the address stored in the dpc register.

Any debugger operations: reading/writing registers, reading/writing memory, setting breakpoints in the code, single-step execution – are performed by modifying the code in the program buffer and executing abstract commands.

28.3. **Connecting a JTAG adapter to the chip**

If the debug interface is not locked, a JTAG adapter can be connected to the chip, and development tools can use it to work with the integrated circuit in debug mode. The JTAG lines must be pulled up to the supply voltage through resistors with resistance from 4.7 to 10 kOhm, taking into account that these pull-ups must not interfere with system operation (see Figure 107).

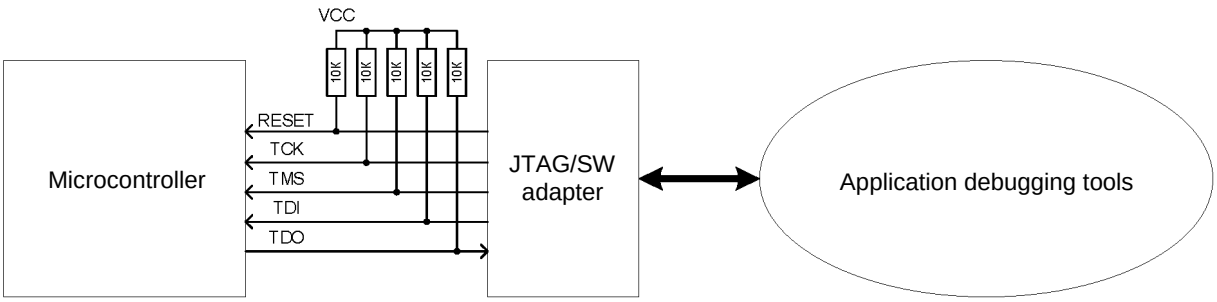


Figure 107 – Debug mode operation diagram

The JTAG debug interface is located on port A pins (see Table 547). In debug mode, the information given in Section 18 “Input/output ports” must be taken into account.

Table 547 – Reassignment of JTAG interface pins

JTAG pin	Chip pin	Description
TCK	PA6/TMR1_CH4/TCK	Port A pins that are shared with Timer 1 pins are used as JTAG interface pins. It is prohibited to use Timer 1 pins that are shared with JTAG when the JTAG port is enabled.
TDO	PA7/TMR1_CH4N/TDO	
TMS	PA8/TMR1_ETR/TMS	
TDI	PA9/TMR1_BRK/TDI	
JTAG_EN	JTAG_EN	It must be driven to logic “0” for operation with the JTAG debug interface.

29 Watchdog timers

29.1. Watchdog timer block registers

Table 548 – Description of the IWDG watchdog timer block registers

Base Address	Name	Description
0x5005_0000	IWDG	IWDG Watchdog Timer
Offset		
0x00	IWDG_KR[15:0]	Key register
0x04	IWDG_PR[2:0]	Watchdog timer clock divider register
0x08	IWDG_RLR[11:0]	Watchdog timer counting base register
0x0C	IWDG_SR[1:0]	Watchdog timer status register

Table 549 – Description of the WWDG watchdog timer block registers

Base Address	Name	Description
0x5004_8000	WWDG	WWDG watchdog timer
Offset		
0x00	WWDG_CR[7:0]	Control register
0x04	WWDG_CFR[9:0]	Configuration register
0x08	WWDG_SR[0]	Status register

29.1.1. IWDG watchdog timer block

IWDG is an independent 12-bit watchdog timer that counts down from the counting base (reload value, the programmed initial value) to zero. When the counter reaches zero, the watchdog timer generates a reset signal that resets the system. To prevent a reset from the IWDG, the timer must be reloaded in advance by writing the key value 0xAAAA to the IWDG_KR register, after which the timer again starts counting down from the programmed value. This provides control over software lockup at certain points where it may occur (if the IWDG is not reloaded in time, the system is reset). A system reset generated by the independent watchdog timer resets the IWDG. Any other system reset (for example, an external reset) does not affect the configuration and operation of the IWDG timer.

The block diagram of the independent watchdog timer is shown in Figure 108.

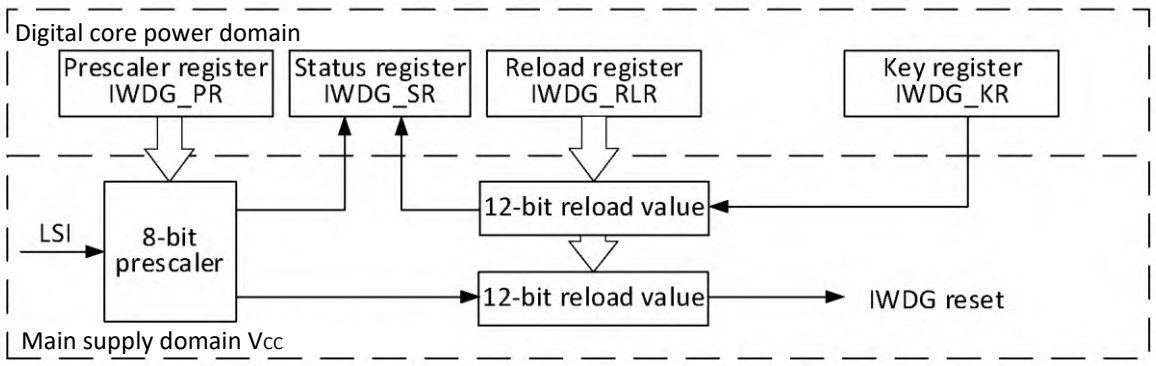


Figure 108 – Block diagram of the independent watchdog timer

Structurally, the IWDG consists of two parts. The part with the control registers is located in the digital core power domain, is accessible over APB, is clocked by PCLK, and is reset by all types of reset. The part with the independent watchdog timer itself is located in the main chip supply domain V_{CC} and is clocked from the LSI oscillator, therefore the LSI oscillator must be running for actual writes to the IWDG registers in the main supply domain V_{CC}. The watchdog timer part in the main supply domain V_{CC} is reset only by the reset generated by the IWDG; after an IWDG reset, the watchdog timer stops its operation.

The watchdog timer period from enabling operation to generating a reset is a function of the counting base (reload value) and the LSI frequency divided by the prescaler value.

The IWDG prescaler value in the main supply domain is updated by writing the IWDG_PR register accordingly. The counting base (the value from which the timer starts counting down when started) in the main supply domain is updated by writing the IWDG_RLR register and writing the key value 0xAAAA to the key register. In order to allow writes to the IWDG_PR and IWDG_RLR registers, the key value 0x5555 must first be written to the key register to enable write access to them. When a prescaler value is written to the IWDG_PR register and the key 0xAAAA is written to the IWDG_KR register, update requests for the corresponding parameters (prescaler and counting base) are generated and passed to the IWDG in the main supply domain. Update requests are not latched (not saved) by the part in the main supply domain, that is, for any type of reset the update requests from the IWDG in the core power domain are cleared. The presence of an update request from the IWDG in the core power domain and/or information about the actual update process on the IWDG side in the main supply domain are indicated by the PVU (for the prescaler) and RVU (for the counting base) flags. Actual updating of the values (with clearing of the corresponding flags after the update) is performed only when IWDG operation is enabled and the LSI clock is present. During the actual update of the prescaler and counting base, writes to the IWDG_PR and IWDG_RLR registers are not allowed, respectively. After the actual update, the update requests from the IWDG in the core power domain V_{CC} are cleared.

Example of working with the IWDG block:

1 Enabling the LSI oscillator and waiting for it to become operational.

2 Enabling access to the IWDG_PR and IWDG_RLR registers by writing the value 0x5555 to the IWDG_KR register.

3 Writing the prescaler value to the IWDG_PR register.

4 Enabling IWDG operation by writing the value 0xCCCC to the IWDG_KR register.

5 Waiting for the prescaler value update (PVU flag cleared to 0).

6 Enabling access to the IWDG_PR and IWDG_RLR registers by writing the value 0x5555 to the IWDG_KR register.

7 Checking that the RVU flag is 0 before writing to IWDG_RLR.

8 Writing the reload value to the IWDG_RLR register.

9 Periodically reloading the watchdog timer during program execution with the value in the IWDG_RLR register by writing the key value 0xAAAA to the IWDG_KR register.

29.1.1.1. **IWDG_KR**

Table 550 – IWDG_KR Register

Number	31…16	15…0
Access	U	W
Reset	0	0
	-	KEY[15:0]

Table 551 – Description of IWDG_KR register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…16	–	Reserved
15...0	KEY[15:0]	Key value (write-only, reads as 0x0000). On a write: – 0xAAAA – reloads the timer with the value from the IWDG_RLR register. This value must be written periodically by software while IWDG operation is enabled; otherwise, the watchdog timer generates a reset if the timer reaches zero. – 0x5555 – enables write access to the IWDG_PR and IWDG_RLR registers. If, after write access is enabled, a different key value is written to the key register, write access to the IWDG_PR and IWDG_RLR registers is disabled. – 0xCCCC – enables watchdog timer operation (if watchdog timer operation is not enabled; timer operation is stopped in hardware by an IWDG reset).

29.1.1.2. **IWDG_PR**

Table 552 – IWDG_PR Register

Number	31…3	2…0
Access	U	R/W
Reset	0	0
	-	PR[2:0]

Table 553 – Description of the IWDG_PR register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...3	–	Reserved

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
2...0	PR[2:0]	Watchdog timer clock divider. 00 – division by 4: LSI/4 01 – division by 8: LSI/8 10 – division by 16: LSI/16 11 – division by 32: LSI/32 100 – division by 64: LSI/64 101 – division by 128: LSI/128 110 – division by 256: LSI/256 111 – division by 256: LSI/256 To allow writes to the register, write access must first be enabled by writing the key value 0x5555 into the IWDG_KR register. When the register is written, an update request is generated for the prescaler value in the IWDG part in the main supply domain (see Figure 108). Writes to the IWDG_PR register are not allowed during the actual update of the prescaler in the IWDG part in the main supply domain V_{CC} (see Figure 108). On read, the value written into the register is returned, and not the IWDG prescaler value in the main supply domain.

29.1.1.3. **IWDG_RLR**

Table 554 – IWDG_RLR Register

Number	31…12	11…0
Access	U	R/W
Reset	0	0xFFF
	-	RLR[11:0]

Table 555 – Description of the IWDG_RLR register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...12	–	Reserved
11...0	RLR[11:0]	Watchdog timer reload value (counting base). The watchdog timer is decremented starting from this value. Defines the value that is loaded into the watchdog timer when the value 0xAAAA is written to the IWDG_KR register. To allow writes to the register, write access must first be enabled by writing the key value 0x5555 into the IWDG_KR register. Writes to the IWDG_RLR register are not allowed during the actual update of the counting base in the IWDG part in the main supply domain V_{CC} (see Figure 108). On read, the value written into the register is returned, and not the IWDG reload value in the main supply domain.

29.1.1.4. **IWDG_SR**

Table 556 – IWDG_SR Register

Number	31…2	1	0
Access	U	R	R
Reset	0	0	0
	-	RVU	PVU

Table 557 – Description of the IWDG_SR register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31…2	–	Reserved
1	RVU	Watchdog timer value update flag. Set by hardware and indicates that the IWDG watchdog timer value in the main supply domain VCC is being updated from the IWDG_RLR reload register (see Figure 108). **IWDG operation enabled and LSI running:** Reflects a request from the control registers to update the watchdog timer value with the IWDG_RLR register value and/or the actual update of the watchdog timer value in the main supply domain VCC. Is cleared upon completion of the actual reload value update. The IWDG_RLR reload register value can be updated only if this bit is zero. **IWDG operation not enabled or LSI not running:** Reflects a request from the control registers to update the watchdog timer value in the main supply domain VCC with the IWDG_RLR register value. Is cleared by any type of reset. The IWDG_RLR reload register value can be updated regardless of the state of this bit, after the IWDG is started, it will be updated with the last value written to IWDG_RLR.
0	PVU	Watchdog timer clock divider update flag. Set by hardware and indicates that the IWDG clock divider value in the main supply domain VCC is being updated from the IWDG_PR prescaler register (see Figure 108). **IWDG operation enabled and LSI running:** Reflects a request from the control registers to update the watchdog timer clock divider with the IWDG_PR register value and/or the actual update of the prescaler value in the main supply domain VCC. Is cleared upon completion of the actual divider update. The IWDG_PR prescaler register value can be updated only if this bit is zero. **IWDG operation not enabled or LSI not running:** Reflects a request from the control registers to update the watchdog timer value in the main supply domain VCC with the IWDG_PR register value. Is cleared by any type of reset. The IWDG_PR reload register value can be updated regardless of the state of this bit, after the IWDG is started, it will be updated with the last value written to IWDG_PR.

29.1.2. **WWDG watchdog timer registers**

In this controller, the WWDG is a window watchdog timer whose counter counts down from the value that is written to the WWDG_CR register (bits T[6:0]) not down to zero, but down to 63 (0x3F). In addition to the value from which the WWDG counts and the value 0x63 down to which this watchdog timer counts, there is one more intermediate value located between these two values. It is also a programmable value and thus a window is formed between this value and 0x3F, which is why this watchdog timer is a window watchdog timer. Like the IWDG, the WWDG watchdog timer, after counting down to 0x3F, issues a command to reset the system unless this window watchdog timer is reloaded in time. However, it must reloaded not only no later than when its counter has counted down to 0x3F, but also not earlier than when it has counted down to the intermediate value, which is also written into the WWDG_CFR register (bits W[6:0]). This value therefore must lie in the interval from 0x3F to the programmed counter value from which it starts counting down. In this way, certain time intervals in the execution of our program can be checked.

The block diagram of the window watchdog timer is shown in Figure 109.

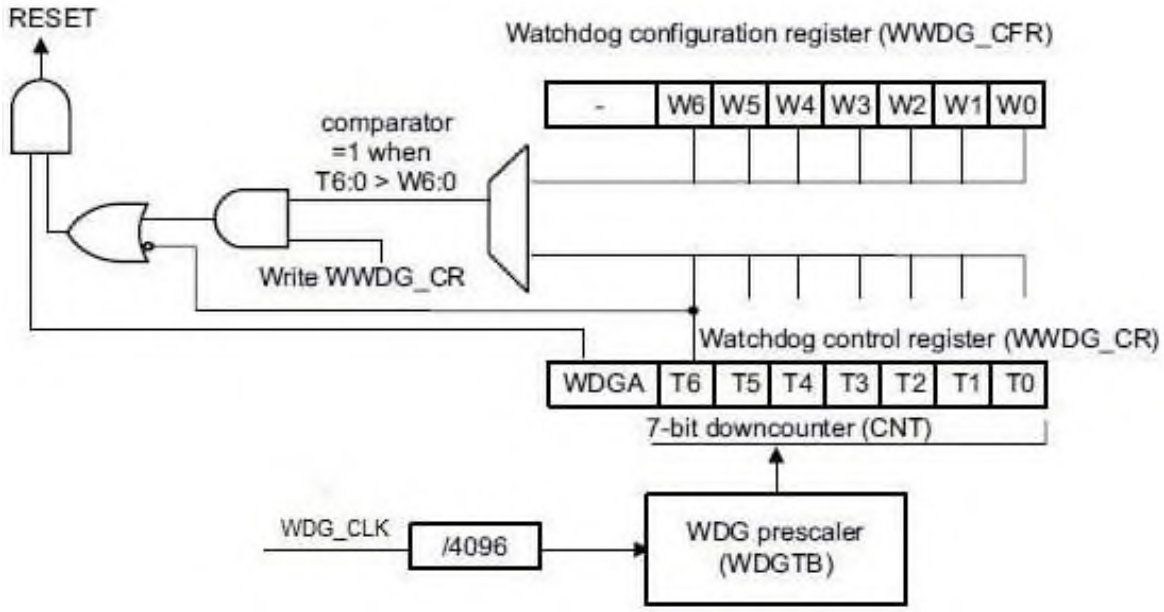


Figure 109 – Block diagram of the window watchdog timer

The counter clock signals pass through the first prescaler with a fixed division factor of 4096 from the WDG_CLK bus to a second, programmable prescaler whose division factor can be set using the WGTB register bits. After the second prescaler, the clock signals go to the down-counter.

The value stored in bits T6:T0 of the WWDG_CR register, after the window timer is started, begins counting down, and as soon as it reaches the value 0x3F, that is, as soon as bit T6 is cleared, a system reset signal is applied to one of the multiplexers. But as a multiplexer, it can receive a reset signal not only when bit T6 is cleared. It can also receive a similar signal from the comparator where the value of bits T6:T0 of the WWDG_CR register is compared with the value of bits W6:W0 of the WWDG_CFR register. If the first value is greater than the second, that is, if the down-counter has not yet counted down to the first boundary value and during this time there was an attempt to reload the window watchdog timer, then the

multiplexer receives this signal, which also causes the system to reset.

The operation of the window watchdog timer is also shown in the timing diagram in Figure 110.

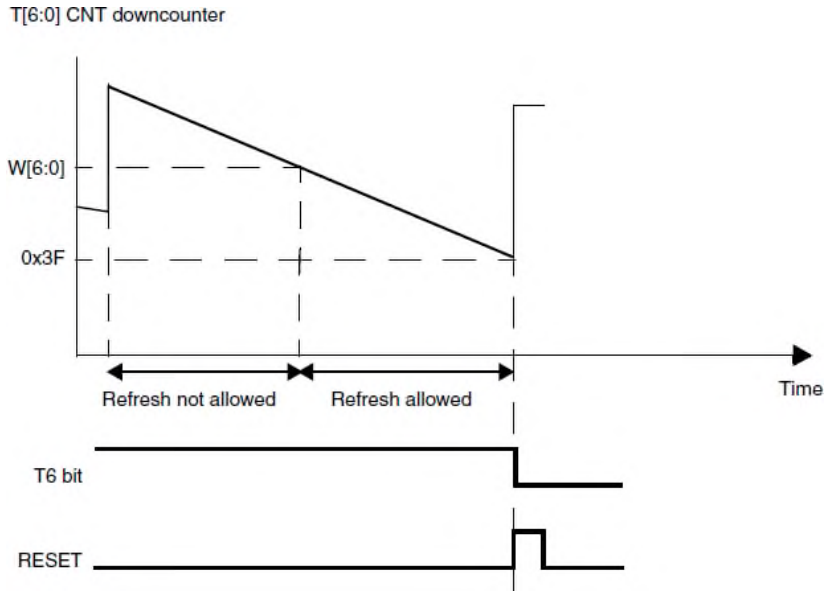


Figure 110 – Timing diagram of the window watchdog timer

The WWDG operation time from start to end of counting and reset can be calculated using the following formula

$$t_{WWDG} = t_{WDG\_CLK} \cdot 4096 \cdot 2^{WGTB} \cdot (T[5:0] + 1)\text{, мс} \quad (33)$$

where t_{WWDG} – timeout;

t_{WDG_CLK} – APB bus clock period in ms.

29.1.2.1. **WWDG_CR**

Table 558 – WWDG_CR Register

Number	31…8	7	6…0
Access	U	R/S	R/W
Reset		0	1
	-	WDGA	T6…T0

Table 559 – Description of the WWDG_CR register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...18		Reserved
7	WDGA	Activation bit. This bit is set by software and cleared only by hardware on reset. When WDGA = 1, the watchdog timer can generate a reset. 0 – watchdog timer disabled 1 – watchdog timer enabled
6…0	T[6:0]	Value of the 7-bit counter (from the most significant bit to the least significant bit). These bits store the watchdog timer value, which is decremented every 4096×2^{WDGTB} WDG_CLK cycles of the APB peripheral bus.

29.1.2.2. **WWDG_CFR**

Table 560 – WWDG_CFR Register

Number	31…10	9	8	7	6…0
Access	U	R/S	R/W	R/W	R/W
Reset		0	0	0	1
	-	EWI	WDGTB1	WDGTB0	W6…W0

Table 561 – Description of the WWDG_CFR register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...10		Reserved
9	EWI	Early wakeup interrupt. If this bit is set, generation of an interrupt is enabled when the watchdog timer reaches the value 40 h. The interrupt is disabled only by a hardware reset.
8, 7	WGTB[1:0]	Watchdog timer clock divider: 00 – timer frequency (WDG_CLK/4096)/1; 01 – timer frequency (WDG_CLK/4096)/2; 10 – timer frequency (WDG_CLK/4096)/4; 11 – timer frequency (WDG_CLK/4096)/8.
6..0	W[6:0]	Window value. These bits store the window value within which it is allowed to initialize bits T[6:0] with a value in the range 40h-7Fh. If the bits are initialized when T>W, a reset is generated on the RESET output. If the timer reaches the value T =3Fh, a reset is also generated.

29.1.2.3. **WWDG_SR**

Table 562 – WWDG_SR Register

Number	31…2	1	0
Access	U	R	R/C
Reset		1	0
	-	WDG_WEC	EWIF

Table 563 – Description of the WWDG_SR register bits

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values
31...2		Reserved
1	WDG_WEC	Write-completion flag for the WWDG_CR and WWDG_CFR registers: 1 – write completed; 0 – write not completed.
0	EWIF	Early wakeup interrupt flag. This bit is set by hardware when the watchdog timer reaches the value 40 h. The bit is cleared by software by writing zero. Writing 1 has no effect. This bit is also set if the interrupt is disabled (EWI = 0).

30 Random number generator RANDOM

30.1. Purpose and properties

The module is used to control two ring oscillators, to enable them, to test them, and to generate a random 32-bit number.

The module has the following features:

– contains two generators;
– allows the generators to be enabled and disabled in groups of 1;
– allows a pause to be set after the generators are enabled, before random-number collection starts;
– allows the generators to be enabled by a bit in the control register;
– has several modes for starting random-number collection:
– single;
– continuous;
– generates an interrupt signal upon completion of random-number collection.

The random number generator is a mechanism for obtaining an unpredictable sequence of numbers based on a physical non-deterministic process.

To increase the amount of random numbers and improve their statistical characteristics, the user can use a hardware-software pseudo-random number generation mechanism. In this case, the random number generator output is used as a randomizing factor for performing an encryption operation using a strong cipher.

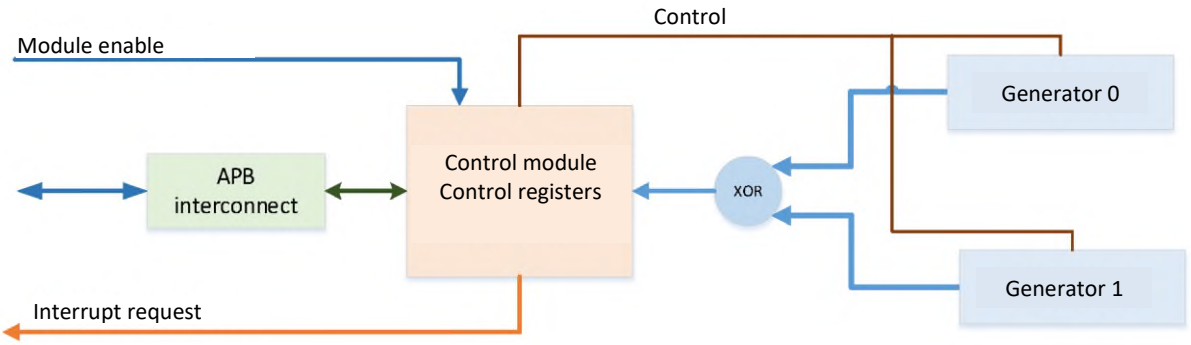


Figure 111 – Block diagram of the random number generator block

30.2. Module registers

Table 564 – Module registers

Base Address	Name	Description
0x500B_8000	RANDOM	Random number generator block
Offset		
0x00	STAT_CTRL	Status and control register
0x04	INT_CTRL	Interrupt control register
0x08	CLK_DIV	Generator clock divider register
0x0C	PAUSE	Enable-pause register
0x10	OUTPUT	Random value register
0x14	PAUSE_CNT	Pause counter register
0x18	TEMP	Random-collection register

30.2.1. **Status and control register STAT_CTRL**

Relative address: 0x00, initial value 0x000000F0.

Table 565 – STAT_CTRL register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values	Access
0	–	Reserved. Always write 0.	RW
1	EN_STATE	Generators enabled, this bit is set to 1 if there is at least one group of enabled generators (MASK value is nonzero) and the pause after enabling has expired (see *PAUSE*).	RO
2	BUSY	Generator busy, random-number collection is in progress.	RO
3	UNREAD_DATA	Unread data flag. After a random number is collected, this bit is set to 1; the bit is cleared when the random number is read (see *OUTPUT*).	RO
5, 4	MASK	Mask for selecting the active generator: bit 4 – generator 0; bit 5 – generator 1.	RW
7, 6	–	Reserved	RO
9, 8	WORK_MODE	Random-number collection mode: 00 – random-number collection stopped; 01 – single start of random-number collection (after collection starts, the field is automatically cleared to 00); 10 – collection of a new random number starts if there is no unread random number (UNREAD_DATA equals 0); 11 – continuous random-number collection; collection of a new random number starts immediately after the previous number is collected.	RW
11, 10	–	Reserved	RO
17…12	BIT_CN	Counter of remaining bits until the new random number is collected.	RO
31…18	–	Reserved	RO

30.2.2. **Interrupt control register INT_CTRL**

Relative address: 0x04.

Table 566 – INT_CTRL Register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values	Access
0	INT_EN	Interrupt enable. If this bit is 1 and the UNREAD_DATA field is 1, the interrupt output goes 1.	RW
1	UNREAD_DATA	Unread data flag. After a random number is collected, this bit is set to 1; the bit is cleared when the random number is read (see OUTPUT).	RO
31…2	-	Reserved	RO

30.2.3. **Generator clock divider register CLK_DIV**

Relative address: 0x08, initial value 7.

Table 567 – CLK_DIV Register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values	Access
15…0	DIV	To generate the generator data latch frequency, the module input clock is divided by 2·• (**DIV** + 1).	RW
31…16	-	Reserved	RO

30.2.4. **Enable-pause register PAUSE**

Relative address: 0x0C.

Table 568 – PAUSE Register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values	Access
31…0	PAUSE	After the generators are enabled through an external signal or a control-register bit, when any block is enabled via the mask, or when newly enabled blocks are added to blocks already running, random number collection may be started only after this pause has elapsed. If enabling occurs while a random number is being collected, collection is suspended until this pause completes.	RW

30.2.5. **Random value register OUTPUT**

Relative address: 0x10.

Table 569 – OUTPUT Register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values	Access
31…0	DATA	The last collected random number. While the next random number is being collected, the value in this register does not change. The data is updated when random number collection completes.	RO

30.2.6. **Pause counter register PAUSE_CNT**

Relative address: 0x14.

Table 570 – PAUSE_CNT Register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values	Access
31…0	CUR_PAUSE	Current value of the generator enable pause counter (value for reference). A zero means that the pause has elapsed.	RO

30.2.7. **Random-collection register TEMP**

Relative address: 0x18.

Table 571 – TEMP Register

Bit number	Bit functional name	Explanation of the bit functional name, brief description of purpose and possible values	Access
31…0	TEMP_DATA	Current value of the random number collection shift register (value for reference).	RO

31 USART controller (ISO7816)

31.1. Introduction

The universal synchronous/asynchronous receiver/transmitter (USART) provides flexible means for full-duplex communication with external equipment that requires the standard NRZ data encoding format. The USART module supports a wide range of data rates using a fractional baud rate generator.

The module supports synchronous one-direction data transmission and half-duplex single-wire data transmission. The Smartcard protocol and modem control (CTS/RTS) are also supported.

31.2. USART module features

The USART module has the following features:

- full-duplex asynchronous data communication;
- standard NRZ (Mark/Space) data format;
- configurable oversampling by 16 or 8 to provide flexibility between speed and clock tolerance;
- fractional baud rate generator:
- programmable transmit and receive baud rate (the baud rate value at the maximum APB bus clock is given in the specifications);
- programmable data word length (8 or 9 bits);
- support for 2 stop bits;
- transmitter clock output for synchronous data transmission;
- Smartcard emulation capability (Smartcard mode):
- the Smartcard interface supports the asynchronous protocol for smart cards in accordance with ISO 7816-3;
- 0.5 and 1.5 stop bits for the Smartcard interface, without the possibility of using back2back frames in 0.5 stop-bit mode. For more details, see subclause 31.3.1
- “USART character description”;
- single-wire half-duplex data transmission;
- separate enable bits for transmitter and receiver;
- transmission-detection flags:
- receiver buffer full;
- transmit buffer empty;
- transmission complete flag;
- parity checking:
- parity bit transmission;
- parity checking of the received data byte;

- error detection flags:
- overrun;
- noise detected;
- framing error;
- parity error;
- interrupts with flags:
- CTS state change;
- transmit data register empty;
- transmission complete;
- receive data register full;
- overrun error;
- framing error;
- noise detected;
- parity error.

31.3. **USART functional description**

The interface is connected to external devices via three pins (see Figure 112). Any bidirectional USART data transmission requires at least two pins: a receive data input (RX) and a transmit data output (TX):

- **RX**: serial data input. Oversampling is used to recover the data and separate the required incoming data from noise;
- **TX**: transmit data output. When the transmitter is disabled, the pin returns to the state defined by the I/O port configuration.

When the transmitter is enabled but no data is being sent, the TX pin is held high (logic 1). In single-wire mode and Smartcard mode, this I/O port is used for both data transmission and reception (SW_RX).

Through these pins, serial data is transmitted and received in normal USART operation as frames containing:

- an idle line before transmission or reception;
- a start bit;
- a data word (8 or 9 bits), with the LSB transmitted first;
- 0.5, 1.5, or 2 stop bits indicating that the frame is complete;
- this interface uses a fractional baud rate generator with a 12-bit mantissa and a 4-bit fractional part;
- status register (USART_SR);
- data register (USART_DR);
- baud rate register (USART_BRR) – 12-bit mantissa and 4-bit fractional part;
- guard time register (USART_GTPR) when Smartcard mode is used.

A description of the registers and their bits is given in subclause 31.5 “USART registers”. For operation in synchronous mode, one additional pin is required:

- **CK:** Transmitter clock output. This pin outputs the data clock for synchronous transmission corresponding to SPI master mode (no clock pulses in the start and stop bits, and a software option to output a clock pulse on the last data bit). In parallel, data can be received synchronously on the RX pin. This can be used to control external peripheral devices that have shift registers (for example, LCD drivers). Clock phase and polarity are selected by software. In Smartcard mode, the CK pin provides the clock for the smart card.

In hardware flow control mode, additional pins are required:

- **CTS**: the Clear To Send signal blocks data transmission after the current frame when the signal is high;
- **RTS**: the Request To Send signal indicates that USART is ready to receive data (when the signal is low).

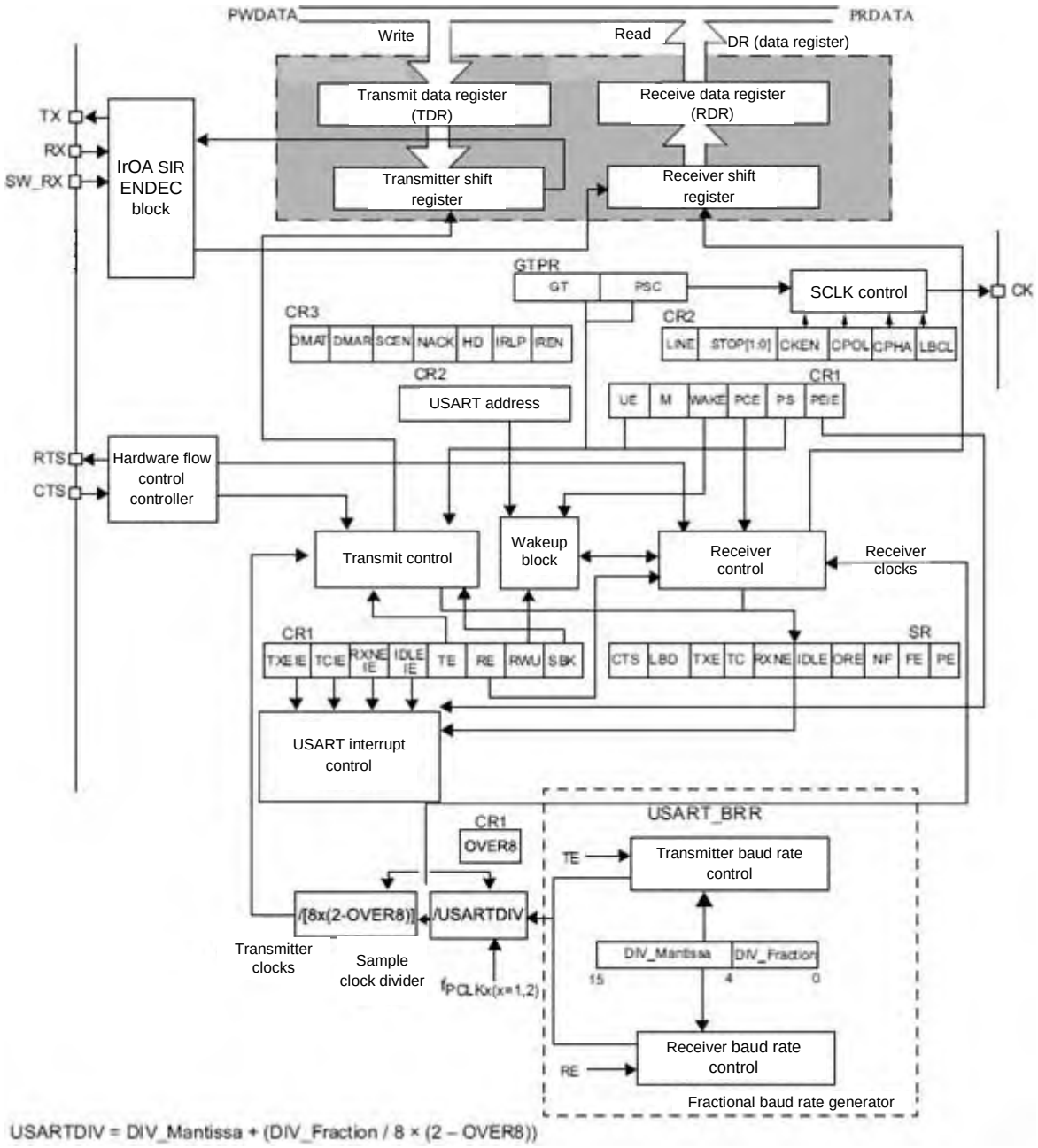


Figure 112 – USART block diagram

31.3.1. **Description of USART characters**

For transmission, a word length of 8 or 9 bits can be selected by programming bit M in the USART_CR1 register (see Figure 113).

The TX output is at logic 0 while the start bit is transmitted, and at logic 1 while the stop bit is transmitted.

Transmission and reception are controlled by the baud rate generator. Clock pulses for the transmitter and receiver are generated when the corresponding enable bit is set.

Each block is described in detail below.

Note: When using the 0.5 stop-bit mode, do not use back2back frames (frames with a pause shorter than one bit period), because the USART module does not have time to receive the frame due to a one-bit wait between frames (T = 1 mode, ISO7816-3).

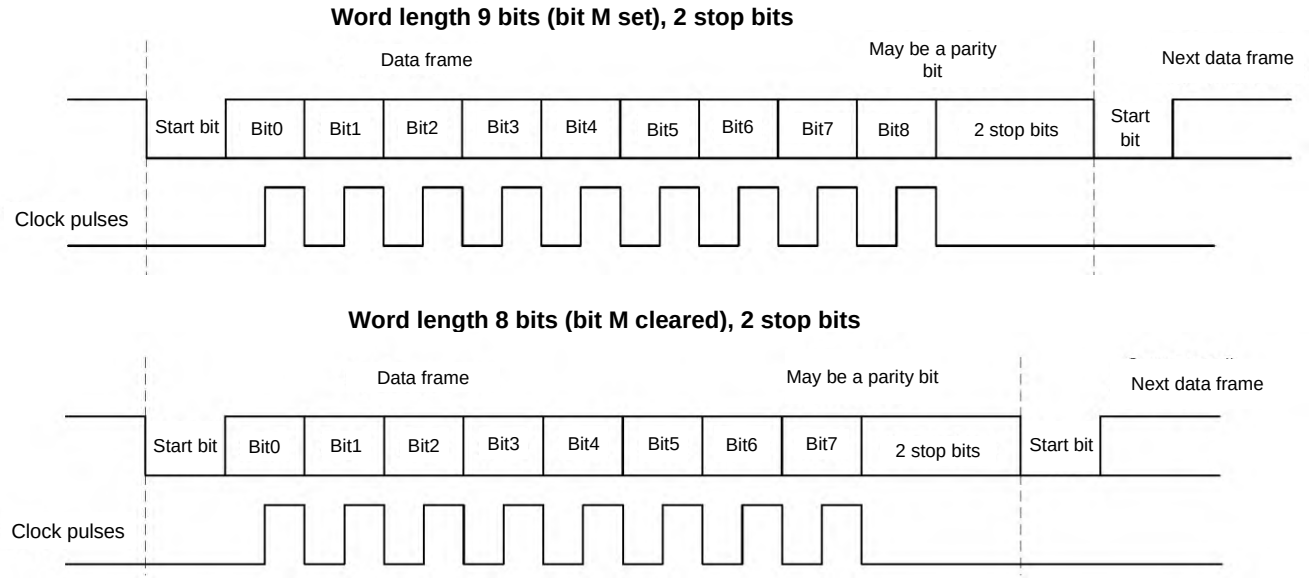


Figure 113 – Programming the word length

31.3.2. **Transmitter**

The transmitter can send 8-bit or 9-bit data words, depending on the state of bit M.

When the transmit enable bit (TE) is set, data from the shift register is driven on the TX output, and the corresponding clock pulses are driven on the CK output.

31.3.2.1. **Character transmission**

During USART transmission, data is shifted out on TX, with the least significant bit (LSB) sent first. In this mode, the USART_DR register consists of a buffer (TDR) between the internal bus and the transmit shift register (see Figure 112).

Each character is preceded by a low-level start bit (logic 0) with the duration of one bit period. The character ends with a programmable number of stop bits.

The USART module supports the following stop bits: 0.5, 1.5, and 2 stop bits.

Note: The TE bit must not be cleared while data is being transmitted. Clearing TE during transmission corrupts the data on TX, because the baud rate generator counters are stopped. The data currently being transmitted is lost.

After TE is set again, an idle frame is transmitted.

31.3.2.2. **Configurable stop bits**

The number of stop bits transmitted with each character is programmed in bits 13 and 12 of the USART_CR2 register.

- *2 stop bits*: supported in normal USART mode, single-wire mode, and modem mode;
- *0.5 stop bit*: used when receiving data in Smartcard mode;
- *1.5 stop bits*: used when transmitting and receiving data in Smartcard mode.

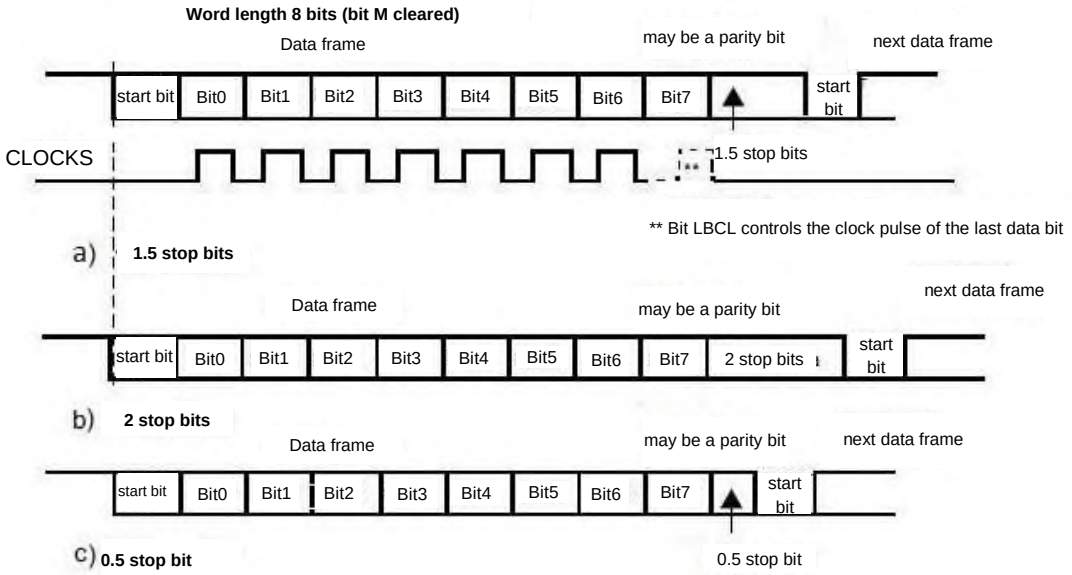


Figure 114 – Configurable stop bits

Programming procedure:

1. Enable the USART by setting bit UE in the USART_CR1 register to 1;
2. Program bit M in the USART_CR1 register to define the data word length;
3. Program the number of stop bits in the USART_CR2 register;
4. Select the required baud rate using the USART_BRR register;
5. Set bit TE in the USART_CR1 register;
6. Write the data to be transmitted to the USART_DR register (this clears TXE). Repeat this operation for each character transmitted when using a single buffer;
7. After writing the last data to USART_DR, wait until TC becomes 1. This indicates that transmission of the last frame is complete. For example, this is required when the USART module is disabled or when the device enters Halt mode, to avoid corrupting the last transmission.

31.3.2.3. **Single-byte transfer**

TXE is always cleared when writing to the data register.

TXE is set by hardware and indicates that:

– data has been moved from TDR to the shift register and transmission has started;
– TDR is empty;
– the next data can be written to USART_DR without risk of overwriting the previous data.

This flag generates an interrupt if the TXE interrupt enable bit is set.

During transmission, a write to USART_DR loads TDR, and the contents of TDR are then copied into the shift register after the current transmission completes.

If no transmission is in progress, a write to USART_DR loads the shift register directly, starts transmission, and TXE is set immediately.

If a frame is being transmitted (after the stop bit) and TXE is set, the TC bit is driven high (logic 1). An interrupt is generated if TCIE is set in USART_CR1.

After writing the last data to USART_DR, it is important to wait until TC is set to 1 before disabling the USART or placing the IC into a low-power mode (see Figure 115).

TC is cleared by the following software sequence:

1. read USART_SR;
2. write USART_DR.

Note: TC can also be cleared by writing 0 to it. It is recommended to use this sequence to clear TC only in multi-buffer transfers.

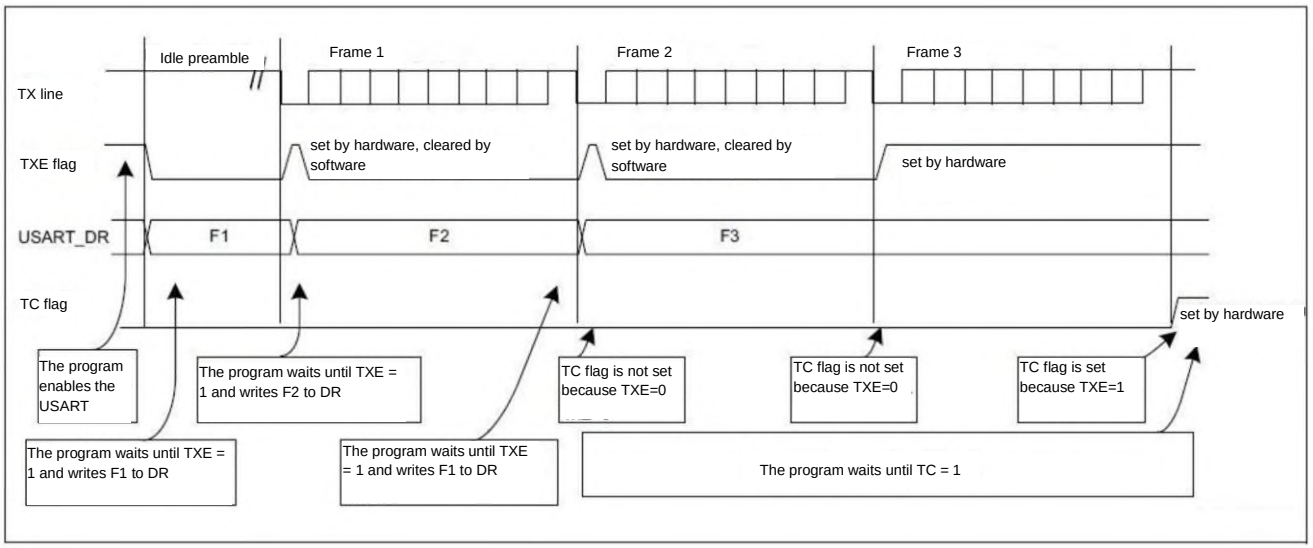


Figure 115 – TC/TXE status during transmission

31.3.3. **Receiver**

The USART receiver can accept an 8-bit or 9-bit data word, depending on the value of bit M in the USART_CR1 register.

31.3.3.1. **Start-bit detection**

The start-bit detection sequence is the same for oversampling by 16 and by 8.

In the USART transceiver, the start bit is detected when the following sampling pattern is observed on the line: 1110X0X0X0000.

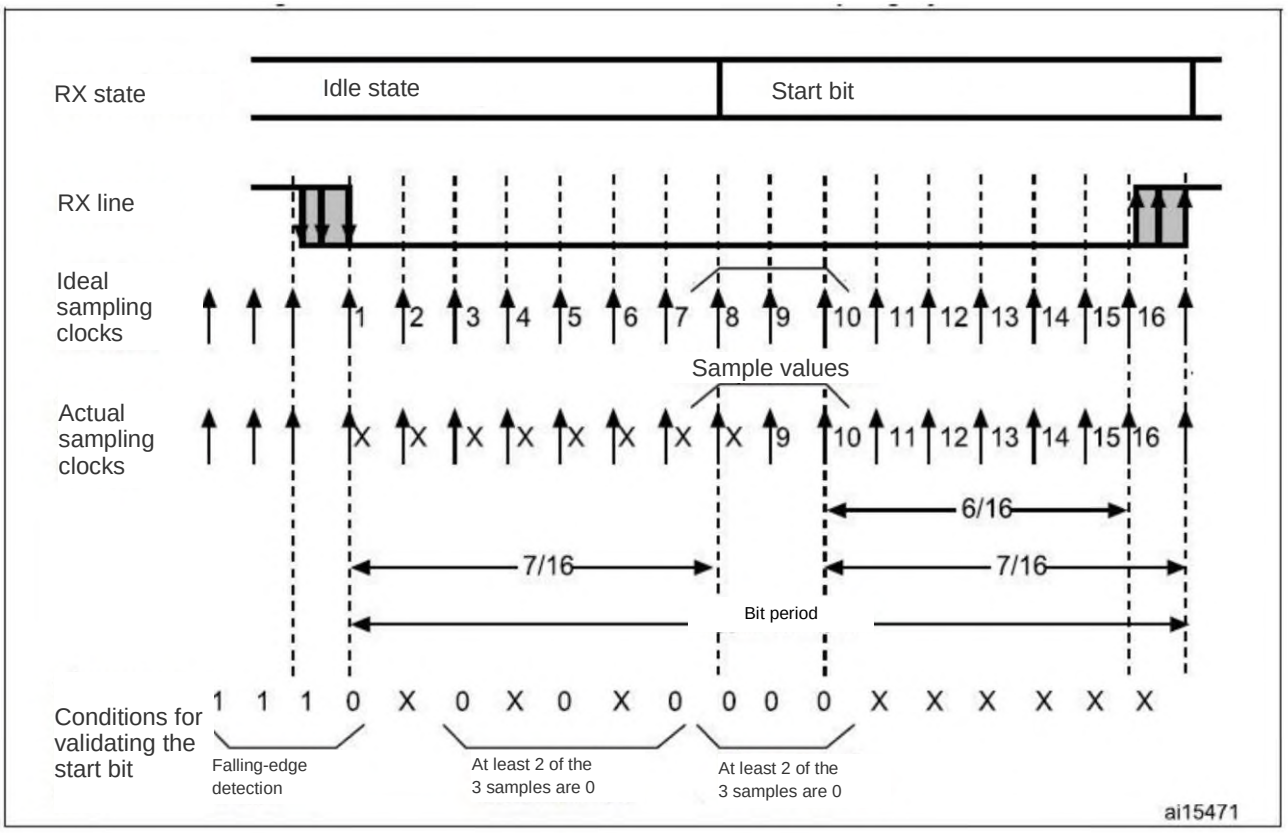


Figure 116 – Start-bit detection with oversampling by 16 and 8

Note: If the sequence is not completed, start bit detection is aborted, the receiver returns to the idle state (no flags are set), and waits for the next falling edge.

The start bit is confirmed (RXNE flag set and an interrupt generated when RXNEIE=1) if 3 samples are 0 (the first sampling at bit times 3, 5, and 7 finds three 0 bits, and the second sampling at bit times 8, 9, and 10 also finds three 0 bits).

The start bit is confirmed (RXNE flag set and an interrupt generated when RXNEIE=1), and the NF flag is set if, for both groups, at least 2 out of 3 samples are 0 (sampling at bit times 3, 5, 7 and at bit times 8, 9, 10). If this condition is not met, the start bit is not detected, and the receiver returns to the idle state (no flags are set).

If, for only one of the groups (sampling at bit times 3, 5, 7 or at bit times 8, 9, 10), 2 out of 3 samples are 0, the start bit is confirmed, but the NE flag is set.

31.3.3.2. **Character reception**

During reception, bits are shifted in on RX starting from the LSB. In this mode, the USART_DR register contains a buffer (RDR) between the internal bus and the receiver shift register.

Programming procedure:

1. Enable the USART by setting bit UE in the USART_CR1 register to 1.
2. Program bit M in the USART_CR1 register to select the data word length.
3. Program the number of stop bits in the USART_CR2 register.
4. Select the required baud rate using the USART_BRR register
5. Set bit RE in the USART_CR1 register. This enables the receiver and starts start bit detection.

When a character has been received:

- Bit RXNE is set. This indicates that the contents of the shift register have been transferred into RDR. In other words, the data have been received and can be read (together with the associated error flags).
- An interrupt is generated if bit RXNEIE is set.
- An error flag is set if a framing, noise, or overrun error is detected during reception.
- In multibuffer mode, bit RXNE is set after each received byte and is cleared by reading the data register.
- In single buffer mode, bit RXNE is cleared by software when the register is read;
- USART_DR. The RXNE flag can be cleared by writing 0. Bit RXNE must be cleared before reception of the next character is completed to avoid an overrun error.

Note: Bit RE must not be cleared while data is being received. If RE is disabled while a character is being received, reception of the current byte is aborted.

31.3.3.3. **Overrun error**

An overrun error occurs when a character is received while RXNE is still set. Data cannot be transferred from the shift register to RDR until RXNE has been cleared.

The RXNE flag is set after each received byte. An overrun error occurs if RXNE is still set when the next data word is received. When an overrun error occurs:

- Bit ORE is set.

- The contents of RDR are not lost. The previous data remain available when reading USART_DR.
- The shift register is overwritten. Any data received at the moment of the overrun error are lost.
- An interrupt is generated if RXNEIE or EIE is set.
- Bit ORE is cleared by reading USART_SR followed by a read of USART_DR.

Note: A set ORE bit indicates that at least one data element has been lost. There are two cases:

- If RXNE=1, the last valid data is stored in RDR and can be read;
- If RXNE=0, the last valid data has already been read, so there is nothing to read from RDR. This may occur when the last valid data is read from RDR at the same time as new data is received. It may also occur when new data is received during the read sequence (between the read access to USART_SR and the read access to USART_DR).

31.3.3.4. **Selecting the oversampling method**

The receiver uses several oversampling techniques, configurable by the user (except in synchronous mode), to recover data and separate it from noise.

The oversampling method is selected by programming bit OVER8 in the USART_CR1 register. Oversampling can be 16x or 8x the baud rate clock (see Figures 128 and 129).

Depending on the application:

- Oversampling by 8 (OVER8=1) is used to achieve a higher baud rate (up to fPCLK/8). In this case the maximum clock-error tolerance of the receiver is reduced (see Receiver USART clock-error tolerance).
- Oversampling by 16 (OVER8=0) is used to increase receiver clock-error tolerance. In this case the maximum baud rate is limited to fPCLK/16.

Programming bit ONEBIT in the USART_CR3 register selects the method used to evaluate the logic level. There are two options:

- Majority voting from 3 samples near the center of the received bit. In this case, if the 3 samples are not identical, the NF bit is set;
- A single sample at the center of the received bit.

Depending on the application:

- Majority sampling from 3 samples (ONEBIT=0) is used in noisy conditions, and data is discarded when noise is detected, as this indicates a failure during sampling.
- Single sampling (ONEBIT=1) is used when the line is noise-free, to increase the receiver clock error tolerance (see Receiver USART clock-error tolerance). In this case NF is not set.

When noise is detected in a frame:

- NF is set on the rising edge of RXNE.
- Invalid data is transferred from the shift register to USART_DR.
- In single-byte transfer mode, no dedicated interrupt is generated. However, the bit is set at the same time as RXNE, and RXNE itself generates an interrupt. In multibuffer mode, an interrupt is generated if bit EIE in the USART_CR3 register is set.

NF is cleared by reading USART_SR followed by a read of USART_DR.

Note: Oversampling by 8 is not available in Smartcard modes. In this mode the OVER8 bit is forced to 0 by hardware.

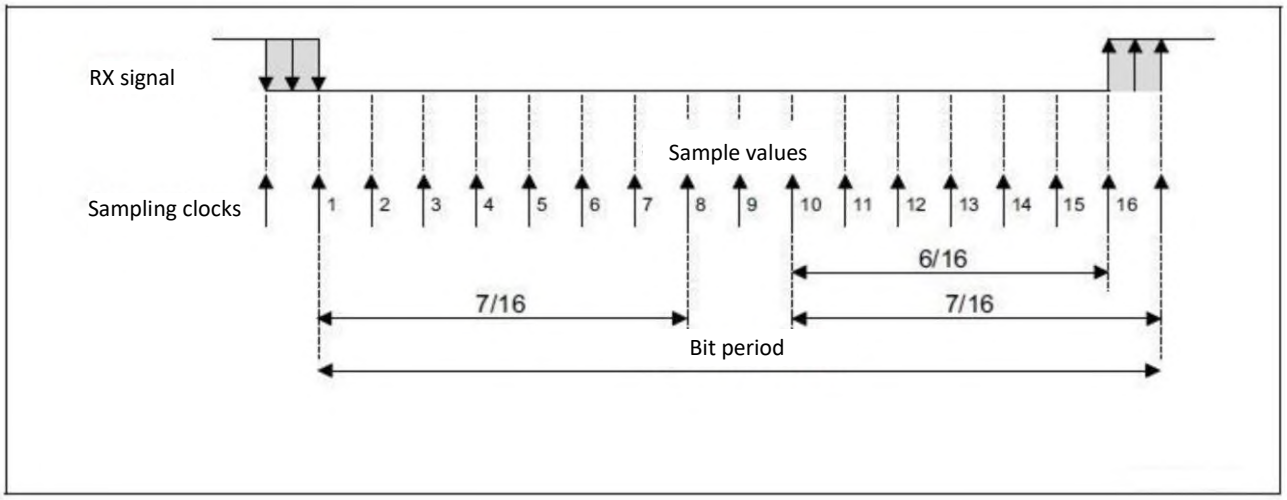


Figure 117 – Data sampling with oversampling by 16

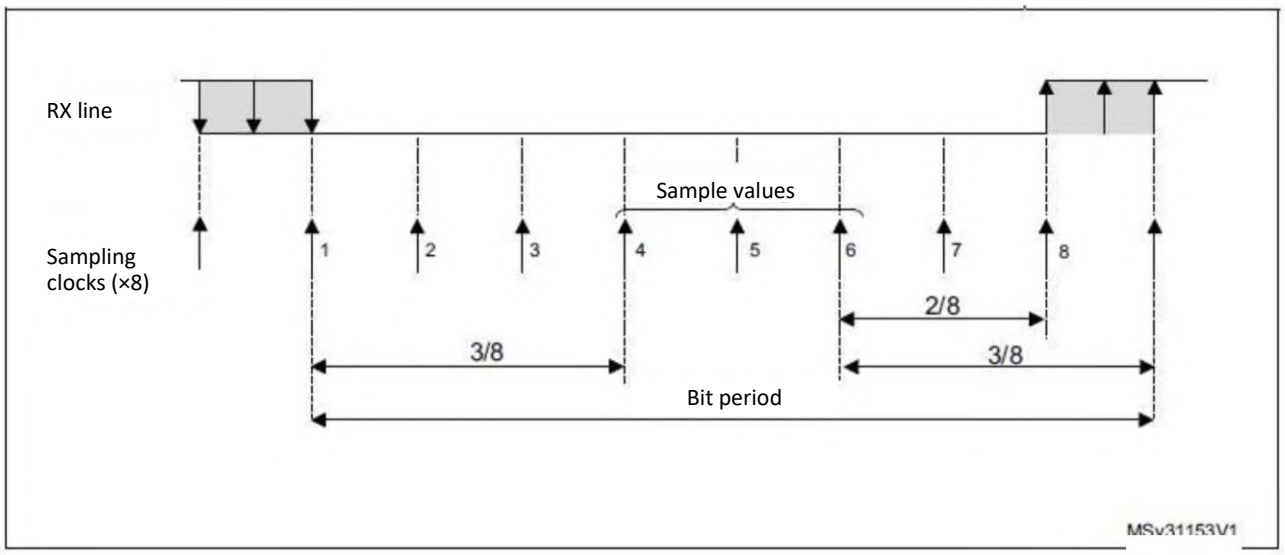


Figure 118 – Data sampling with oversampling by 8

Table 572 – Noise detection for received data

Value	NF status	Received bit value
000	0	0
001	1	0
010	1	0
011	1	1
100	1	0
101	1	1
110	1	1
111	0	1

31.3.3.5. **Framing error**

A framing error is detected when the stop bit is not recognized during reception within the expected time window because of loss of synchronization or excessive noise.

When a framing error is detected:

– Bit FE is set by hardware.

– Invalid data is transferred from the shift register to USART_DR.

– In single-byte transfer mode, no interrupt is generated. However, the bit is set at the same time as RXNE, and RXNE itself generates an interrupt. In multibuffer mode, an interrupt is generated if bit EIE in the USART_CR3 register is set.

FE is cleared by reading USART_SR followed by a read of USART_DR.

31.3.3.6. **Configurable stop bits in reception**

The number of stop bits to be received is configurable in Control Register 2. In normal mode, 2 stop bits are supported; in Smartcard mode, 0.5 or 1.5 stop bits are supported.

1 ***0.5 stop bit (Smartcard reception mode***): no sampling is performed for this mode. As a result, when 0.5 stop bit is selected, a framing error or break cannot be detected.

3 ***1.5 stop bits (Smartcard mode***): when data is transmitted in Smartcard mode, the device must verify the validity of the transmitted data. Thus, the receiver block must be enabled (RE=1 in USART_CR1), and the stop bit is checked for parity error. If a parity error is present, the smartcard pulls the data line to logic 0 during sampling (NACK signal), which is recognized as a framing error. FE is then set together with RXNE at the end of the 1.5 stop bits. For 1.5 stop bits, the level is evaluated at samples 16, 17, and 18 (1 bit period after the start of the stop bit). The 1.5 stop bits can be viewed as 2 parts: a first part of 0.5 bit period during which nothing happens, and a second part of 1 normal stop bit period during which the signal is evaluated. See Smartcard mode.

4 ***2 stop bits***: Evaluation for 2 stop bits is performed at samples 8, 9, and 10 of the first stop bit. If a framing error is detected during the first stop bit, the error flag is set. The second stop bit is not checked for a framing error. RXNE is set at the end of the first stop bit.

31.3.4. **Fractional baud rate generator**

The receiver and transmitter baud rates (Rx and Tx) are set to the same value, programmed by the Mantissa (integer part) and Fraction (fractional part) of the USARTDIV divider.

The baud rate for the standard USART (including SPI mode) is calculated by the formula

$$\text{Tx/Rx baud} = \frac{f_{CK}}{8 \cdot (2 - OVER8) \cdot USARTDIV}. \quad (34)$$

The baud rate for Smartcard modes is calculated by the formula

$$\text{Tx/Rx baud} = \frac{f_{CK}}{16 \times USARTDIV}, \quad (35)$$

where USARTDIV is an unsigned fixed-point value programmed in the USART_BRR register.

- If OVER8 = 0, the fractional part is encoded in 4 bits and programmed via DIV_Fraction[3:0] in USART_BRR.

– If OVER8 = 1, the fractional part is encoded in 3 bits and programmed via DIV_Fraction[2:0] in USART_BRR; bit DIV_Fraction[3] must be cleared.

Note: The baud rate counters are updated from the baud rate register after a write to USART_BRR. Therefore, the baud rate register value must not be changed while a data transfer is in progress.

31.3.4.1. **Examples of calculating USARTDIV and USART_BRR for OVER8 = 0**

Example 1:

If DIV_Mantissa = 0d27 and DIV_Fraction = 0d12 (USART_BRR = 0x1BC), then:
Mantissa (USARTDIV) = 0d27
Fraction (USARTDIV) = 12/16 = 0d0.75
Therefore, USARTDIV = 0d27.75

Example 2:

To program USARTDIV = 0d25.62:
then:
DIV_Fraction = 16 • 0d0.62 = 0d9.92
The nearest integer is 0d10 = 0xA
DIV_Mantissa = mantissa (0d25.620) = 0d25 = 0x19
Then USART_BRR = 0x19A, and USARTDIV = 0d25.625

Example 3:

To program USARTDIV = 0d50.99:
then:
DIV_Fraction = 16 • 0d0.99 = 0d15.84
The nearest integer is 0d16 = 0x10 => overrun of DIV_Fraction[3:0] =>a carry must be added to the mantissa.
DIV_Mantissa = mantissa (0d50.990 + carry) = 0d51 = 0x33
Then USART_BRR = 0x330, and USARTDIV = 0d51.000

31.3.4.2. **Examples of calculating USARTDIV and USART_BRR for OVER8 = 1**

Example 1:

If DIV_Mantissa = 0x27 and DIV_Fraction[2:0] = 0d6 (USART_BRR = 0x1B6), then:
Mantissa (USARTDIV) = 0d27
Fraction (USARTDIV) = 6/8 = 0d0.75
Therefore USARTDIV = 0d27.75

Example 2:

To program USARTDIV = 0d25.62:
then:
DIV_Fraction = 8 • 0d0.62 = 0d4.96
The nearest integer is 0d5 = 0x5
DIV_Mantissa = mantissa (0d25.620) = 0d25 = 0x19
Then USART_BRR = 0x195, and USARTDIV = 0d25.625

Example 3:

To program USARTDIV = 0d50.99:

then:

$$DIV\_Fraction = 8 \cdot 0\ d\ 0.99 = 0\ d\ 7.92$$

The nearest integer is 0d8 = 0x8 => overrun of DIV_Fraction[2:0] => a carry must be added to the mantissa.

DIV_Mantissa = mantissa (0d50.990 + carry) = 0d51 = 0x33
Then, USART_BRR = 0x0330 => USARTDIV = 0d51.000

Table 573 – Determining the error for programmed baud rates at f_{PCLK} = 60 MHz(1)

No.	Required data rate	Oversampling by 16 (OVER8 = 0)			Oversampling by 8 (OVER8 = 1)		
		Actual data rate	Value in USART_BRR baud-rate register	Error, %	Actual data rate	Value in baud-rate register USART_BRR	Error, %
1	2.4 Kbps	2.4 Kbps	1562.5	0	2.4 Kbps	3125	0
2	9.6 Kbps	9.6 Kbps	390.625	0	9.6 Kbps	781.25	0
3	19,2 Kbps	19,2 Kbps	195.3125	0	19,2 Kbps	390.625	0
4	57.6 Kbps	57.582 Kbps	65.125	0.03	57.582 Kbps	130.25	0.03
5	115.2 Kbps	115.163 Kbps	32.5625	0.03	115.163 Kbps	65.125	0.03
6	230.4 Kbps	230.769 Kbps	16.25	0.16	230.769 Kbps	32.5	0.16
7	460.8 Kbps	461.538 Kbps	8.125	0.16	461.538 Kbps	16.25	0.16
8	896 Kbps	895.522 Kbps	4.1875	0.05	895.522 Kbps	8.375	0.05
9	921.6 Kbps	923.077 Kbps	4.0625	0.16	923.077 Kbps	8.125	0.16

10	2 Mbps	2 Mbps	1.875	0	2 Mbps	3.75	0
11	3 Mbps	3 Mbps	1.25	0	3 Mbps	2.5	0
12	4 Mbps	NA	NA	NA	4 Mbps	1.875	0
13	5 Mbps	NA	NA	NA	5 Mbps	1.5	0
14	6 Mbps	NA	NA	NA	6 Mbps	1.25	0

(1) The lower the CPU clock frequency, the lower the accuracy for a given baud rate value. The upper limit of the achievable baud rate can be derived from these data.

31.3.5. **USART receiver clock-error tolerance**

The asynchronous USART receiver operates correctly only if the total clock deviation remains greater than the USART receiver tolerance. Contributors to the total deviation:

- DTRA: deviation due to transmitter error (including deviation of the transmitter's local clock generator);
- DQUANT: receiver baud-rate quantization error;
- DREC: deviation of the receiver's local clock generator;
- DTCL: deviation introduced by the transmission line (usually because transceivers can have asymmetric 0 to 1 versus 1 to 0 transition times).

$$DTRA + DQUANT + DREC + DTCL < \text{receiver tolerance } USART$$

The USART receiver tolerance for correct data reception is the maximum allowable deviation and depends on:

- character length of 10 or 11 bits, determined by bit M in USART_CR1;
- oversampling by 8 or 16, determined by bit OVER8 in USART_CR1;
- whether fractional baud rate generation is used;
- whether 1 or 3 samples per bit are used for digitizing the data, depending on bit ONEBIT in USART_CR3.

Table 574 – USART receiver tolerance for DIV_Fraction = 0

Bit M	OVER8 = 0		OVER8 = 1	
	ONEBIT=0	ONEBIT=1	ONEBIT=0	ONEBIT=1
0	3.75%	4.375%	2.50%	3.75%
1	3.41%	3.97%	2.27%	3.41%

Table 575 – USART receiver tolerance for DIV_Fraction not equal to 0

Bit M	OVER8 = 0		OVER8 = 1	
	ONEBIT=0	ONEBIT=1	ONEBIT=0	ONEBIT=1
0	3.33%	3.88%	2%	3%
1	3.03%	3.53%	1.82%	2.73%

31.3.6. **Multiprocessor communication**

USART supports multiprocessor communication (several USART transceivers connected in a single network). For example, one USART can act as the master device, with its TX output connected to the RX input of another USART transceiver. The other transceivers act as slaves, their TX outputs are wire-ANDed together and connected to the RX input of the master.

In multiprocessor configurations it is preferable that only one receiver actively accepts an entire message, in order to reduce processing of redundant data on receivers for which the data is not intended.

Devices that are not addressed can be placed into mute mode by means of the receiver wakeup feature. In this mode:

- no receive status bits can be set;
- all receive interrupts are disabled;
- bit RWU in USART_CR1 is set to 1. Bit RWU can be controlled automatically by hardware or written by software under specific conditions.

USART can enter or exit mute mode in one of two ways, depending on the value of bit WAKE in USART_CR1:

- Idle line detection, if WAKE is cleared;
- Address mark detection, if WAKE is set.

31.3.6.1. **Idle line detection (WAKE=0)**

USART enters mute mode when RWU is set to 1. The transceiver leaves mute mode when an Idle frame is detected. RWU is then cleared by hardware, but the IDLE flag in USART_SR is not set. RWU can also be cleared to 0 by software.

Figure 119 shows an example of device behavior in mute mode using Idle line detection.

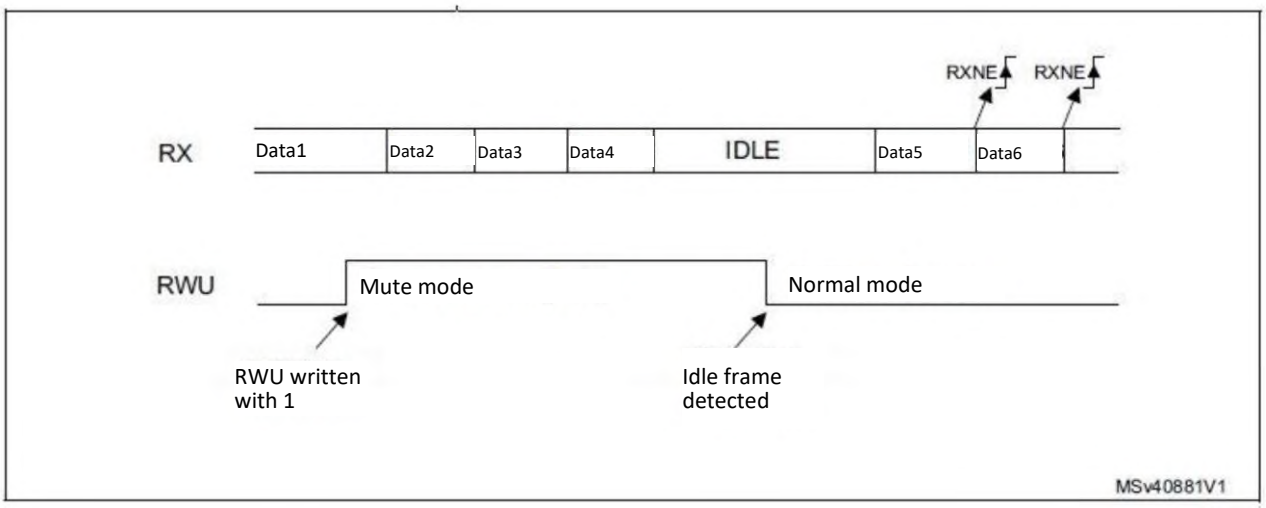


Figure 119 – Mute mode with Idle line detection

31.3.6.2. **Address mark detection (WAKE = 1)**

In this mode, bytes are recognized as address bytes if the MSB is 1, otherwise they are recognized as data bytes. In an address byte, the receiver address is placed in the 4 LSBs. The receiver compares this 4-bit value with its own address, programmed in the ADD bits of USART_CR2.

USART enters mute mode when an address character is received that does not match the programmed address. In this case RWU is set by hardware. RXNE is not set for this address byte and no interrupt is generated, because the transceiver has switched to mute mode.

The transceiver leaves mute mode when an address character matching the programmed address is received. RWU is then cleared and subsequent data bits are accepted. The address character is then presented with RXNE set, since RWU has been cleared.

RWU can be written with 0 or 1 only when the receive buffer is empty (RXNE = 0 in USART_SR). Otherwise the write is ignored.

Figure 120 shows an example of transceiver behavior in mute mode with address mark detection.

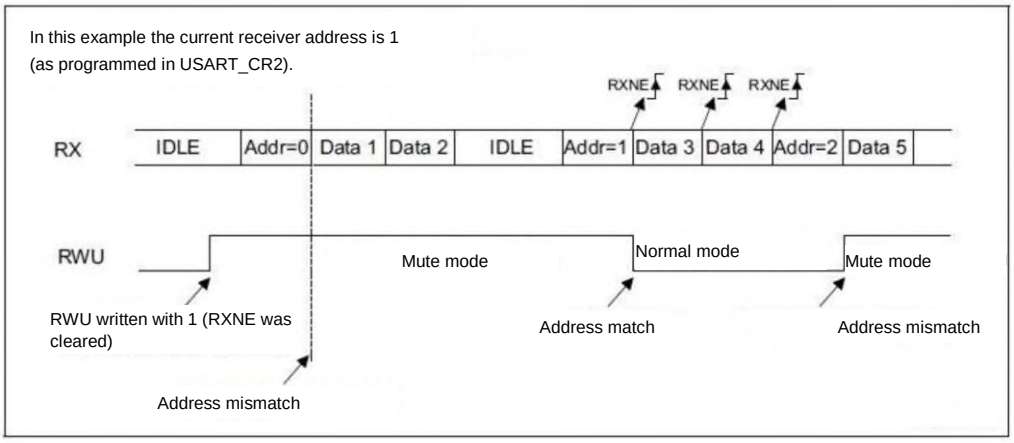


Figure 120 – Mute mode with address mark detection

31.3.7. **Parity check**

Parity check (generation of a parity bit on transmit and checking of a parity bit on receive) is enabled by setting bit PCE in USART_CR1. Depending on the frame length selected by bit M, the possible USART frame formats are shown in Table 576.

Table 576 – Frame formats

Bit M	Bit PCE	USART frame(1)
0	0	| SB | 8 data bits | STB |
0	1	| SB | 7 data bits | PB | STB |
1	0	| SB | 9 data bits | STB |
1	1	| SB | 8 data bits | PB | STB |

(1) SB: start bit, STB: stop bit, PB: parity bit

31.3.7.1. **Even parity**

The parity bit is computed so that the total number of “1” bits in the 7- or 8-bit data field (depending on bit M) plus the parity bit is even.

For example, data=00110101; 4 bits are set => parity bit = 0 when even parity is selected (bit PS in USART_CR1 = 0).

31.3.7.2. **Odd parity**

The parity bit is computed so that the total number of “1” bits in the 7- or 8-bit data field (depending on bit M) plus the parity bit is odd.

For example, data=00110101; 4 bits are set => parity bit = 1 when odd parity is selected (bit PS in USART_CR1 = 1).

31.3.7.3. **Parity checking on reception**

If a parity error is detected, flag PE in USART_SR is set and an interrupt is generated if bit PEIE in USART_CR1 is set. PE is cleared by the standard software sequence (a read of the status register followed by a read or write of USART_DR).

Note: In address mark wakeup mode: The MSB is used to identify an address, not as a parity bit. The receiver does not check the parity bit of the address byte (PE is not set in case of a parity error in an address byte).

31.3.7.4. **Parity bit generation during transmission**

If bit PCE is set in USART_CR1, the MSB of the data written to the data register is transmitted, but it is overridden with the parity bit (an even number of “1” bits if even parity is selected (PS=0); an odd number of “1” bits if odd parity is selected, (PS=1).

Note: The software that controls transmission can trigger the software sequence used to clear the PE flag (read of the status register followed by a read or write access to the data register). In half-duplex mode this may, depending on the software, lead to an unintended clearing of the PE flag.

31.3.8. **Synchronous USART mode**

Synchronous mode is selected by setting bit CLKEN to “1” in the USART_CR2 register.

In synchronous mode the following bits must be cleared:

– SCEN and HDSEL in the USART_CR3 register.

The USART supports control of bidirectional synchronous serial communication in master transceiver mode. Pin CK is the USART transmitter clock output. During the start and stop bits, no clock pulses are output on CK. Depending on the state of bit LBCL in USART_CR2, clock pulses are either generated or suppressed during the last valid data bit (address mark). Bit CPOL in USART_CR2 selects the clock polarity, and bit CPHA in USART_CR2 selects the phase of the external clock (see Figures 121, 122, and 123).

During the Idle state, the preamble, and the break character, the external CK clock signal is inactive.

In synchronous mode, the transmitter operates in the same way as in asynchronous mode. However, since CK is synchronized with TX (depending on CPOL and CPHA), the data on TX is synchronous.

The receiver in this mode operates differently from the asynchronous mode. If RE=1, data is sampled synchronously with CK (on the rising or falling edge, depending on CPOL and CPHA), without oversampling. The setup and hold times must be met (they depend on the baud rate: 1/16 of a bit time).

Note: CK operates together with TX. Clock pulses are therefore generated only when the transmitter is enabled (TE=1) and data is being transmitted (USART_DR has been written). This implies that synchronous data cannot be received unless data is transmitted.

Bits LBCL, CPOL, and CPHA must be configured while the transmitter and receiver are disabled (TE=RE=0) to ensure correct clock behavior. These bits must not be changed while the transmitter or receiver is enabled.

It is recommended to set TE and RE in a single write operation in order to minimize the setup and hold time of the receiver.

The USART transceiver supports only master mode: it cannot receive or transmit data that is driven from an external clock input (CK always operates as an output).

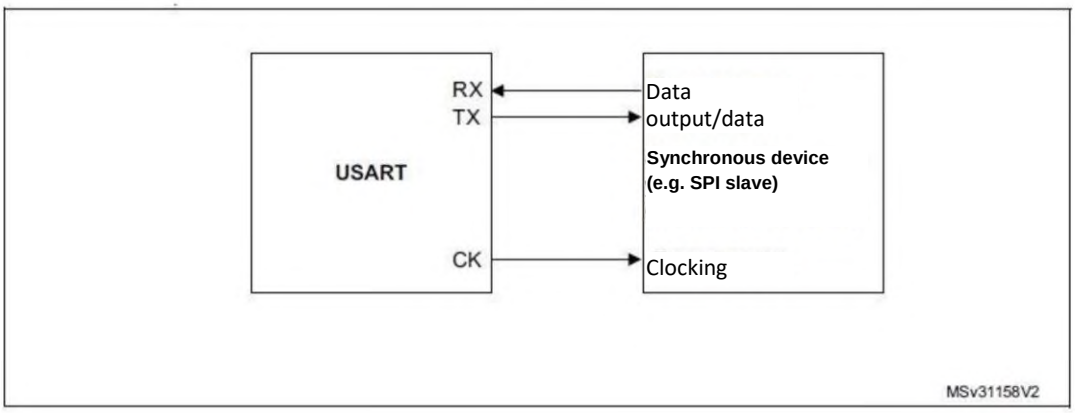


Figure 121 – Example of USART synchronous transfer

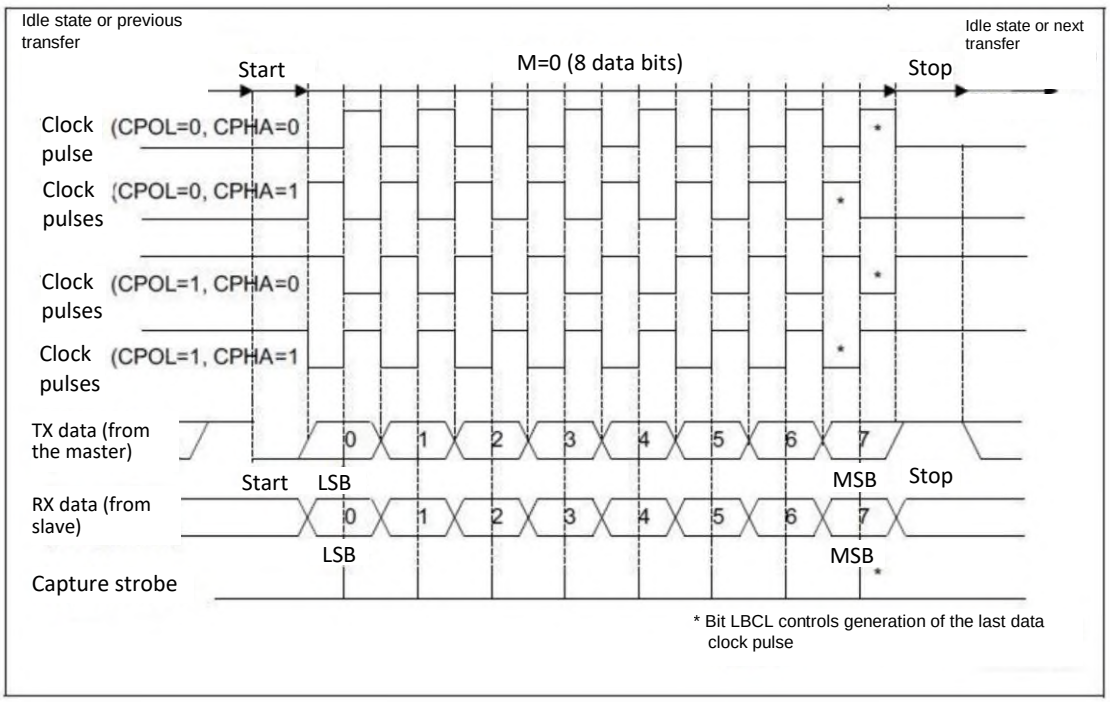


Figure 122 – USART data clocking diagram (M=0)

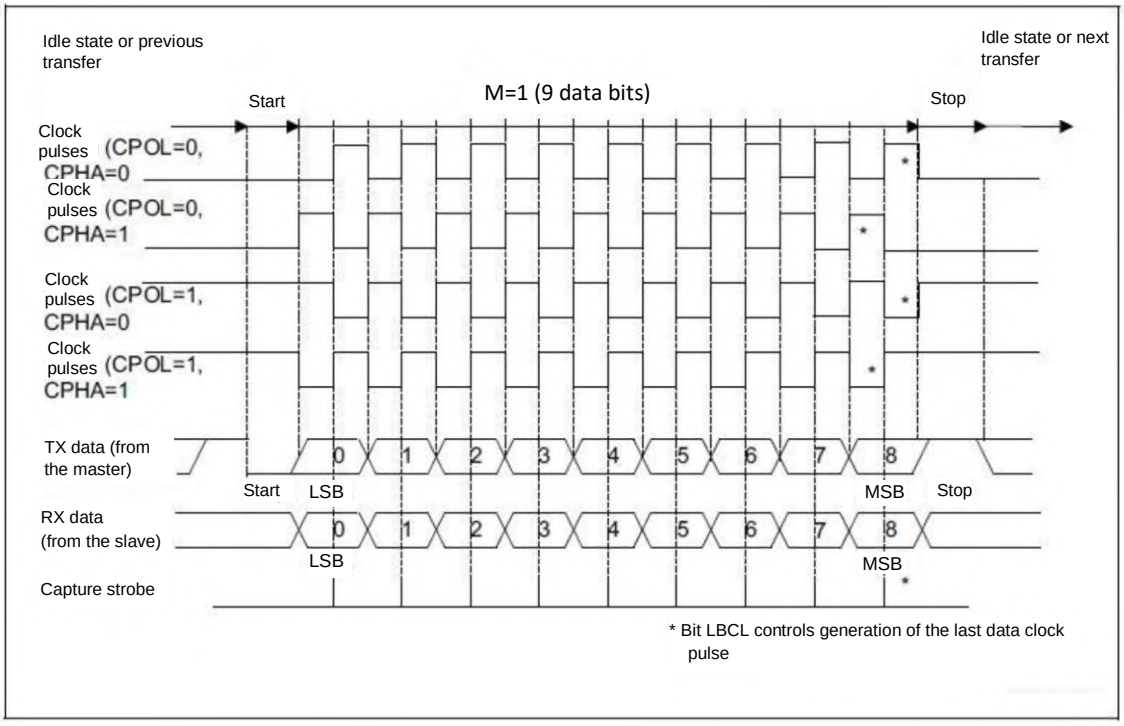


Figure 123 – USART data clocking diagram (M=1)

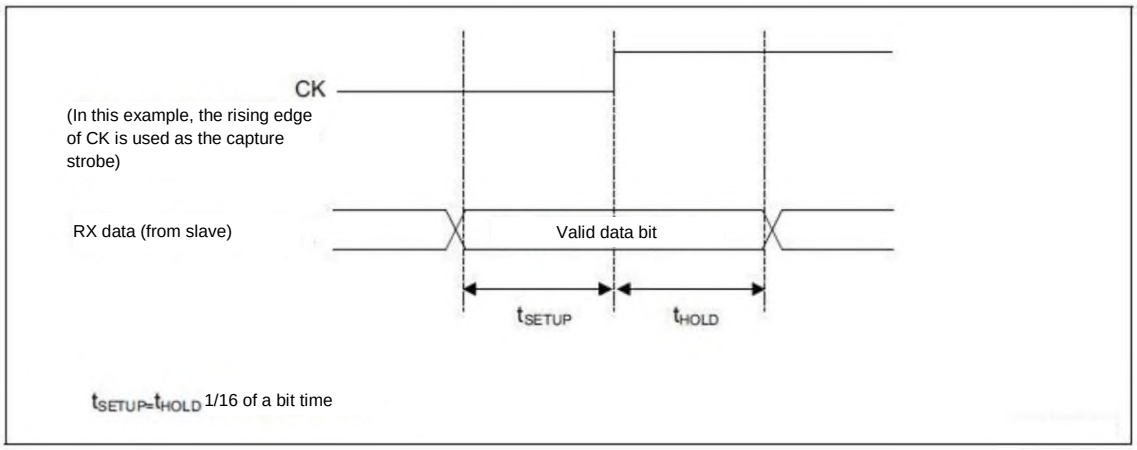


Figure 124 – RX setup/hold time

Note: The CK clock function differs in Smartcard mode (see the Smartcard mode description).

31.3.9. **Single-wire half-duplex communication**

Single-wire half-duplex mode is selected by setting bit HDSEL in the USART_CR3 register. In this mode the following bits must be cleared:

– SCEN in the USART_CR3 register.

USART can be configured to support a single-wire half-duplex protocol in which the TX and RX signal lines are internally connected. Selection between half-duplex and full-duplex communication is performed by the HALF DUPLEX SEL control bit (HDSEL in USART_CR3).

Once HDSEL is set to 1:

- TX and RX signal lines are internally tied together;
- the RX pin is no longer used;
- the TX pin is always released when no data is being transmitted. Thus, TX is used as a standard input/output in Idle or during reception. This input/output must be configured so that it behaves as a high-impedance input (or open-drain output) when it is not driven by USART.

Apart from this, data transfer proceeds as in normal USART communication mode.

Contention on the communication line must be handled by firmware (for example, using a centralized bus arbiter). In particular, transmission is never blocked by hardware and continues as soon as data is written to the data register, as long as bit TE is set.

31.3.10. Smartcard mode

Smartcard mode is selected by setting bit SCEN in the USART_CR3 register. In Smartcard mode, the following bits must be cleared:

- bit HDSEL in the USART_CR3 register.

In addition, bit CLKEN can be set to provide the smartcard clock.

The Smartcard mode interface is designed to support the asynchronous protocol in accordance with ISO 7816-3. USART must be configured as follows:

- 8 data bits + parity bit: M=1 and PCE=1 in the USART_CR1 register
- 1.5 stop bits for both transmission and reception: STOP=11 in the USART_CR2 register.

Note: 0.5 stop bit can also be selected for reception, but it is recommended to use 1.5 stop bits for both transmission and reception to avoid switching between two configurations.

Figure 125 shows an example of the data signal with and without a parity error.

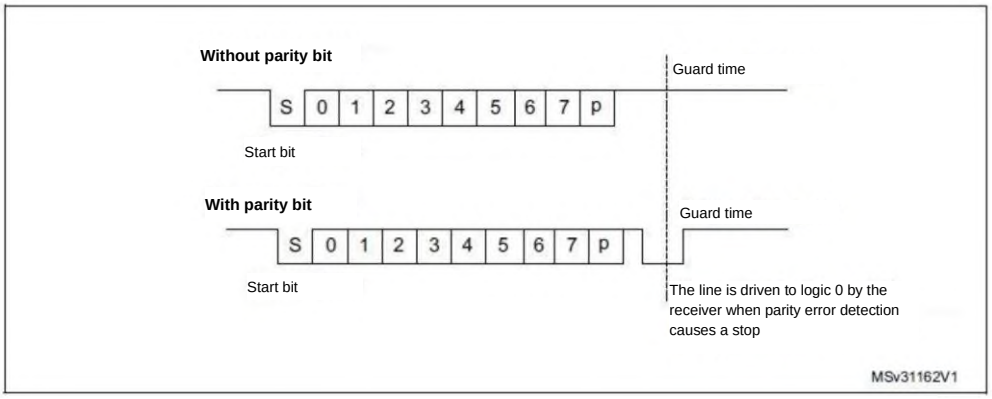


Figure 125 – ISO 7816-3 asynchronous protocol

When connected to a smartcard, the TX output of the USART transceiver drives a bidirectional line that is also driven by the smartcard. The TX pin must be configured as open-drain.

Smartcard is a single-wire half-duplex data transmission protocol.

- Transmission of data from the transmitter shift register is performed with a guaranteed delay of at least 1/2 bit period. In normal mode, the complete transfer from the shift register starts shifting on the next baud rate clock edge. In Smartcard mode, this transfer is additionally delayed by 1/2 bit period.
- If a parity error is detected when receiving a frame configured for 0.5 or 1.5 stop bits, the transmit line is pulled to 0 for one baud period after the end of frame reception. This indicates to the smartcard that the data sent to the USART transceiver were not received correctly. This NACK signal (pulling the transmit line to 0 for 1 baud period) generates a framing error on the transmitter side (1.5 stop-bit configuration). The application must handle this situation by retransmitting the data in accordance with the protocol. The parity error is not acknowledged by the receiver (no NACK signal) if the NACK control bit is set, otherwise, the NACK signal is not transmitted.
- Setting of the TC flag can be delayed by programming the guard time register. In normal operation, the TC flag is set when the transmitter shift register is empty and there are no pending transmit requests. In Smartcard mode, an empty transmitter shift register starts a guard-time counter that counts up to the value programmed in the guard time register. The TC flag is pulled to 0. When the counter reaches the programmed value, the TC flag is set.
- Clearing of the TC flag is not affected by Smartcard mode.
- If a framing error is detected on the transmitter side (NACK response from the receiver), the NACK signal is not recognized as a start bit by the receiver block inside the transmitter. According to the ISO standard, the duration of a received NACK signal may be 1 or 2 baud periods.
- On the receiver side, if a parity error has been detected and a NACK signal has been sent, the receiver does not interpret the NACK signal as a start bit.

Note: The break character has no special meaning in Smartcard mode. A 0x00 data value with a framing error is handled as data, not as a break character.

An Idle frame is not transmitted when toggling bit TE. The Idle frame (as defined for other configurations) is not defined in the ISO protocol.

Figure 126 shows how the NACK signal is handled. In this example, the USART is transmitting data and is configured for 1.5 stop bits. The USART receive block is enabled to check data integrity and the NACK signal.

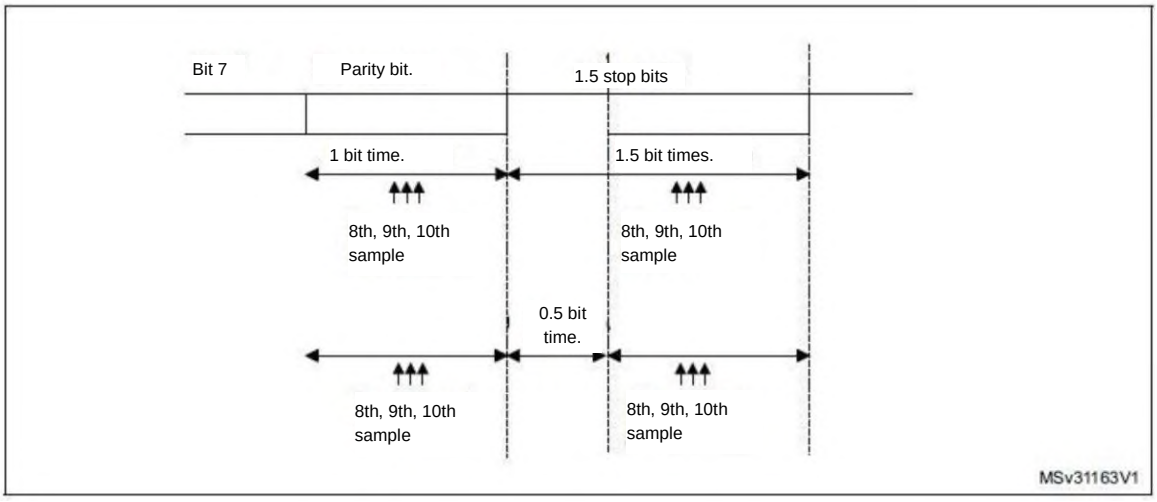


Figure 126 – Parity error detection using 1.5 stop bits

USART can provide the smartcard clock on the CK output. In Smartcard mode, the CK output is not tied to the data transfer, it only generates a clock derived from the internal peripheral clock input using a 5-bit prescaler. The division factor is configured in the USART_GTPR prescaler register. The CK frequency is programmable from $f_{CK}/2$ to $f_{CK}/62$, where f_{CK} is the peripheral clock input frequency.

31.3.11. **Hardware flow control**

The serial data flow between two devices can be controlled using the CTS input and RTS output. Figure 127 shows a connection diagram of two devices for this mode:

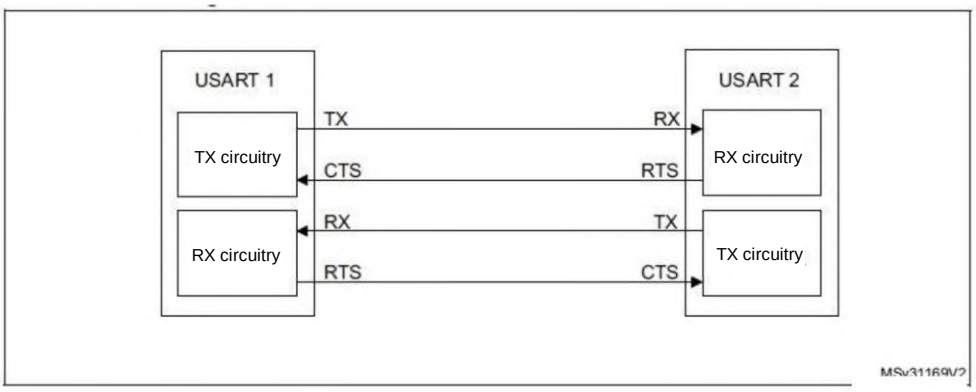


Figure 127 – Hardware data flow control between two USART devices

RTS and CTS flow control can be enabled independently by setting bits RTSE and CTSE, respectively, to 1 (in the USART_CR3 register).

31.3.11.1. **RTS flow control**

When RTS flow control is enabled (RTSE=1), the RTS signal (logic 0) is asserted when the USART receiver is ready to accept new data. If the receive register is full, the RTS signal is deasserted, indicating that transmission is expected to stop at the end of the current frame.

Figure 128 shows an example of communication with RTS flow control enabled.

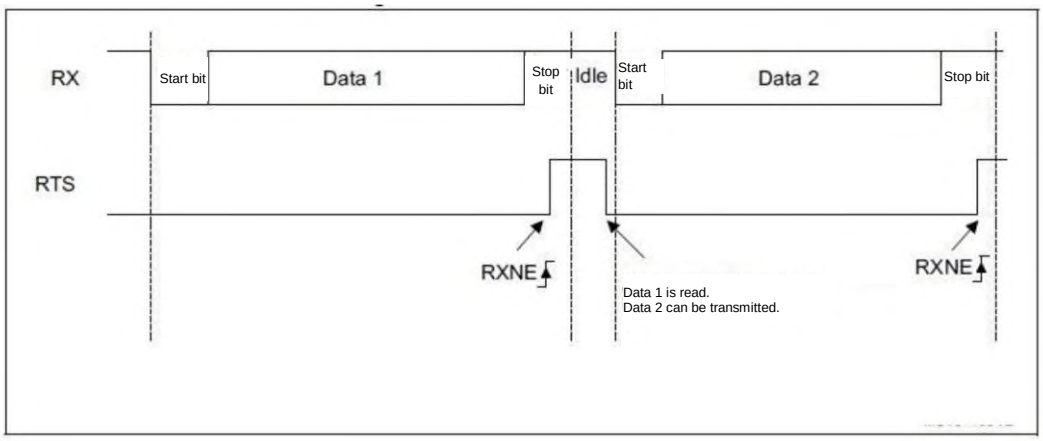


Figure 128 – RTS flow control

31.3.11.2. **CTS flow control**

If CTS flow control is enabled (CTSE=1), the transmitter samples the CTS input before sending the next frame. If CTS is asserted (logic 0), transmission of the next data begins (assuming data is pending for transmission, that is, TXE=0); otherwise, transmission does not start. When CTS is deasserted during an ongoing transmission, the current transmission is completed before the transmitter stops.

If CTSE=1, the CTSIF status flag is set automatically by hardware as soon as the CTS input toggles. This flag indicates when the receiver will be ready for data transfer. An interrupt is generated if the CTSIE bit is set in the USART_CR3 register. Figure 129 shows an example of communication with CTS flow control enabled.

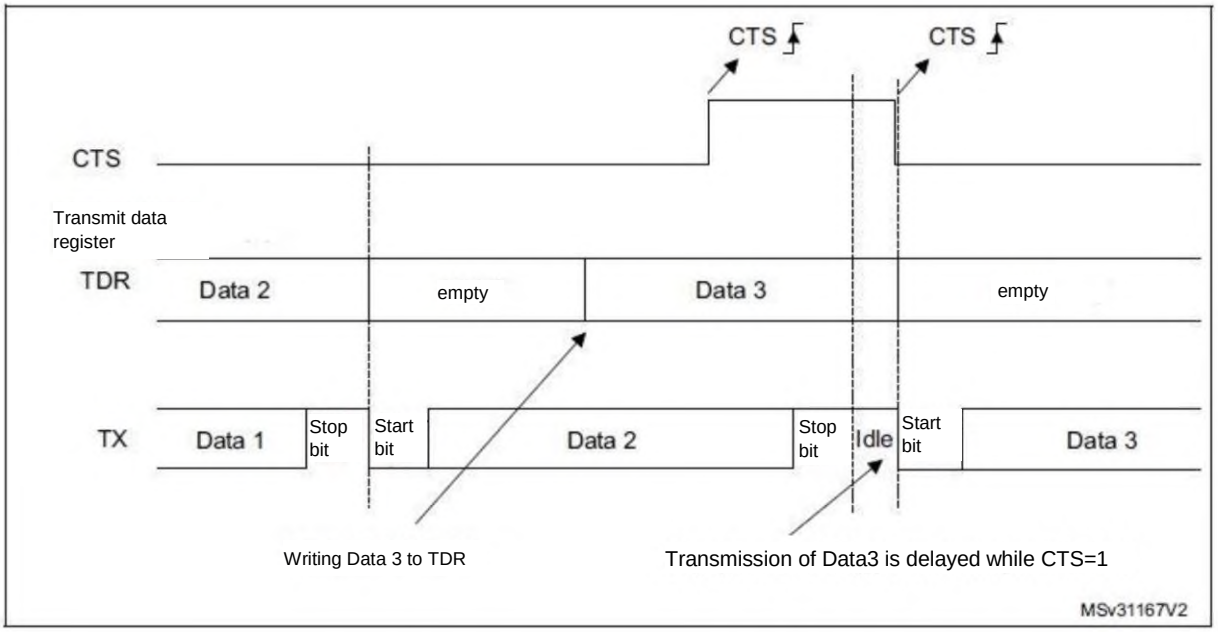


Figure 129 – CTS flow control

Note: Special behavior for break frames: when CTS flow control is enabled, the transmitter does not check the CTS input status when sending a break signal.

31.4. **USART interrupts**

Table 577 – USART interrupt requests

Event	Event flag	Enable bit
Transmit data register empty	TXE	TXEIE
CTS flag	CTS	CTSIE
Transmission complete	TC	TCIE
Received data ready to be read	RXNE	RXNEIE
Overrun error detected	ORE	
Idle line detected	IDLE	IDLEIE
Parity error	PE	PEIE
Noise, overrun, or framing error flag in multibuffer mode	NF or ORE or FE	EIE

The interrupt event sources are connected to the same interrupt vector (see Figure 130).

- During transmission: Transmission Complete, Clear to Send, or Transmit Data Register Empty.
- During reception: Idle line detection, overrun error, Receive Data Register Not Empty, parity error, noise flag (multibuffer mode only), and framing error (multibuffer mode only).

All of these events generate an interrupt if the corresponding interrupt-enable bit is set.

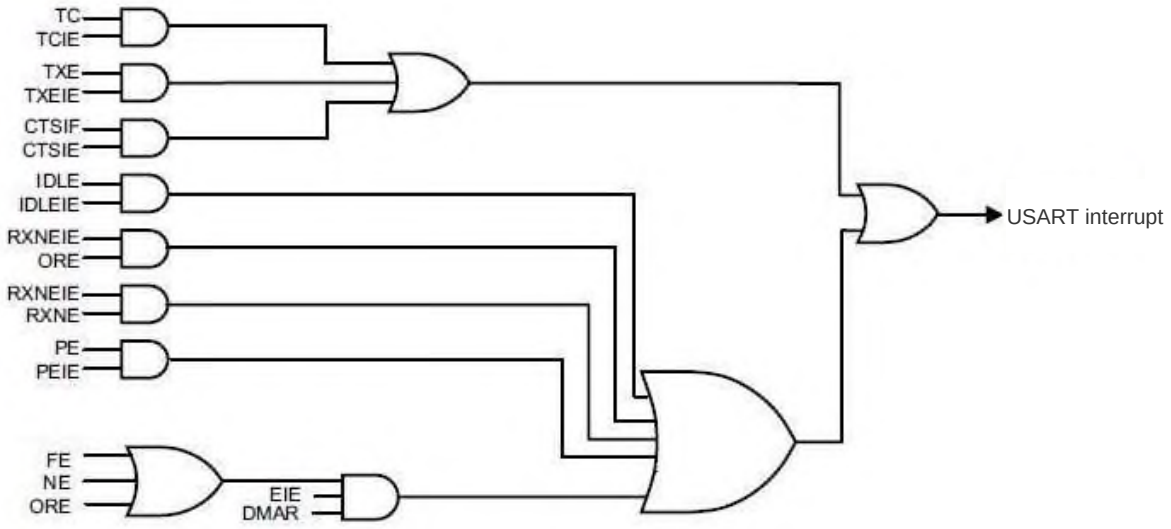


Figure 130 – Diagram of USART events to the interrupt

31.5. **USART registers**

Access to peripheral registers is performed using 16-bit half-words or 32-bit words. Tables 579–591 show the USART register map and the reset values of the registers.

Table 578 – USART_CNTR block registers

Base Address		Name	Description
0x500C_0000		USART_CNTR	USART controller block (ISO7816)
Offset			
0x0000	0	USART_SR	Status register
0x0004	1	USART_DR	Data register
0x0008	2	USART_BRR	Baud rate register
0x000c	3	USART_CR1	Control register 1
0x0010	4	USART_CR2	Control register 2
0x0014	5	USART_CR3	Control register 3
0x0018	6	USART_GTPR	Prescaler guard time register

31.5.1. **USART_SR**

Table 579 – USART_SR Register

Base ADDR=			0x500C_0000			Offset=		0x0000_0000			Reset=		0x0000_00C0		
REG Name:			USART_SR												
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
.-						CTS	-	TXE	TC	RXNE	IDLE	ORE	NF	FE	PE
						RW		R	RW	RW	R	R	R	R	R

Table 580 – Description of USART_SR register bits

Bit	Name	Value	Description
31…10	-	0	Reserved
9	CTS	0	CTS flag. This bit is set by hardware when the level on the CTS input changes, if CTSE is set. The bit is cleared by writing 1. An interrupt is generated if CTSIE=1 in the USART_CR3 register. 0 – no CTS line status change detected; 1 – CTS line level change detected.
8	-	0	Reserved
7	TXE	1	Transmit Data Register Empty. This bit is set by hardware when the contents of the TDR register have been transferred to the shift register. An interrupt is generated if TXEIE=1 in the USART_CR1 register. The bit is cleared by writing to the USART_DR register. 0 – data have not been transferred into the shift register; 1 – data have been transferred into the shift register. Note: This bit is used during single-buffer transmission.
6	TC	1	Transmission Complete. This bit is set by hardware when transmission of a frame containing data is complete and TXE is set. An interrupt is generated if TCIE=1 in the USART_CR1 register. The bit is cleared by a software sequence (read of USART_SR followed by a write to USART_DR). The TC bit can also be cleared by writing 1. This clear is recommended only in multibuffer transfers. 0 – transmission not complete; 1 – transmission complete.
5	RXNE	0	Receive Data Register Not Empty. This bit is set by hardware when the contents of the RDR shift register have been transferred into the USART_DR register. An interrupt is generated if RXNEIE = 1 in the USART_CR1 register. The bit is cleared by reading the USART_DR register. The RXNE flag can also be cleared by writing 1. This clear sequence is recommended for multibuffer transfers. 0 – no data received; 1 – received data ready to be read.

Bit	Name	Value	Description
4	IDLE	0	Idle line detected. This bit is set by hardware when an Idle line is detected. An interrupt is generated if IDLEIE=1 in the USART_CR1 register. The bit is cleared by a software sequence (read of USART_SR followed by a read of USART_DR). 0 – Idle line not detected; 1 – Idle line detected. Note: The IDLE bit will not be set again until RXNE becomes set (that is, until an Idle condition has again been detected on the line).
3	ORE	0	Overrun error. This bit is set by hardware when the word currently in the shift register is ready to be transferred into RDR while RXNE=1. An interrupt is generated if RXNEIE = 1 in the USART_CR1 register. The bit is cleared by a software sequence (read of USART_SR followed by a read of USART_DR). 0 – no overrun error; 1 – overrun error detected. Note: When this bit is set, the contents of the RDR register are not lost, but the shift register is overwritten. In multibuffer mode, an interrupt is generated on ORE if the EIE bit is set.
2	NF	0	Noise flag. This bit is set by hardware when noise is detected in a received frame. The bit is cleared by a software sequence (read of USART_SR followed by a read of USART_DR). 0 – no noise detected; 1 – noise detected. Notes 1 This bit does not generate an interrupt by itself, because it is asserted at the same time as RXNE, which already generates an interrupt in multibuffer mode on NF flag. if the EIE bit is set. 2 If the line is noise-free, the NF flag can be disabled by setting 1 in ONEBIT to increase the allowed clock deviation of the transceiver (see section 31.3.5 “USART receiver clock-frequency tolerance”).

Bit	Name	Value	Description
1	FE	0	Framing error. This bit is set by hardware when desynchronization occurs, excessive noise is present, or a break character is received. The bit is cleared by a software sequence (read of USART_SR followed by a read of USART_DR). 0 – no framing error detected; 1 – framing error or break character detected. Notes 1 This bit does not generate an interrupt by itself, because it is asserted at the same time as RXNE, which already generates an interrupt in multibuffer mode on NF flag, if the EIE bit is set. 2 If the line is noise-free, the NF flag can be disabled by setting 1 in ONEBIT to increase the allowed clock deviation of the transceiver (see section 31.3.5 “USART receiver clock-frequency tolerance”).
0	PE	0	Parity error. This bit is set by hardware when a parity error is detected in receive mode. The bit is cleared by a software sequence (read of the status register followed by a read or write of the USART_DR data register). Software must wait for the RXNE flag to be set before clearing the PE bit. An interrupt is generated if PEIE = 1 in the USART_CR1 register. The PE bit is asserted together with all other RX-related flags (RXNE, ORE, NF, FE), so this bit can generate an interrupt request only at the end of a reception. 0 – no parity error; 1 – parity error detected.

31.5.2. **USART_DR**

Table 581 – USART_DR Register

Base ADDR=			0x500C_0000			Offset=		0x0000_0004			Reset=		0xXXXX_XXXX		
REG Name:			USART_DR												
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															

15	14	13	12	11	10	9	8…0
-							DR
							RW

Table 582 – Description of USART_DR register bits

Bit	Name	Value	Description
31…9	-	0	Reserved
8…0	DR [8:0]	X	Data value. Contains the received or transmitted data character, depending on whether the register is read or written. The data register has a dual function (read and write), since it consists of two registers: one for data transmission (TDR) and one for data reception (RDR). The TDR register provides a parallel interface between the internal bus and the transmit shift register. The RDR register provides a parallel interface between the receive shift register and the internal bus. When transmission with parity generation is enabled (PCE bit set to 1 in the USART_CR1 register), the value written into the MSB (bit 7 or bit 8, depending on data length) is ignored, because it is replaced by the parity bit. When parity checking is enabled on reception, the value read from the MSB is the received parity bit.

31.5.3. **USART_BRR**

Note: The baud-rate counters stop counting if the TE or RE bits are cleared, respectively.

Table 583 – USART_BRR Register

Base ADDR=			0x500C_0000			Offset=		0x0000_0008			Reset=		0x0000_0000		
REG Name:			USART_BRR												
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIV_Mantissa[11:0]												DIV_Fraction[3:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Table 584 – Description of the USART_BRR register bits

Bit	Name	Value	Description
31…16	-	0	Reserved
15…4	DIV_Mantissa [11:0]	0	USARTDIV mantissa. These 12 bits define the mantissa of the USART divider (USARTDIV).
3…0	DIV_Fraction [3:0]	0	USARTDIV fractional part. These 4 bits define the fractional part of the USART divider (USARTDIV). If OVER8=1, the DIV_Fraction3 bit is ignored and must be cleared.

31.5.4. **USART_CR1**

Table 585 – USART_CR1 Register

Base ADDR=			0x500C_0000			Offset=		0x0000_000C			Reset=		0x0000_0000		
REG Name:			USART_CR1												
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVER8	-	UE	M	WAKE	PCE	PS	PEIE	TXEIE	TCIE	RXNEIE	IDLEIE	TE	RE	RWU	SBK
RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Table 586 – Description of USART_CR1 register bits

Bit	Name	Value	Description
31…16	-	0	Reserved
15	OVER8	0	Oversampling mode: 0 – oversampling by 16; 1 – oversampling by 8. Note: Oversampling by 8 is not available in Smartcard mode: if SCEN = 1, OVER8 is forced to 0 by hardware.
14	-	0	Reserved
13	UE	0	USART enable. Clearing this bit to 0 immediately disables TX, RX and the CK clock generator. Any ongoing TX and RX transfers are aborted (if any). This bit is set and cleared by software. 0 – prescaler and USART outputs disabled; 1 – USART enabled.
12	M	0	Word length. This bit defines the data word length. This bit is set and cleared by software. 0 – 1 start bit, 8 data bits, n stop bits; 1 – 1 start bit, 9 data bits, n stop bits. Note: The M bit must not be modified during data transfer (both reception and transmission).
11	WAKE	0	Wakeup method. This bit selects the USART wakeup method. This bit is set and cleared by software. 0 – Idle line; 1 – Address mark.

Bit	Name	Value	Description
10	PCE	0	Parity control enable. This bit enables hardware generation and detection of the parity bit. If parity control is enabled, the computed parity value is inserted in place of the MSB (9th bit if M = 1; 8th bit if M = 0), and on reception this bit is checked for parity. This bit is set and cleared by software. Once set, PCE becomes effective after the current byte (both in reception and transmission): 0 – parity control disabled; 1 – parity control enabled.
9	PS	0	Parity selection. This bit selects whether the parity bit (with PCE set) provides even or odd parity. This bit is set and cleared by software. The even/odd parity mode becomes effective after the current byte. 0 – even parity; 1 – odd parity.
8	PEIE	0	PE interrupt enable. This bit is set and cleared by software: 0 – interrupt disabled; 1 – interrupt generated when PE = 1 in the USART_SR register
7	TXEIE	0	TXE interrupt enable. This bit is set and cleared by software: 0 – interrupt disabled; 1 – interrupt generated when TXE = 1 in the USART_SR register.
6	TCIE	0	Transmission-complete interrupt enable. This bit is set and cleared by software: 0 – interrupt disabled; 1 – interrupt generated when TC = 1 in the USART_SR register.
5	RXNEIE	0	RXNE interrupt enable. This bit is set and cleared by software: 0 – interrupt disabled; 1 – interrupt generated when ORE = 1 or RXNE = 1 in the USART_SR register.
4	IDLEIE	0	IDLE interrupt enable. This bit is set and cleared by software: 0 – interrupt disabled; 1 – interrupt generated when IDLE = 1 in the USART_SR register.

Bit	Name	Value	Description
3	TE	0	Transmitter enable. This bit enables the transmitter. This bit is set and cleared by software: 0 – transmitter disabled; 1 – transmitter enabled. Note: During transmission, a zero pulse on TE (“0” followed by “1”) inserts a preamble (idle line signal) after the current word, except in Smartcard mode. When TE is set, there is a 1 bit delay before transmission starts.
2	RE	0	Receiver enable. This bit enables the receiver. This bit is set and cleared by software: 0 – receiver disabled; 1 – receiver enabled and starts searching for a start bit.
1	RWU	0	Receiver wakeup. This bit selects whether the transceiver is in mute mode or not. This bit is set and cleared by software and can also be cleared by hardware when a wakeup sequence is recognized. 0 – receiver in active mode; 1 – receiver in mute mode. Note: Before selecting Mute mode (setting RWU), the USART must first receive one data byte; otherwise it cannot operate in mute mode with wakeup on Idle line detection. In address-mark wakeup configuration (WAKE = 1), the RWU bit cannot be modified by software while RXNE is set.
0	SBK	0	Break character transmission. This bit is used to send break characters. This bit is set and cleared by software. It must be set by software and is automatically cleared by hardware during the stop bit of the break character. 0 – no break character transmitted; 1 – break character transmitted.

31.5.5. **USART_CR2**

Table 587 – USART_CR2 Register

Base ADDR=			0x500C_0000			Offset=		0x0000_0010			Reset=		0x0000_0000		
REG Name:			USART_CR2												
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-		STOP[1:0]		CLKEN	CPOL	CPHA	LBCL	-				ADD [3:0]			
		RW	RW	RW	RW	RW	RW					RW	RW	RW	RW

Table 588 – Description of USART_CR2 register bits

Bit	Name	Value	Description
31…14	-	0	Reserved
13...12	STOP [1:0]	0	Stop bits. These bits are used to program the number of stop bits: 00 – reserved; 01 – 0.5 stop bit; 10– 2 stop bits; 11– 1.5 stop bits
11	CLKEN	0	Clock enable. This bit enables the CK output: 0 – CK pin disabled; 1 – CK pin enabled.
10	CPOL	0	Clock polarity. This bit selects the polarity of the CK output in synchronous mode. Together with the CPHA bit, this bit defines the relationship between clock and data. 0 – CK pin is always 0 when outside the frame; 1 – CK pin is always 1 when outside the frame
9	CPHA	0	Clock phase. This bit selects the phase of the CK output in synchronous mode. Together with CPOL, this bit defines the relationship between clock and data. 0 – the first clock transition is the first data capture edge; 1 – the second clock transition is the first data capture edge.

Bit	Name	Value	Description
8	LBCL	0	Last-bit clock pulse. This bit selects whether the clock pulse corresponding to the last transmitted data bit (MSB) is output on CK in synchronous mode. 9 – no clock pulse for the last data bit on CK; 1 – clock pulse for the last data bit is output on CK. Note: The last bit is the 8th or 9th data bit, depending on the format selected by the M bit in the USART_CR1 register.
7…4	-	0	Reserved
3…0	ADD [3:0]	0	USART node address. This bitfield specifies the USART address. These bits are used in multiprocessor communication in mute mode to wake up the transceiver when an address mark is detected. Note: The CPOL, CPHA and LBCL bits must not be written while the transmitter is enabled.

31.5.6. **USART_CR3**

Table 589 – USART_CR3 Register

Base ADDR=			0x500C_0000			Offset=		0x0000_0014			Reset=		0x0000_0000		
REG Name:			USART_CR3												
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-				ONEBIT	CTSIE	CTSE	RTSE	DMAT	DMAR	SCEN	NACK	HDSEL	-		EIE
				RW	RW	RW	RW	RW	RW	RW	RW	RW			RW

Table 590 – Description of USART_CR3 register bits

Bit	Name	Value	Description
31…12	-	0	Reserved
11	ONEBIT	0	One-sample-per-bit mode enable. This bit selects the signal digitizing method: 0 – three-sample method; 1 – single-sample method. When the single-sample method is selected, the noise detection flag (NF) is disabled. Note: The ONEBIT function applies only to data bits. It does not apply to the start bit.
10	CTSIE	0	CTS interrupt enable: 0 – interrupt disabled; 1 – an interrupt is generated each time CTS = 1 in the USART_SR register.

Bit	Name	Value	Description
9	CTSE	0	CTS enable: 0 – hardware CTS flow control disabled; 1 – CTS mode enabled; data is transmitted only when the CTS input is at logic 0. If the CTS input is at logic 1 during data transmission, the current transmission is completed before the transmitter stops. If data is written to the data register while CTS is at logic 1, transmission is deferred until the CTS input goes to logic 0.
8	RTSE	0	RTS enable: 0 – hardware RTS flow control disabled; 1 – RTS flow control enabled; data is requested when there is space in the receive buffer. It is expected that data transmission will stop after the last current data character has been transmitted. The RTS output is at logic 0 when the transceiver is able to receive data.
7	DMAT	0	DMA transmitter enable: This bit is set and cleared by software. 1 – DMA mode enabled for transmission; 0 – DMA mode disabled for transmission.
6	DMAR	0	DMA receiver enable. This bit is set and cleared by software: 1 – DMA mode enabled for reception; 0 – DMA mode disabled for reception.
5	SCEN	0	Smartcard mode enable. This bit is used to enable Smartcard mode. 0 – Smartcard mode disabled; 1 – Smartcard mode enabled.
4	NACK	0	NACK enable in Smartcard mode: 0 – NACK response on parity error disabled; 1 – NACK response on parity error enabled.
3	HDSEL	0	Half-duplex mode selection. This bit selects single-wire half-duplex mode: 0 – half-duplex mode not selected; 1 – half-duplex mode selected.
2, 1	-	0	Reserved
0	EIE	0	Error interrupt enable. This bit enables generation of an interrupt on framing error, overrun error, or noise flag (FE = 1 or ORE = 1 or NF = 1 in the USART_SR register) in the case of multibuffer communication (DMAR = 1 in the USART_CR3 register). 0 – interrupt disabled; 1 – an interrupt is generated each time DMAR = 1 in the USART_CR3 register and FE = 1 or ORE = 1 or NF = 1 in the USART_SR register.

31.5.7. **USART_GTPR**

Table 591 – USART_GTPR Register

Base ADDR=			0x500C_0000			Offset=		0x0000_0018			Reset=		0x0000_0000		
REG Name:			USART_GTPR												
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GT [7:0]								-			PSC [4:0]				
RW	RW	RW	RW	RW	RW	RW	RW				RW	RW	RW	RW	RW

Table 592 – Description of USART_GTPR register bits

Bit	Name	Value	Description
31…16	-	0	Reserved
15…8	GT[7:0]	0	Guard time value. This bitfield holds the guard time value in units of bit periods. The bits are used in Smartcard mode. The transmission-complete flag is set after this guard time value.
4…0	PSC[4:0]	0	Smartcard prescaler value. Used to program the prescaler that divides the system clock to provide the Smartcard clock. The value written to the register (5 significant bits) is multiplied by 2 to obtain the division factor of the original clock: 00000: reserved – do not program this value; 00001: divides the clock source by 2; 00010: divides the clock source by 4; 00011: divides the clock source by 6.

32 Electrical parameters of the integrated circuits

Table 593 – Electrical parameters of the integrated circuits at acceptance and delivery

Parameter name, unit of measure, measurement conditions	Parameter symbol	Parameter specification		Ambient temperature, °C
		minimum	maximum	
High-level output voltage, V: – for IRL1206FI at port outputs PA, PB, PC, PD	U_{OH}	U_{CC} – 0.6	–	25, 85, – 50[1)]
Low-level output voltage, V, – for IRL1206FI: a) at port outputs PA, PB, PC, PD at: – $U_{CC} \geq 3.0$ V;	U_{OL}	–	0.4	25, 85, – 50[1)]
– $U_{CC} < 3.0$ V;		–	0.5	
b) at SW output at $U_{CCA} = 1.8$ V, − $U_{CC} = 1.8$ V, $I_{OL} = 1.0$ mA		–	0.4	
Low-level output current in the Off state, µA, for IRL1206FI at SW output at $U_{OZ} = 0$ V.	I_{OZL}	– 1.0	1.0	25, 85, – 50
High-level output current in the Off state, µA, for IRL1206FI at SW output at $U_{OZ} = 0$ V.	I_{OZH}	– 1.0	1.0	
Low-level input leakage current, µA: – for IRL1206FI at inputs of ports PA, PB, PC, PD, nRESET, WAKEUP, JTAG_EN, OSC_IN32, SAR_AIN.	I_{ILL}	– 1.0	1.0	25, 85, – 50[1)]
High-level input leakage current, µA: – for IRL1206FI: a) at inputs of ports PA, PB, PC, PD, nRESET, WAKEUP, OSC_IN32, SAR_AIN;	I_{ILH}	– 1.0	1.0	25, 85, – 50[1)]
b) at input JTAG_EN;		10	80	
c) at input OSC_IN.		1	20	
Dynamic supply current from UCC, mA, – for IRL1206FI ICs at f_C [1)] ≤ 50 MHz.	I_{CCO_UCC}	–	30	25, 85, – 50[1)]
Dynamic supply current from AUcc, mA, – for IRL1206FI ICs at $f_C \leq 50$ MHz.	I_{CCO_AUCC}	–	7	25, 85, – 50
Dynamic supply current from BUcc, µA, – for IRL1206FI ICs at $f_C \leq 50$ MHz.	I_{CCO_BUCC}	–	10	25, 85, – 50
HSI RC oscillator output frequency, MHz, after trimming	f_{O_HSI}	7.5	8.5	25, 85, – 50[1)]
LSI RC oscillator output frequency, kHz, after trimming.	f_{O_LSI}	29.3	33.2	
PLL output frequency, MHz: – maximum;	f_{O_PLL}	50	–	
– minimum		–	8	
ΔΣ ADC parameters for IRL1206FI				

Parameter name, unit of measure, measurement conditions	Parameter symbol	Parameter specification		Ambient temperature, °C
		minimum	maximum	
Signal-to-noise ratio, dB, gain +6 dB, f_{NADCO} [2] = 4 kHz	SNR_{D6}	82	–	25, 85, – 50

[1] f_C is the core clock frequency.

[2] f_{NADCO} is the oversampling frequency at the IC output.

The ICs are resistant to electrostatic discharge with a potential of at least 2000 V.

33 Maximum allowable parameters and limits

Table 594 – Maximum allowable electrical operating conditions and electrical condition limits of the IC

Parameter name, unit of measure, measurement conditions	Parameter symbol	Parameter specification			
		Maximum allowable operating condition		Limit condition	
		minimum	maximum	minimum	maximum
Supply voltage, V, at V_{CC} pins	U_{CC}	1.8	3.6	–	3.9
Analog supply voltage, V, for IRL1206FI at V_{CCA} pins	U_{CCA}	1.8	3.6	–	3.9
− when using the ADC.		3.0	3.6	–	3,9
Battery-domain supply voltage, V, for IRL1206FI at V_{CCB} pin	U_{CCB}	1.8	3.6	–	3.9
High-level input voltage, V, − for IRL1206FI: at JTAG_EN input and at port inputs PA, PB, PC, PD when: 1) $U_{CC} \geq 3.0$ V;	U_{IH}	2.0	U_{CC}	–	U_{CC} + 0.3
2) $U_{CC} < 3.0$ V;		0.65·U_{CC}			
Low-level input voltage, V, − for IRL1206FI: at JTAG_EN input and at port inputs PA, PB, PC, PD when: 1) $U_{CC} \geq 3.0$ V;	U_{IL}	0	0.8	– 0.3	–
2) $U_{CC} < 3.0$ V;			0.6		
High-level input voltage, V, for IRL1206FI at OSC_IN input when HSE_BYP = 1	$U_{IH\_OSC\_IN}$	U_{CC} – 0.6	U_{CC}	–	U_{CC} + 0.3
Low-level input voltage, V, for IRL1206FI at OSC_IN input when HSE_BYP = 1	$U_{IL\_OSC\_IN}$	0	0.4		–
High-level input voltage, V, for IRL1206FI at OSC_IN32 input when LSE_BYP = 1.	$U_{IH\_OSC\_IN32}$	U_{CCB} – 0.6	U_{CCB}		U_{CCB} + 0.3
Low-level input voltage, V, for IRL1206FI at OSC_IN32 input when LSE_BYP = 1.	$U_{IL\_OSC\_IN32}$	0	0.4	– 0.3	–
High-level input voltage, V, for IRL1206FI at WAKEUP1, WAKEUP2, WAKEUP3 inputs when: a) $U_{CCB} \geq 3.0$ V	U_{IH_WAKEUP}	2.0	U_{CCB}	–	U_{CCB} + 0.3
b) $U_{CCB} < 3.0$ V		0.65·U_{CCB}			
Low-level input voltage, V, for IRL1206FI at WAKEUP1, WAKEUP2, WAKEUP3 inputs when:	U_{IL_WAKEUP}	0		– 0.3	–
a) $U_{CCB} \geq 3.0$ V			0.8		
b) $U_{CCB} < 3.0$ V			0.5		
Output voltage in Off state, V, for IRL1206FI at SW output.	U_{OZ}	0	U_{CC}	–	–
High-level output current, mA: − for IRL1206FI at port outputs PA, PB, PC, PD.	I_{OH}	– 4.0	–	– 6.0	–

Parameter name, unit of measure, measurement conditions	Parameter symbol	Parameter specification			
		Maximum allowable operating condition		Limit condition	
		minimum	maximum	minimum	maximum
Low-level output current, mA, − for IRL1206FI: a) at port outputs PA, PB, PC, PD;	I_{OL}	–	4.0	–	6.0
b) at SW output;		–	1.0	–	2,0
High-level output current at OSC_OUT pin, mA.	$I_{OH\_OSC\_OUT}$	– 0.3	–	– 1	–
Low-level output current at OSC_OUT pin, mA.	$I_{OL\_OSC\_OUT}$	–	0.3	–	1
PLL clock signal frequency, MHz.	f_{C_PLL}	8	16	–	–
HSE clock signal frequency, MHz, for IRL1206FI at OSC_IN input when: − BYPASS = “0”;	f_{C_HSE}	8	16	–	–
− BYPASS = “1” (CHIP_ID = 217)		–	50	–	–
− BYPASS = “1” (CHIP_ID = 215)		–	42	–	–
LSE clock signal frequency, kHz, for IRL1206FI at OSC_IN32 input when: − BYPASS = “0”;	f_{C_LSE}	32	33	–	–
− BYPASS = “1”		–	1000	–	–
Load capacitance, pF, − for IRL1206FI at port outputs PA, PB, PC, PD.	C_L	–	30	–	–
ΔΣ ADC parameters for IRL1206FI.					
Reference voltage, V, at VREF pin with the internal reference source disabled.	U_{REF_ADC}	2.33	2.53	–	–
Differential input voltage, V, at inputs I0P – I3P, I0N – I3N, V0P – V2P, V0N – V2N.	U_{ID_ADCD}	–1.0	1.0	–	–
Voltage, V, at inputs I0P – I3P, I0N – I3N, V0P – V2P, V0N – V2N.	U_{I_ADCD}	–0.5	0.5	–0.8	U_{CC} + 0.3
ΔΣ ADC clock frequency, MHz.	f_{C_ADCD}	0.9	4.1	–	–
SAR ADC parameters for IRL1206FI.					
SAR ADC input voltage, V, at inputs SAR_AIN1, SAR_AIN2, SAR_AIN3 when: − using the internal reference voltage source; − applying a reference voltage to VREF pin;	U_{AIN}	0	U_{REF_ADC}	–0.3	3.9
− operating without a reference voltage.		0	U_{CCA}		
SAR ADC clock frequency, MHz.	f_{C_ADCS}	–	28	–	–
Note: Two limiting conditions must not be applied simultaneously.					

34 Reference parameters

Table 595 – Reference parameters of the integrated circuits

Parameter name, unit of measure, measurement conditions	Parameter symbol	Parameter specification			Ambient temperature, °C
		minimum	typical	maximum	
IRL1206FI					
Static supply current, mA	I_{CC}	–	0,6	–	25, 85, – 50
Flash memory data retention time, years, at: – T = 25 °C	t_{SG}	25	–	–	
– T = 85 °C		10	–	–	
Microcontroller core clock frequency, MHz IRL1206FI	f_C	–	–	50	
Resistance of internal programmable pull-down resistors to ground rail, kΩ	R_{PD}		76		
Resistance of internal programmable pull-up resistors to the supply rail, kΩ	R_{PU}		76		
SAR ADC parameters					
ADC resolution, bits	E_{NADC}	10	–	–	25, 85, – 50
Differential nonlinearity, LSV [1)] – maximum	E_{DLADC}	–	0.26	1.00	
– minimum		– 1.00	– 0.6	–	
Integral nonlinearity, LSV [1)] – maximum	E_{ILADC}	–	0.4	1.0	
– minimum		– 1.0	– 0.4	–	
Offset error, LSV [1)]	E_{OFFADC}	– 1.00	– 0.25	1.00	
Gain error, %	$E_{GAINADC}$	– 0.10	– 0.05	0.10	
IRL1206FI					
Battery-domain reset supply voltage U_{CCB}, V	U_{CCB_RES}	–	0.7	–	25
Static supply current of the analog supply, mA	I_{CCA}	–	0.13	–	25, 85, – 50
Static supply current of the battery domain, µA	I_{CCB}	–	0.5	–	
SPI parameters					
SSP_CLK clock frequency, MHz, – in master mode; – in slave mode	f_{OP}	– –	– –	$0,5 \cdot f_C$ $0,25 \cdot f_C$	25, 85, – 50

Parameter name, unit of measure,	Parameter	Parameter specification			Ambient
High-/low-level duration of the SSP_CLK clock signal, ns	t_{WSSP_CLK}	$\frac{1}{2 \cdot f_{OP}}$	–	–	25, 85, – 50
Data propagation delay when SSP_TXD output switches from Off to high/low level on SSP_FSS signal, ns	$t_{PZH(SSP\_CLK-SSP\_TXD)}$ $t_{PZL(SSP\_CLK-SSP\_TXD)}$	$\frac{1}{2 \cdot f_{OP}}$	–	–	
Data propagation delay when SSP_TXD output switches from high/low level to Off on SSP_FSS signal, ns	$t_{PHZ(SSP\_CLK-SSP\_TXD)}$ $t_{PLZ(SSP\_CLK-SSP\_TXD)}$	$\frac{1}{2 \cdot f_{OP}}$	–	–	
SSP_RXD input data setup time relative to SSP_CLK, ns	$t_{SU(SSP\_RXD–SSP\_CLK)}$	2,5	–	–	
SSP_RXD input data hold time relative to SSP_CLK, ns	$t_{H(SSP\_CLK–SSP\_RXD)}$	3,5	–	–	
SSP_TXD output data hold time relative to SSP_CLK, ns	$t_{V(SSP\_CLK-SSP\_TXD)}$	0	–	–	
∆∑ ADC parameters					
Internal reference voltage, V, at pin VREF	U_{REFO}	–	2,45	–	25, 85, – 50
Signal-to-noise ratio, dB, gain + 0 dB, U_{ID_ADCD} = +/- 1.0 V, f_{NADCO} = 4 kHz	SNR_{D0}	–	87	–	
Signal-to-noise ratio, dB, gain + 6 dB, U_{ID_ADCD} = +/- 1.0 V, f_{NADCO} = 4 kHz	SNR_{D6}	–	91	–	
Signal-to-noise ratio, dB, gain + 12 dB, U_{ID_ADCD} = +/- 0.5 V, f_{NADCO} = 4 kHz	SNR_{D12}	–	88	–	
Signal-to-noise ratio, dB, gain + 18 dB, U_{ID_ADCD} = +/- 0.25 V, f_{NADCO} = 4 kHz	SNR_{D18}	–	85	–	
Signal-to-noise ratio, dB, gain + 24 dB, U_{ID_ADCD} = +/- 0.125 V, f_{NADCO} = 4 kHz	SNR_{D24}	–	82	–	
Preamplifier gain error, dB	$GAIN_{ERR}$	– 2.0	– 0.5	2.0	

[1)] LSB, least significant value – the value of differential voltage which, for a given reference voltage (U_{REF_ADC}), corresponds to the change of ADC input voltage required to change the output code by one least significant bit.

$1\ LSV = \frac{U_{REF\_ADC}}{2^{10}}$ for a 10-bit ideal ADC

34.1. Dynamic characteristics of the ΔΣ ADC

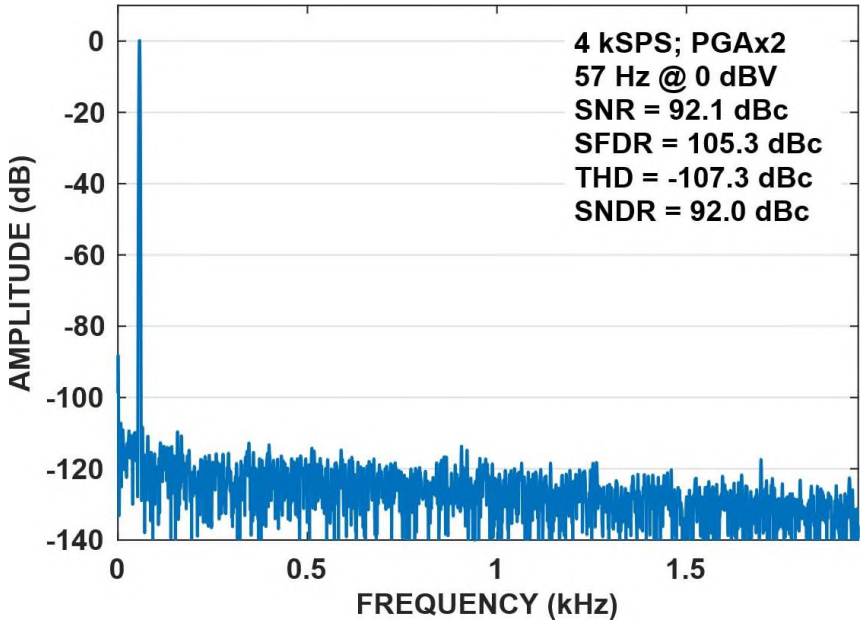


Figure 131 – Typical spectrum under nominal conditions for the ΔΣ ADC of the IRL1206FI (CHIP_ID = 217) on channel F2IDAT for a differential input signal of ±1 V (0.707 RMS). Gain = 6 dB (F2IGAIN = 00).

34.2. Performance in tests

Table 596 – CoreMark performance

IC	Execution of code from FLASH		Execution of code from TCM
	Delay[2:0] = 0b001	Delay[2:0] = 0b000	
IRL1206FI Coremark/MHz	2.15	3.4	3.7

34.3. **IC pin characteristics**

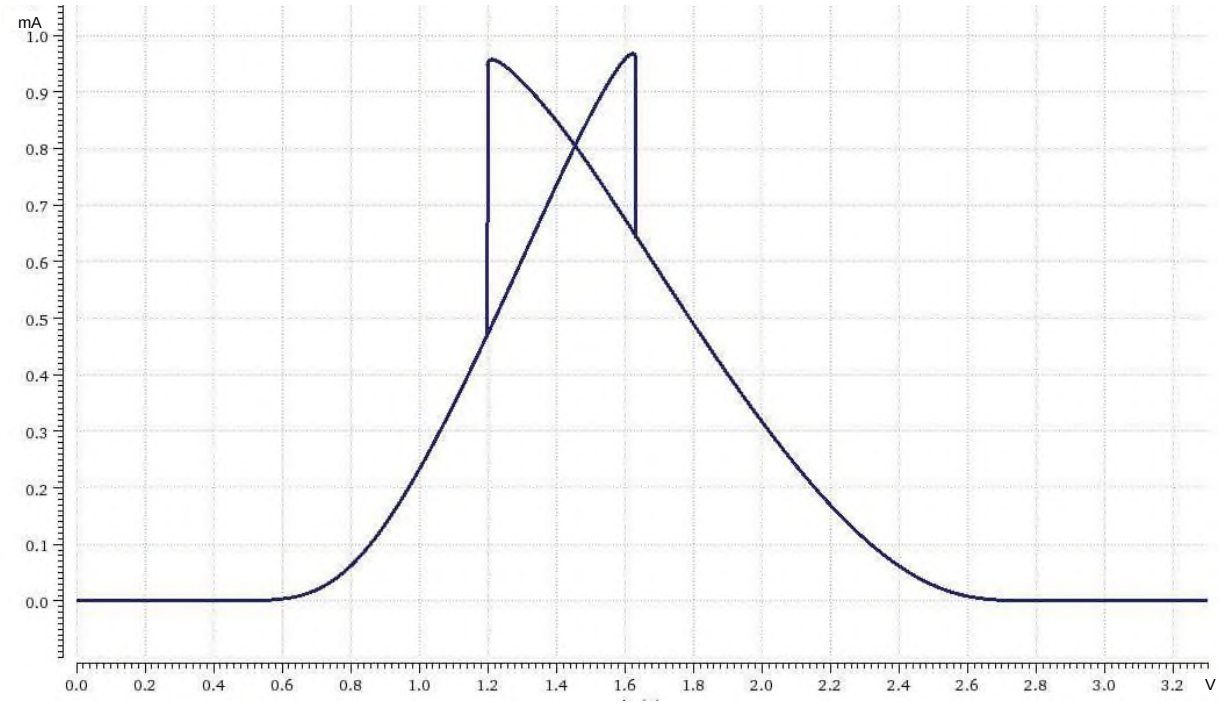


Figure 132 – Typical dependence of additional U_{CC} supply current on digital input voltage at U_{CC} = 3.3 V under nominal conditions

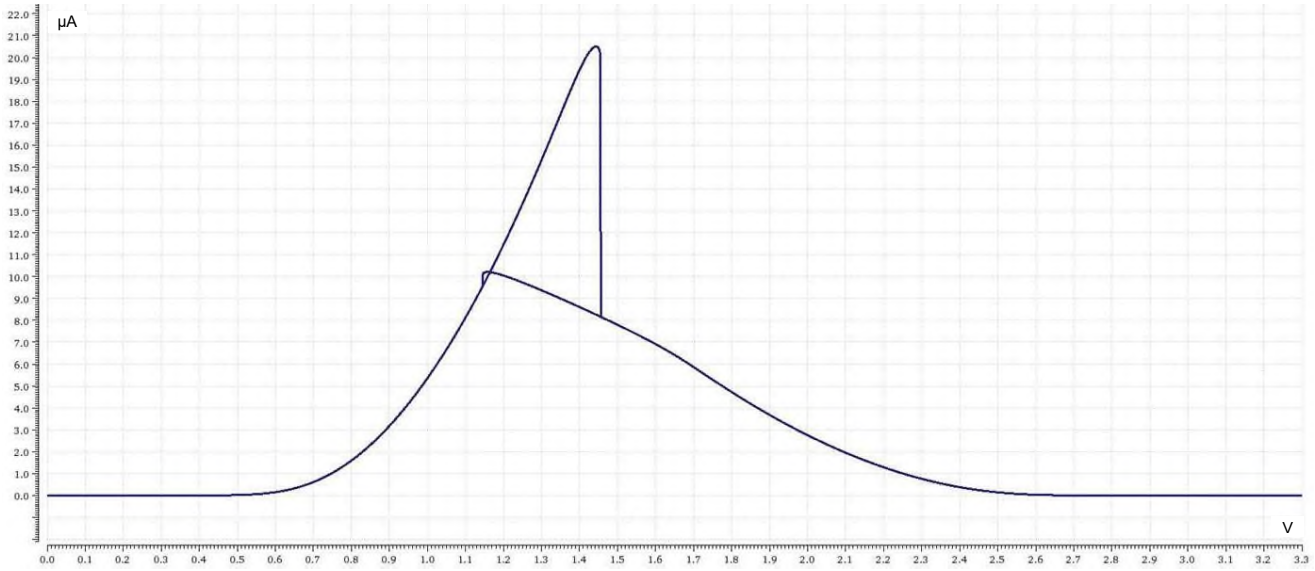


Figure 133 – Typical dependence of additional U_{CCB} supply current on WAKEUP1-WAKEUP3 input voltage at U_{CCB} = 3.3 V under nominal conditions

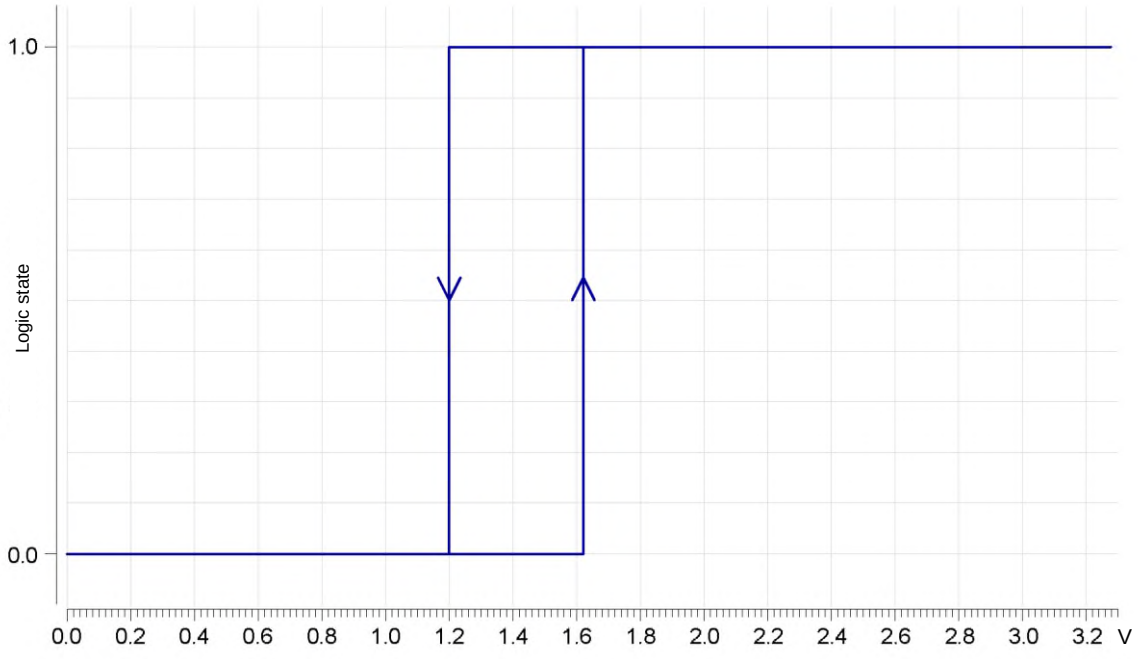


Figure 134 – Typical dependence of logic state on digital input voltage at U_{CC} = 3.3 V under nominal conditions

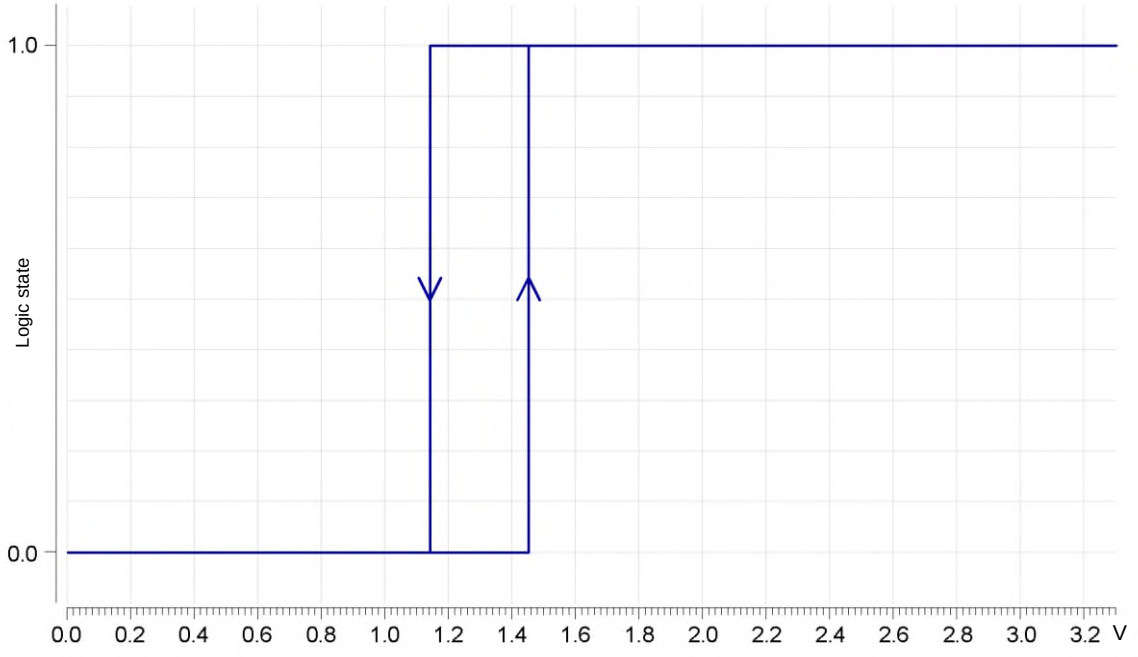


Figure 135 – Typical dependence of logic state on WAKEUP1-WAKEUP3 input voltage at U_{CCB} = 3.3 V under nominal conditions

35 Typical dependencies

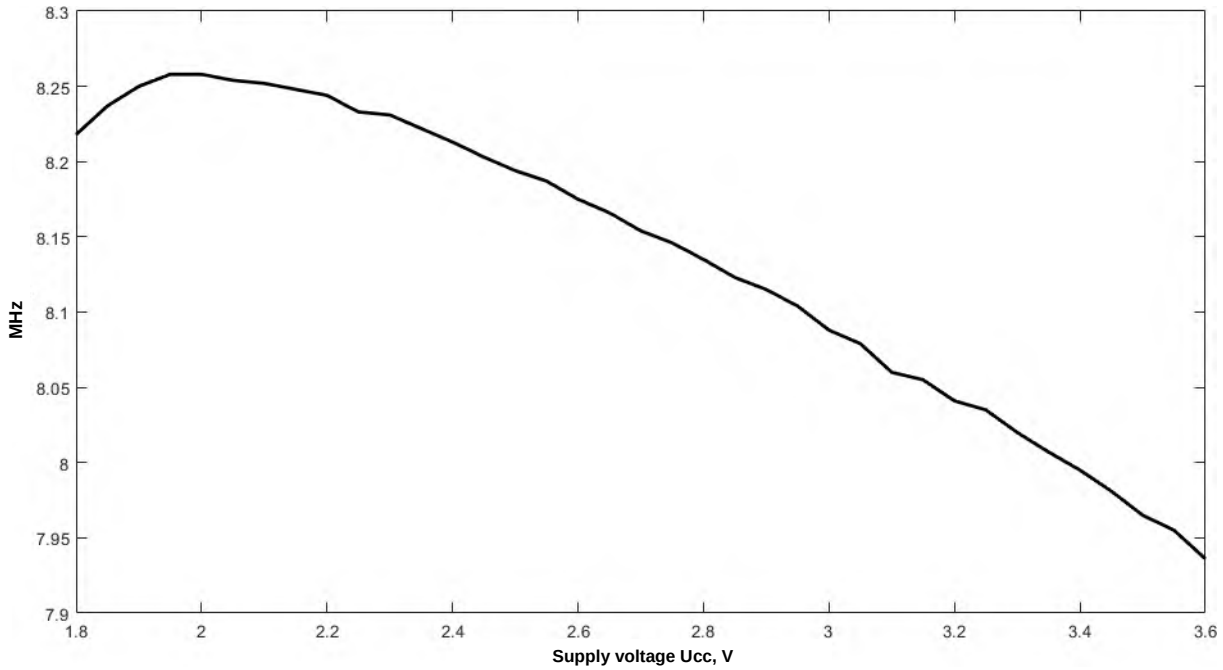


Figure 136 – Dependence of HSI RC oscillator frequency of IRL1206FI (CHIP_ID = 217) on supply voltage U_{CC} under nominal conditions

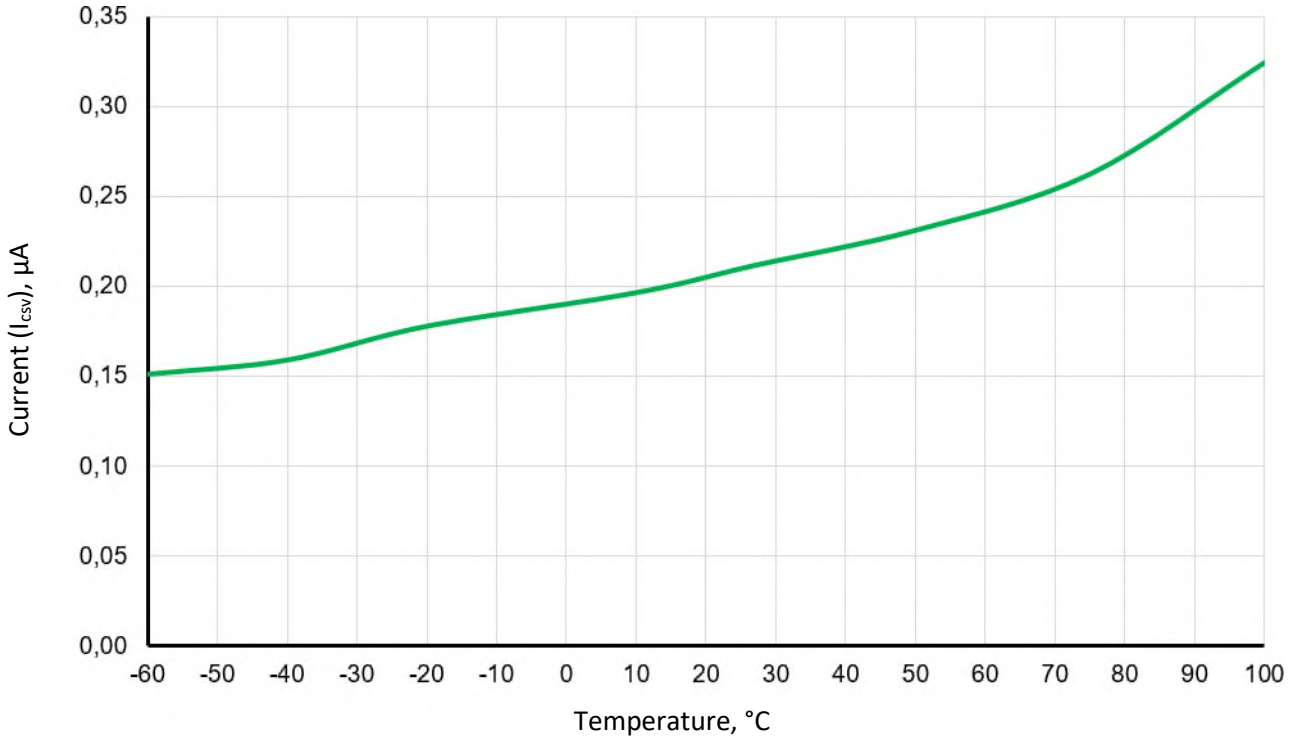


Figure 137 – Dependence of static battery-domain current I_{CCB} of IRL1206FI (CHIP_ID = 217) on temperature at U_{CCB} = 3.3 V

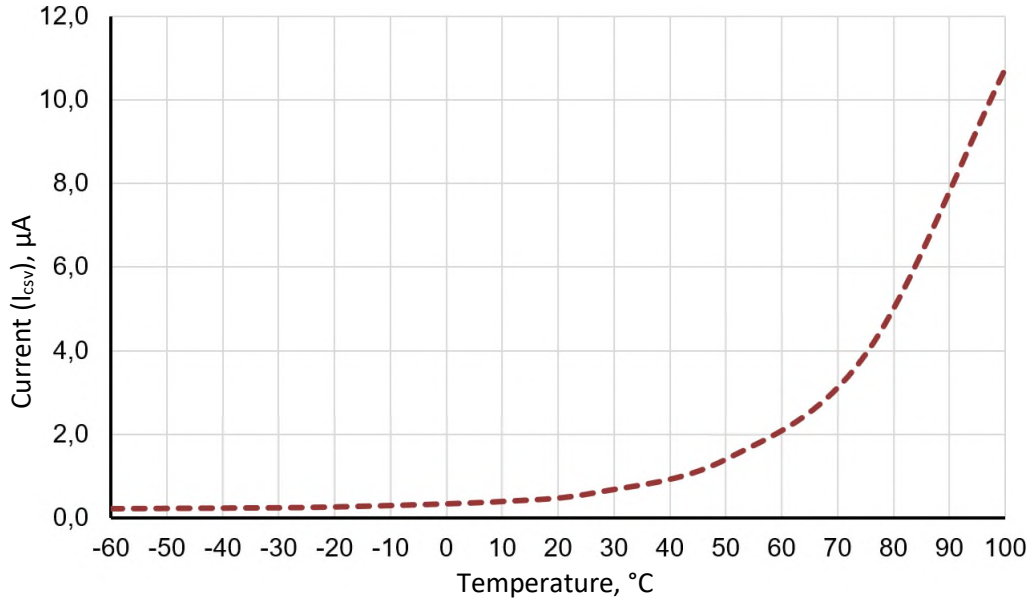


Figure 138 – Dependence of static battery-domain current I_{CCB} of IRL1206FI (CHIP_ID = 217) on temperature at U_{CCB} = 3.3 V with main supply absent

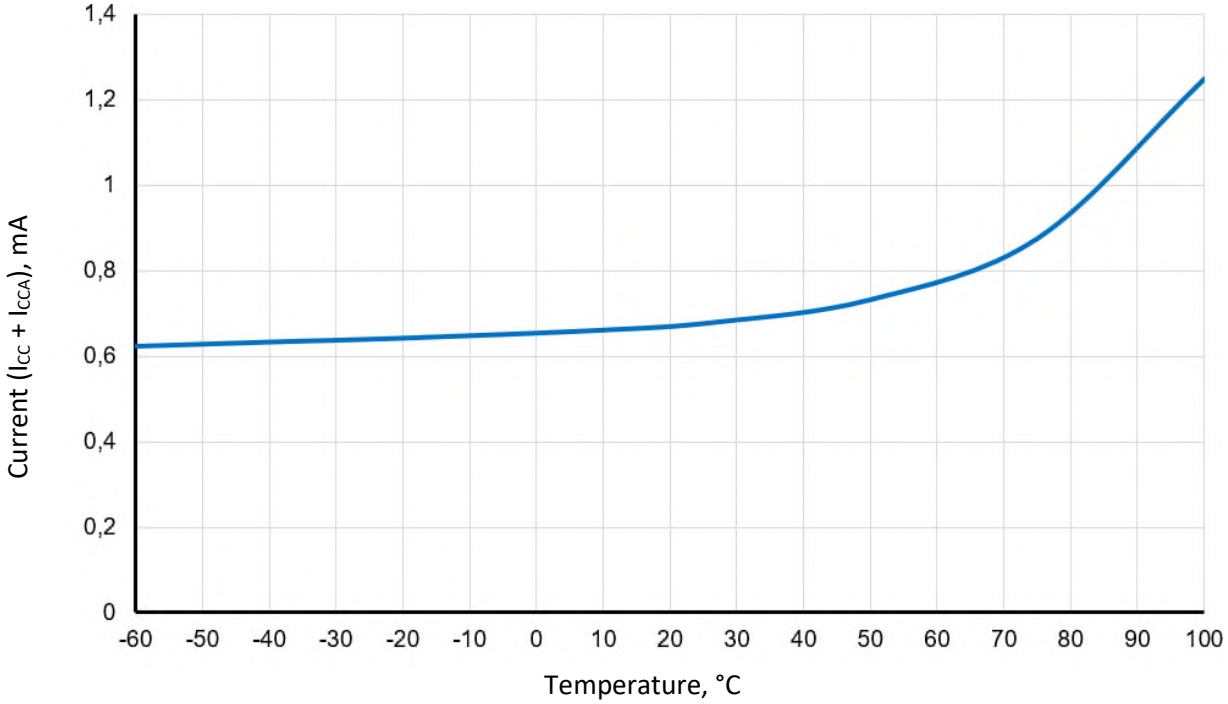


Figure 139 – Dependence of total static current (nRESET = 0) I_{CC} and I_{CCA} of IRL1206FI (CHIP_ID = 217) on temperature at U_{CC} = U_{CCA} = 3.3 V

36 Integrated circuit outline

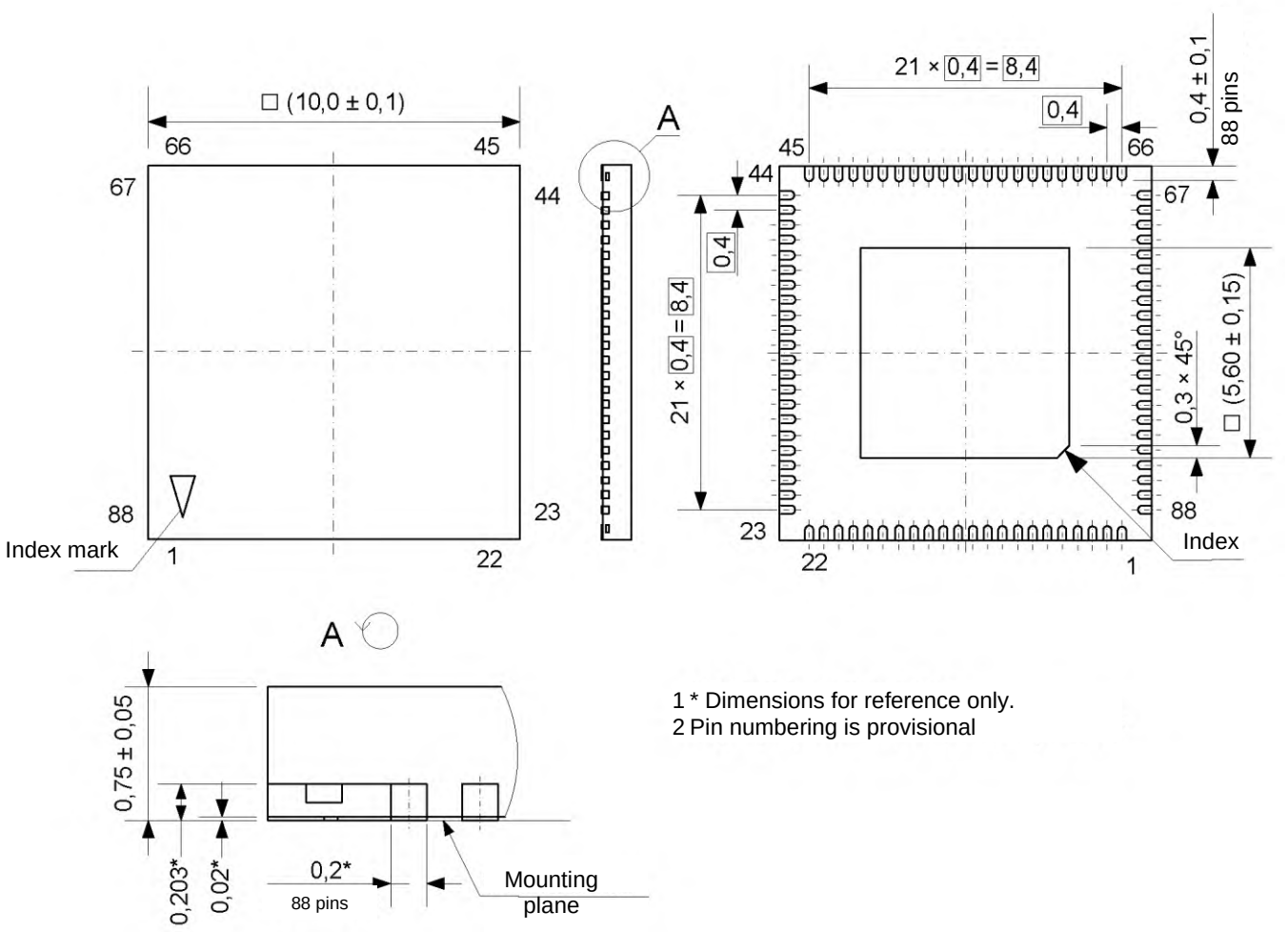


Figure 140 – IRL1206FI in QFN88 10×10×0.75 (0.4) package

37 Ordering information

Designation	Marking	Package type	Temperature range, °C
IRL1206FI	1206	QFN88 10×10×0.75 (0.4)	-50 to +85