



BL702/704/706

Reference Manual

Version: 1.2

Copyright @ 2023

www.bouffalolab.com

Contents

1	System and memory overview	19
1.1	Introduction	19
1.2	Main features	19
1.3	Function description	19
2	Reset and clock	23
2.1	Introduction	23
2.2	Reset source	23
2.3	Clock source	24
3	GLB	26
3.1	Introduction	26
3.2	Function description	26
3.2.1	Clock management	26
3.2.2	Reset management	26
3.2.3	Bus management	27
3.2.4	Memory management	28
3.2.5	GPIO overview	28
3.2.6	GPIO main features	28
3.2.7	GPIO function description	28
3.2.8	GPIO function setting	29
3.2.9	GPIO output settings	33
3.2.10	GPIO input settings	33
3.2.11	GPIO optional function settings	33
3.2.12	GPIO interrupt settings	34
3.3	Register description	34
3.3.1	GPIO_CFGCTL0	35
3.3.2	GPIO_CFGCTL1	37

3.3.3	GPIO_CFGCTL2	38
3.3.4	GPIO_CFGCTL3	39
3.3.5	GPIO_CFGCTL4	40
3.3.6	GPIO_CFGCTL5	41
3.3.7	GPIO_CFGCTL6	42
3.3.8	GPIO_CFGCTL7	43
3.3.9	GPIO_CFGCTL8	44
3.3.10	GPIO_CFGCTL9	45
3.3.11	GPIO_CFGCTL10	46
3.3.12	GPIO_CFGCTL11	47
3.3.13	GPIO_CFGCTL12	48
3.3.14	GPIO_CFGCTL13	49
3.3.15	GPIO_CFGCTL14	50
3.3.16	GPIO_CFGCTL15	51
3.3.17	GPIO_CFGCTL30	53
3.3.18	GPIO_CFGCTL32	54
3.3.19	GPIO_CFGCTL34	56
3.3.20	GPIO_CFGCTL35	58
3.3.21	GPIO_INT_STAT1	58
3.3.22	GPIO_INT_CLR1	59
3.3.23	GPIO_INT_MODE_SET1	59
3.3.24	GPIO_INT_MODE_SET2	59
3.3.25	GPIO_INT_MODE_SET3	60
3.3.26	GPIO_INT_MODE_SET4	60
3.3.27	GPIO_INT2_MASK1	61
3.3.28	GPIO_INT2_STAT1	61
3.3.29	GPIO_INT2_CLR1	61
3.3.30	GPIO_INT2_MODE_SET1	62
3.3.31	GPIO_INT2_MODE_SET2	62
3.3.32	GPIO_INT2_MODE_SET3	63
3.3.33	GPIO_INT2_MODE_SET4	63
4	ADC	64
4.1	ADC introduction	64
4.2	ADC main features	64
4.3	ADC functional description	65
4.3.1	ADC pins and internal signals	66
4.3.2	ADC channel	66
4.3.3	ADC clock	67
4.3.4	ADC conversion mode	68

4.3.5	ADC consequence	69
4.3.6	ADC interrupt	70
4.3.7	ADC FIFO	70
4.3.8	ADC configuration process	71
4.3.9	VBAT measurement	72
4.3.10	TSEN measurement	72
4.4	Register description	73
4.4.1	gpadc_config	73
4.4.2	gpadc_dma_rdata	75
4.4.3	gpdac_config	75
4.4.4	gpdac_dma_config	76
4.4.5	gpdac_dma_wdata	77
4.4.6	gpadc_reg_cmd	77
4.4.7	gpadc_reg_config1	80
4.4.8	gpadc_reg_config2	83
4.4.9	gpadc_reg_scn_pos1	85
4.4.10	gpadc_reg_scn_pos2	85
4.4.11	gpadc_reg_scn_neg1	86
4.4.12	gpadc_reg_scn_neg2	87
4.4.13	gpadc_reg_status	88
4.4.14	gpadc_reg_isr	88
4.4.15	gpadc_reg_raw_result	89
4.4.16	gpadc_reg_define	89
5	DAC	90
5.1	DAC introduction	90
5.2	DAC main feature	90
5.3	DAC function description	90
5.4	Register description	92
5.4.1	gpadc_config	93
5.4.2	gpadc_dma_rdata	94
5.4.3	gpdac_config	94
5.4.4	gpdac_dma_config	95
5.4.5	gpdac_dma_wdata	96
6	DMA	97
6.1	DMA Introduction	97
6.2	DMA main features	97
6.3	DMA functional description	98
6.3.1	Working principle	98
6.3.2	DMA channel configuration	99

6.3.3	Peripheral support	99
6.3.4	Linked List Mode	100
6.3.5	DMA interrupt	101
6.4	Transmission mode	101
6.4.1	Memory to memory	101
6.4.2	Memory to peripheral	101
6.4.3	Peripheral to memory	102
6.4.4	Peripheral to peripheral	102
6.5	Register description	103
6.5.1	DMA_IntStatus	105
6.5.2	DMA_IntTCStatus	105
6.5.3	DMA_IntTCClear	106
6.5.4	DMA_IntErrorStatus	106
6.5.5	DMA_IntErrClr	106
6.5.6	DMA_RawIntTCStatus	107
6.5.7	DMA_RawIntErrorStatus	107
6.5.8	DMA_EnbldChns	108
6.5.9	DMA_SoftBReq	108
6.5.10	DMA_SoftSReq	109
6.5.11	DMA_SoftLBReq	109
6.5.12	DMA_SoftLSReq	109
6.5.13	DMA_Config	110
6.5.14	DMA_Sync	110
6.5.15	DMA_C0SrcAddr	110
6.5.16	DMA_C0DstAddr	111
6.5.17	DMA_C0LLI	111
6.5.18	DMA_C0Control	111
6.5.19	DMA_C0Config	113
6.5.20	DMA_C1SrcAddr	114
6.5.21	DMA_C1DstAddr	114
6.5.22	DMA_C1LLI	115
6.5.23	DMA_C1Control	115
6.5.24	DMA_C1Config	116
6.5.25	DMA_C2SrcAddr	117
6.5.26	DMA_C2DstAddr	117
6.5.27	DMA_C2LLI	117
6.5.28	DMA_C2Control	118
6.5.29	DMA_C2Config	119
6.5.30	DMA_C3SrcAddr	119

6.5.31	DMA_C3DstAddr	120
6.5.32	DMA_C3LLI	120
6.5.33	DMA_C3Control	120
6.5.34	DMA_C3Config	121
6.5.35	DMA_C4SrcAddr	122
6.5.36	DMA_C4DstAddr	122
6.5.37	DMA_C4LLI	123
6.5.38	DMA_C4Control	123
6.5.39	DMA_C4Config	124
6.5.40	DMA_C5SrcAddr	125
6.5.41	DMA_C5DstAddr	125
6.5.42	DMA_C5LLI	125
6.5.43	DMA_C5Control	126
6.5.44	DMA_C5Config	127
6.5.45	DMA_C6SrcAddr	127
6.5.46	DMA_C6DstAddr	128
6.5.47	DMA_C6LLI	128
6.5.48	DMA_C6Control	128
6.5.49	DMA_C6Config	129
6.5.50	DMA_C7SrcAddr	130
6.5.51	DMA_C7DstAddr	130
6.5.52	DMA_C7LLI	131
6.5.53	DMA_C7Control	131
6.5.54	DMA_C7Config	132
7	L1C	133
7.1	L1C introduction	133
7.2	L1C main features	134
7.3	L1C function description	134
7.3.1	Mutual conversion between TCM and Cache RAM resources	134
7.3.2	Cache	134
7.4	Register description	135
7.4.1	l1c_config	136
7.4.2	hit_cnt_lsb	137
7.4.3	hit_cnt_msb	137
7.4.4	miss_cnt	137
8	IR	138
8.1	IR introduction	138
8.2	IR main features	138

8.3	IR function description	138
8.3.1	Fixed receiving protocol	138
8.3.2	Pulse width reception	140
8.3.3	Normal sending mode	140
8.3.4	Pulse width transmission	140
8.3.5	Carrier modulation	141
8.3.6	IR interrupt	141
8.4	Register description	141
8.4.1	irtx_config	142
8.4.2	irtx_int_sts	143
8.4.3	irtx_data_word0	144
8.4.4	irtx_data_word1	144
8.4.5	irtx_pulse_width	145
8.4.6	irtx_pw	145
8.4.7	irtx_swm_pw_0	146
8.4.8	irtx_swm_pw_1	146
8.4.9	irtx_swm_pw_2	147
8.4.10	irtx_swm_pw_3	147
8.4.11	irtx_swm_pw_4	148
8.4.12	irtx_swm_pw_5	148
8.4.13	irtx_swm_pw_6	148
8.4.14	irtx_swm_pw_7	149
8.4.15	irrx_config	149
8.4.16	irrx_int_sts	150
8.4.17	irrx_pw_config	151
8.4.18	irrx_data_count	151
8.4.19	irrx_data_word0	152
8.4.20	irrx_data_word1	152
8.4.21	irrx_swm_fifo_config_0	152
8.4.22	irrx_swm_fifo_rdata	153
9	SPI	154
9.1	SPI introduction	154
9.2	SPI main features	154
9.3	SPI function description	155
9.3.1	Clock control	155
9.3.2	Master continuous transmission mode	155
9.3.3	Acceptance filtering function	156
9.3.4	Receive error correction	156
9.3.5	Slave mode timeout mechanism	156

9.3.6	I/O transfer mode	156
9.3.7	DMA transfer mode	157
9.3.8	SPI interrupt	157
9.4	Register description	157
9.4.1	spi_config	158
9.4.2	spi_int_sts	159
9.4.3	spi_bus_busy	161
9.4.4	spi_prd_0	161
9.4.5	spi_prd_1	162
9.4.6	spi_rxd_ignr	162
9.4.7	spi_sto_value	163
9.4.8	spi_fifo_config_0	163
9.4.9	spi_fifo_config_1	164
9.4.10	spi_fifo_wdata	164
9.4.11	spi_fifo_rdata	165
10	UART	166
10.1	UART introduction	166
10.2	UART main features	166
10.3	UART function description	167
10.3.1	Data format description	167
10.3.2	Clock source	167
10.3.3	Baud rate setting	168
10.3.4	Transmitter	169
10.3.5	Receiver	169
10.3.6	Automatic baud rate detection	170
10.3.7	Hardware flow control	171
10.3.8	LIN transmission mode	171
10.3.9	DMA transfer mode	172
10.3.10	UART interrupt	172
10.4	Register description	173
10.4.1	utx_config	174
10.4.2	urx_config	175
10.4.3	uart_bit_prd	176
10.4.4	data_config	176
10.4.5	utx_ir_position	177
10.4.6	urx_ir_position	177
10.4.7	urx_rto_timer	177
10.4.8	uart_sw_mode	178
10.4.9	uart_int_sts	178

10.4.10	uart_int_mask	179
10.4.11	uart_int_clear	180
10.4.12	uart_int_en	181
10.4.13	uart_status	181
10.4.14	sts_urx_abr_prd	182
10.4.15	uart_fifo_config_0	182
10.4.16	uart_fifo_config_1	183
10.4.17	uart_fifo_wdata	184
10.4.18	uart_fifo_rdata	184
11	I2C	185
11.1	I2C introduction	185
11.2	I2C main features	185
11.3	I2C function description	185
11.3.1	Start and stop conditions	186
11.3.2	Data transmission format	186
11.3.3	Arbitration	187
11.4	I2C clock setting	188
11.5	I2C configuration process	188
11.5.1	Configuration item	188
11.5.2	Read and write flags	189
11.5.3	Slave address	189
11.5.4	Slave register address	189
11.5.5	Slave device address length	189
11.5.6	Data	189
11.5.7	Data length	189
11.5.8	Enable signal	190
11.6	FIFO management	190
11.7	Using DMA	191
11.7.1	DMA transmission process	191
11.7.2	DMA receiving process	192
11.8	I2C interrupt	192
11.9	Register description	192
11.9.1	i2c_config	193
11.9.2	i2c_int_sts	194
11.9.3	i2c_sub_addr	195
11.9.4	i2c_bus_busy	196
11.9.5	i2c_prd_start	196
11.9.6	i2c_prd_stop	197
11.9.7	i2c_prd_data	197

11.9.8	i2c_fifo_config_0	198
11.9.9	i2c_fifo_config_1	199
11.9.10	i2c_fifo_wdata	199
11.9.11	i2c_fifo_rdata	200
12	PWM	201
12.1	PWM introduction	201
12.2	PWM main features	201
12.3	PWM function description	201
12.3.1	Clock and divider	201
12.3.2	Pulse generation principle	202
12.3.3	PWM interrupt	203
12.4	Register description	203
12.4.1	pwm_int_config	205
12.4.2	pwm0_clkdiv	205
12.4.3	pwm0_thre1	205
12.4.4	pwm0_thre2	206
12.4.5	pwm0_period	206
12.4.6	pwm0_config	207
12.4.7	pwm0_interrupt	208
12.4.8	pwm1_clkdiv	208
12.4.9	pwm1_thre1	208
12.4.10	pwm1_thre2	209
12.4.11	pwm1_period	209
12.4.12	pwm1_config	210
12.4.13	pwm1_interrupt	211
12.4.14	pwm2_clkdiv	211
12.4.15	pwm2_thre1	211
12.4.16	pwm2_thre2	212
12.4.17	pwm2_period	212
12.4.18	pwm2_config	213
12.4.19	pwm2_interrupt	214
12.4.20	pwm3_clkdiv	214
12.4.21	pwm3_thre1	214
12.4.22	pwm3_thre2	215
12.4.23	pwm3_period	215
12.4.24	pwm3_config	216
12.4.25	pwm3_interrupt	217
12.4.26	pwm4_clkdiv	217
12.4.27	pwm4_thre1	217

12.4.28 pwm4_thre2	218
12.4.29 pwm4_period	218
12.4.30 pwm4_config	219
12.4.31 pwm4_interrupt	220
13 TIMER	221
13.1 TIMER introduction	221
13.2 TIMER main features	222
13.3 TIMER function description	222
13.3.1 8-bit divider	222
13.3.2 General timer operating mode	223
13.3.3 Watchdog timer operating mode	224
13.3.4 Alarm setting	224
13.3.5 Watchdog alarm	225
13.4 Register description	225
13.4.1 TCCR	227
13.4.2 TMR2_0	228
13.4.3 TMR2_1	228
13.4.4 TMR2_2	228
13.4.5 TMR2_0	229
13.4.6 TMR2_1	229
13.4.7 TMR2_2	229
13.4.8 TCR2	230
13.4.9 TCR3	230
13.4.10 TMSR2	230
13.4.11 TMSR3	231
13.4.12 TIER2	232
13.4.13 TIER3	232
13.4.14 TPLVR2	233
13.4.15 TPLVR3	233
13.4.16 TPLCR2	233
13.4.17 TPLCR3	234
13.4.18 WMER	234
13.4.19 WMR	235
13.4.20 WVR	235
13.4.21 WSR	236
13.4.22 TICR2	236
13.4.23 TICR3	237
13.4.24 WICR	237
13.4.25 TCER	238

13.4.26 TCMR	238
13.4.27 TILR2	239
13.4.28 TILR3	239
13.4.29 WCR	240
13.4.30 WFAR	240
13.4.31 WSAR	240
13.4.32 TCVWR2	241
13.4.33 TCVWR3	241
13.4.34 TCVSYN2	241
13.4.35 TCVSYN3	242
13.4.36 TCDR	242
14 QDEC	243
14.1 QDEC introduction	243
14.2 QDEC main features	243
14.3 QDEC function description	244
14.4 Register description	245
14.4.1 qdec0_ctrl0	246
14.4.2 qdec0_ctrl1	247
14.4.3 qdec0_value	248
14.4.4 qdec0_int_en	249
14.4.5 qdec0_int_sts	249
14.4.6 qdec0_int_clr	250
14.4.7 qdec1_ctrl0	250
14.4.8 qdec1_ctrl1	251
14.4.9 qdec1_value	252
14.4.10 qdec1_int_en	253
14.4.11 qdec1_int_sts	253
14.4.12 qdec1_int_clr	254
14.4.13 qdec2_ctrl0	254
14.4.14 qdec2_ctrl1	255
14.4.15 qdec2_value	256
14.4.16 qdec2_int_en	257
14.4.17 qdec2_int_sts	257
14.4.18 qdec2_int_clr	258
15 KeyScan	259
15.1 KYS introduction	259
15.2 KYS main features	259
15.3 KYS function description	259
15.3.1 Configurable number of rows and columns	259

15.3.2	GPIO selection	259
15.3.3	Key value	260
15.3.4	Interrupt	260
15.4	Register description	260
15.4.1	ks_ctrl	260
15.4.2	ks_int_en	261
15.4.3	ks_int_sts	262
15.4.4	keycode_clr	262
15.4.5	keycode_value	263
16	I2S	264
16.1	I2S introduction	264
16.2	I2S main features	264
16.3	I2S function description	264
16.4	Register description	265
16.4.1	i2s_config	265
16.4.2	i2s_int_sts	267
16.4.3	i2s_bclk_config	268
16.4.4	i2s_fifo_config_0	268
16.4.5	i2s_fifo_config_1	269
16.4.6	i2s_fifo_wdata	270
16.4.7	i2s_fifo_rdata	270
16.4.8	i2s_io_config	271
17	Emac	272
17.1	Emac introduction	272
17.2	Emac main features	272
17.3	Emac function description	273
17.4	Emac clock	275
17.5	Send and receive buffer descriptor (BD, Buffer Descriptor)	275
17.6	PHY interaction	275
17.7	Programming process	276
17.7.1	PHY initialization	276
17.7.2	Send data frame	277
17.7.3	Receive data frame	277
17.8	Register description	278
17.8.1	MODE	279
17.8.2	INT_SOURCE	280
17.8.3	INT_MASK	282
17.8.4	IPGT	283
17.8.5	PACKETLEN	284

17.8.6	COLLCONFIG	284
17.8.7	TX_BD_NUM	285
17.8.8	MIIMODE	286
17.8.9	MIICOMMAND	286
17.8.10	MIIADDRESS	287
17.8.11	MIITX_DATA	288
17.8.12	MIIRX_DATA	288
17.8.13	MIISTATUS	288
17.8.14	MAC_ADDR0	289
17.8.15	MAC_ADDR1	289
17.8.16	HASH0_ADDR	290
17.8.17	HASH1_ADDR	290
17.8.18	TXCTRL	290
18	USB	292
18.1	USB introduction	292
18.2	USB main features	292
18.3	USB function description	292
18.3.1	USB steps	292
18.3.2	Part of the register configuration and function description	293
18.3.3	USB enumeration phase interrupt processing flow	294
18.3.4	Register operation flow of each transfer transaction	295
18.4	Register description	298
18.4.1	usb_config	300
18.4.2	usb_lpm_config	302
18.4.3	usb_resume_config	303
18.4.4	usb_frame_no	303
18.4.5	usb_error	304
18.4.6	usb_int_en	305
18.4.7	usb_int_sts	306
18.4.8	usb_int_mask	308
18.4.9	usb_int_clear	309
18.4.10	ep1_config	311
18.4.11	ep2_config	312
18.4.12	ep3_config	313
18.4.13	ep4_config	314
18.4.14	ep5_config	315
18.4.15	ep6_config	316
18.4.16	ep7_config	317
18.4.17	ep0_fifo_config	318

18.4.18	ep0_fifo_status	318
18.4.19	ep0_tx_fifo_wdata	319
18.4.20	ep0_rx_fifo_rdata	320
18.4.21	ep1_fifo_config	320
18.4.22	ep1_fifo_status	321
18.4.23	ep1_tx_fifo_wdata	322
18.4.24	ep1_rx_fifo_rdata	322
18.4.25	ep2_fifo_config	322
18.4.26	ep2_fifo_status	323
18.4.27	ep2_tx_fifo_wdata	324
18.4.28	ep2_rx_fifo_rdata	324
18.4.29	ep3_fifo_config	325
18.4.30	ep3_fifo_status	326
18.4.31	ep3_tx_fifo_wdata	326
18.4.32	ep3_rx_fifo_rdata	327
18.4.33	ep4_fifo_config	327
18.4.34	ep4_fifo_status	328
18.4.35	ep4_tx_fifo_wdata	329
18.4.36	ep4_rx_fifo_rdata	329
18.4.37	ep5_fifo_config	329
18.4.38	ep5_fifo_status	330
18.4.39	ep5_tx_fifo_wdata	331
18.4.40	ep5_rx_fifo_rdata	331
18.4.41	ep6_fifo_config	332
18.4.42	ep6_fifo_status	333
18.4.43	ep6_tx_fifo_wdata	333
18.4.44	ep6_rx_fifo_rdata	334
18.4.45	ep7_fifo_config	334
18.4.46	ep7_fifo_status	335
18.4.47	ep7_tx_fifo_wdata	336
18.4.48	ep7_rx_fifo_rdata	336
18.4.49	xcvr_if_config	336
19	Revision history	339

List of Figures

2.1 Reset source	24
2.2 Clock architecture	25
3.1 Basic block diagram of GPIO	29
4.1 ADC block diagram	65
4.2 ADC Clock	67
5.1 DAC basic block diagram	91
6.1 DMA architecture	98
6.2 LLI architecture	100
7.1 L1c architecture	133
7.2 Cache architecture	135
8.1 nec logical	139
8.2 nec	139
8.3 rc5 logical	139
8.4 rc5	140
9.1 SPI clock	155
9.2 SPI Ignore waveform	156
10.1 UART data	167
10.2 UART clock	168
10.3 UART sampling waveform	169
10.4 UART fixed character mode waveform	170
10.5 UART flow control	171
11.1 I2C stop/start condition	186

11.2 I2C data transmission format	186
11.3 Master tx and slave rx	187
11.4 Master rx and slave tx	187
11.5 Tx and Rx together	188
12.1 PWM waveform	202
13.1 Timer block diagram	221
13.2 Watchdog timer block diagram	222
13.3 Timer Preload	223
13.4 Watchdog timing	224
13.5 Watchdog alarm mechanism	225
14.1 QDEC functional block diagram	245
17.1 EMAC architecture	274
18.1 USB interrupt trigger mode	294
18.2 USB communication method	297

List of Tables

1.1 Bus connection	19
1.2 Memory Map	20
1.3 Interrupt source	21
3.1 Software reset function table	27
3.2 GPIO function table 1	30
3.3 GPIO function table 2	31
3.4 GPIO function table 3	32
4.1 ADC internal signals	66
4.2 ADC external pins	66
4.3 ADC conversion result format	69
5.1 Internal reference voltage	92
7.1 WayDisable settings	134
11.1 Pin lists	185
12.1 Duty Cycle Parameters	203
16.1 Pin lists	264
17.1 Transmission signal	276
18.1 Register configuration 1	297
18.2 Register configuration 2	298
19.1 Document revision history	339

System and memory overview

1.1 Introduction

The on-chip processor uses RISC-V 32-bit with floating point. With high-speed processing memory system (see the L1C chapter for details), to achieve high-quality computing efficiency. External to the processor is a multilayer 32-bit AHB architecture with low power consumption, low latency, and high flexibility. The memory section contains high-speed tightly coupled memory as well as cache and system shared memory. Off-chip memory supports Flash expansion.

1.2 Main features

- RISC-V 32-bit with floating point
- Multi-layer 32-bit AHB bus architecture
- 132KB RAM
- 192KB ROM
- Off-chip memory Flash

1.3 Function description

The BL702/704/706 bus connection and address access are summarized as follows:

The bus master includes CPU, Ethernet, DMA, encryption engine, debugging interface. The bus slave includes memory, peripherals, and Zigbee/BLE. Except for Ethernet and encryption engine which can only access memory, all other bus masters can access all bus slaves.

Table 1.1: Bus connection

Slave / Master	CPU	Ethernet	DMA	Encryption engine	Debug interface
memory	V	V	V	V	V

Table 1.1: Bus connection(continued)

Slave / Master	CPU	Ethernet	DMA	Encryption engine	Debug interface
Peripheral	V	-	V	-	V
Zigbee/BLE	V	-	V	-	V

The address access mainly distinguishes “memory” or “peripheral” by [27:24], and the [31:28] bits can be ignored. The memory space is consecutive addresses 0x2010000~0x202FFFF (128KB SRAM), the read-only memory address is 0x1000000, and the deep sleep memory address is 0x0010000. The off-chip space address is 0x3000000 (maximum support 8MB Flash). The peripheral space is 0x0000000 ~ 0x000F000.

Table 1.2: Memory Map

Module	Base Address	Size	Description
RETRAM	0x40010000	4KB	Deep sleep memory (Retention RAM)
HBN	0x4000F000	4KB	Deep sleep control (Hibernate)
PDS	0x4000E000	4KB	Sleep control (Power Down Sleep)
USB	0x4000D800	1KB	USB control
EMAC	0x4000D000	2KB	Ethernet MAC control
DMA	0x4000C000	4KB	DMA control
QSPI	0x4000B000	4KB	Flash/pSRAM QSPI control
I2S	0x4000AA00	256B	I2S control
KYS	0x4000A900	256B	Key-Scan control
QDEC2	0x4000A880	64B	Quadrature decoder control
QDEC1	0x4000A840	64B	Quadrature decoder control
QDEC0	0x4000A800	64B	Quadrature decoder control
IRR	0x4000A600	256B	IR Remote control
TIMER	0x4000A500	256B	Timer control
PWM	0x4000A400	256B	Pulse Width Modulation *5 control
I2C	0x4000A300	256B	I2C control
SPI	0x4000A200	256B	SPI master/slave control
UART1	0x4000A100	256B	UART control (support LIN-bus)
UART0	0x4000A000	256B	UART control (support LIN-bus)
L1C	0x40009000	4KB	Cache control
eFuse	0x40007000	4KB	eFuse memory control
SEC	0x40004000	4KB	Security engine
GPIP	0x40002000	4KB	General purpose DAC/ADC/ACOMP interface control
MIX	0x40001000	4KB	Mixed signal register

Table 1.2: Memory Map(continued)

Module	Base Address	Size	Description
GLB	0x40000000	4KB	Global control register
pSRAM	0x24000000	8MB	pSRAM memory
XIP	0x23000000	8MB	XIP Flash memory
OCRAM	0x22020000	64KB	On-chip memory
DTCM	0x22014000	48KB	Data cache memory
ITCM	0x22010000	16KB	Instruction cache memory
ROM	0x21000000	192KB	Read-only memory

There are 64 interrupt sources. The level or edge trigger is configured by the CPU and can be masked. Details as follows:

Table 1.3: Interrupt source

Num	Signal source
54~63	wireless
53	brown-out
51~52	hbn_irq
50	pds_int
47~49	wireless
44	gpio_irq
40~42	qdec_int
39	kys_int
35~38	timer_irq
34	pwm_int
32	i2c_int
30	uart1_irq
29	uart0_irq
27	spi_int
26	efuse_int
25	adc_int
23	flash_int
22	emac_int
21	usb_int
19~20	ir_remote_int
18	i2s_int

Table 1.3: Interrupt source(continued)

Num	Signal source
15	dma_int
9~14	sec_eng_int
8	err_int
5~6	rf_int
0~4	err_int

2.1 Introduction

The reset sources included in the chip: hardware reset, watchdog reset, software reset. The chip contains multiple clock sources: XTAL, PLL, RC. It is allocated to each module through configuration such as frequency division.

2.2 Reset source

The reset sources are as follows:

- Hardware reset: reset via pin
 - Pin power reset (PU_CHIP = 0-> 1): similar to power reset
 - Power-on reset: the chip recovers from power failure, and HBN logic resets the chip system
- Watchdog reset
 - When the watchdog alarm triggers the reset signal, the reset management unit will reset the chip system after necessary preparations, and the internal logic of the watchdog will record the status of the watchdog reset
- Software reset: partial reset by software setting register
 - Software initial reset (reg_ctrl_pwron_rst): trigger the rising edge of this register by software to reset the chip system
 - Software CPU reset (reg_ctrl_cpu_reset): Trigger the rising edge of this register by software to reset the CPU part of the system
 - Software system reset (reg_ctrl_sys_reset): Trigger the rising edge of this register by software, retain necessary logic processing such as power management unit, and reset the chip part of the system
 - Software module reset: Set software reset according to the needs of specific modules

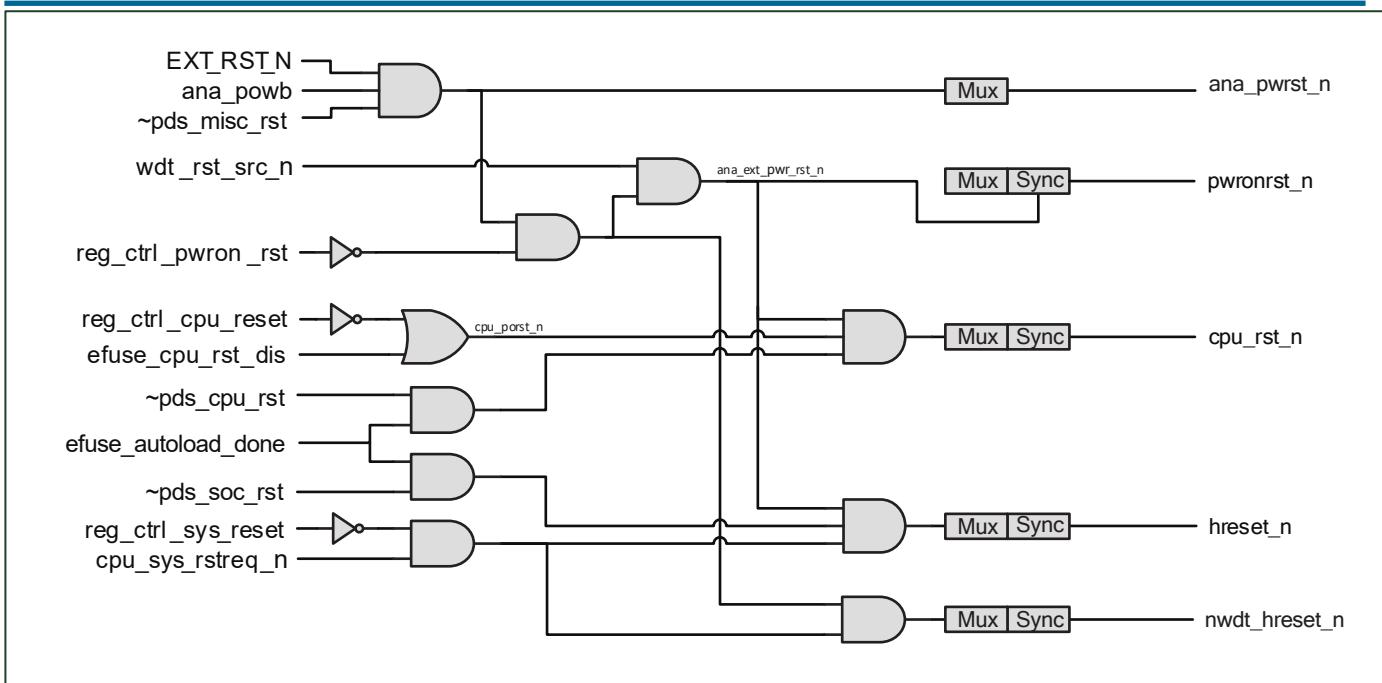


Fig. 2.1: Reset source

2.3 Clock source

The clock source includes:

- XTAL: External crystal oscillator clock, mainly supporting frequency 32MHz
- XTAL32K: external crystal oscillator clock, frequency 32KHz
- RC32M: RC oscillator clock, frequency 32MHz, provides calibration
- PLL: PLL clock, internal system high-speed clock, the highest frequency supports 144MHz

The clock control unit distributes the clock from the oscillator to the core and peripheral devices.

The system clock source, dynamic divider, clock configuration, and sleep use 32Khz clock can be selected to achieve low-power clock management.

Peripheral clocks include: Flash, USB2.0, Ethernet, UART, I2C, I2S, SPI, PWM, IR-remote, QDEC, KeyScan, ADC, DAC.

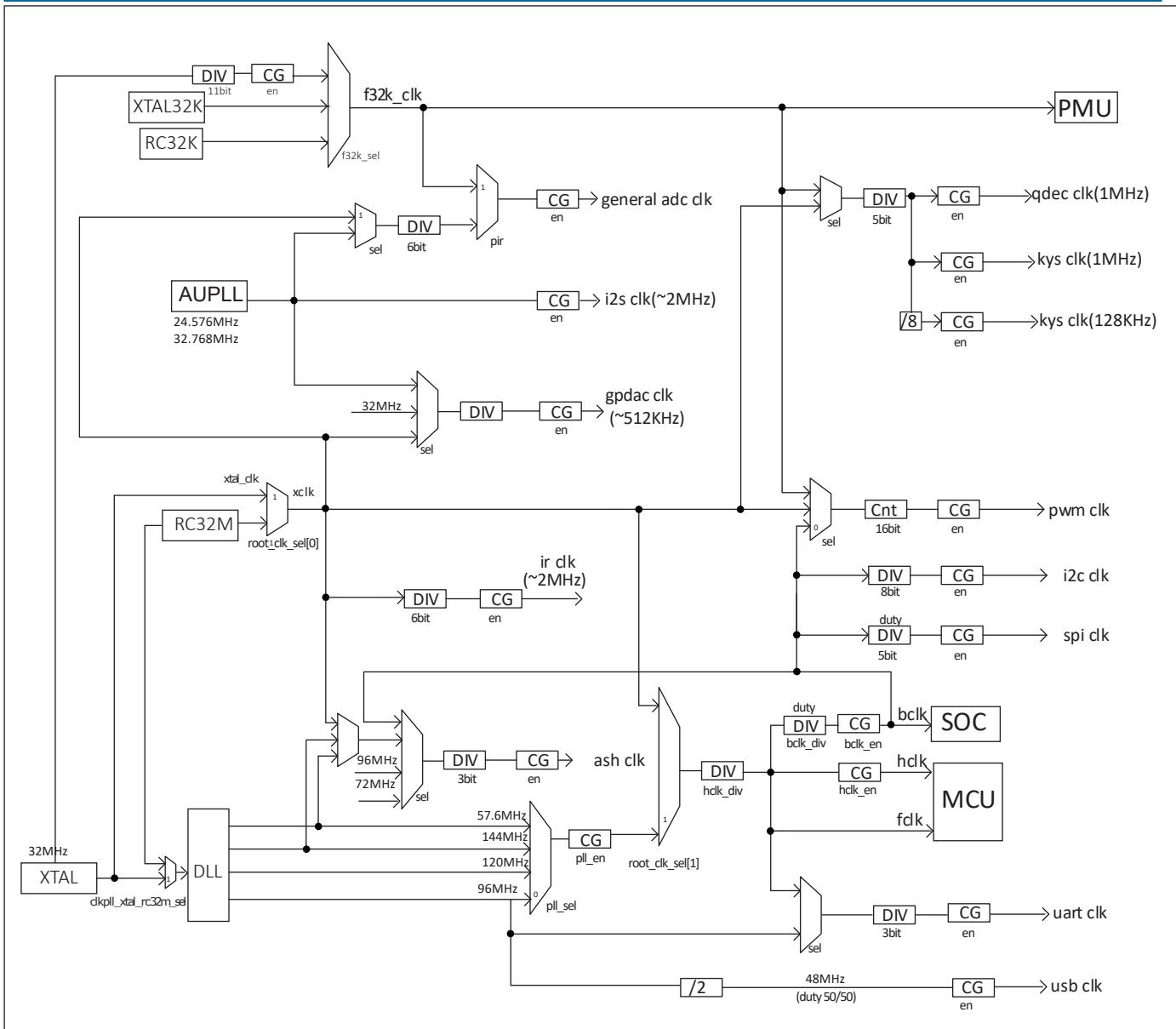


Fig. 2.2: Clock architecture

3.1 Introduction

GLB (Global Register) is a chip general global setting module, which mainly includes functions such as clock management, reset management, bus management, memory management, and GPIO management.

3.2 Function description

3.2.1 Clock management

The clock management function is mainly used to set the clock of the processor, bus, and various peripherals. Through this module, you can set the clock source and clock frequency division of the above-mentioned module. At the same time, it can also realize the gating of the above-mentioned module clock. To achieve the purpose of low power consumption of the system.

For detailed settings, please refer to the chapters on reset and clock.

3.2.2 Reset management

Provide individual reset function and chip reset function of each peripheral module. Chip reset includes:

- CPU reset: just reset the CPU module, the program will run again, and the peripherals will not be reset
- System reset: All peripherals and CPU will be reset, but related registers in the AON domain will not be reset
- Power-on reset: the entire system including the related registers of the AON domain will be reset

The application can choose to use the corresponding reset method as needed.

Table 3.1: Software reset function table

BL702	RST_- PIN/HBN/WDT/POWER ON	SW.Reset	CPU/PDS/SYS	PDS/CPU	PDS
CPU	✓			✓	
bus	✓		✓		
glb	✓	swrst_s1[0]			
gpip	✓	swrst_s1[2]	✓		
I1c	✓	swrst_s1[9]	✓	✓	
dma	✓	swrst_s1[12]	✓		
emac	✓	swrst_s1[13]	✓		
usb	✓	swrst_s1a[12]	✓		✓
pds		swrst_s1[14]			
uart0	✓	swrst_s1a[0]	✓		
uart1	✓	swrst_s1a[1]	✓		
spi	✓	swrst_s1a[2]	✓		
I2C	✓	swrst_s1a[3]	✓		
pwm	✓	swrst_s1a[4]	✓		
timer	✓	swrst_s1a[5]	✓		
irr	✓	swrst_s1a[6]	✓		
qdec0	✓	swrst_s1a[8]	✓		
qdec1	✓	swrst_s1a[8]	✓		
qdec2	✓	swrst_s1a[8]	✓		
kys	✓	swrst_s1a[9]	✓		
i2s	✓	swrst_s1a[10]	✓		

3.2.3 Bus management

Provide bus arbitration settings and bus error settings, you can set whether to generate an interrupt when a bus error occurs, and provide error bus address information to facilitate user debugging.

3.2.4 Memory management

Provides power management of each memory module in the chip system low power mode, including two setting modes:

- Retention mode: In this mode, the data on the memory can be saved, but it cannot be read or written before exiting the low-power mode
- Sleep mode: In this mode, the data in the memory will be lost and it is only used to reduce system power consumption

3.2.5 GPIO overview

The GPIO management function provides GPIO control registers to implement software configuration of GPIO properties, enabling users to conveniently operate GPIO. Each GPIO can be configured as software GPIO or other multiplexing functions. Under each function, three port states are provided: pull-up, pull-down, and floating (must be set to floating when configured as an analog function). In addition, GPIO also provides interrupt function, which can be configured as rising edge trigger, falling edge trigger, high level trigger or low level trigger. Each GPIO can have two sets of interrupt configurations, and the two sets of interrupt configurations can take effect at the same time. For example, if the INT0 of GPIO0 is configured as a rising edge trigger and INT1 is configured as a falling edge trigger, the final effect is that both edges of GPIO will trigger interrupts.

3.2.6 GPIO main features

- Can be configured as software GPIO function
- It can be configured as other multiplexed functions and used with peripheral functions. When configured as analog functions, it must be set to floating
- You can set the input or output mode and set it as pull-up, pull-down or floating
- The drive capacity can be set to provide greater output current
- Schmitt trigger function can be set to provide simple hardware anti-shake function

3.2.7 GPIO function description

Each GPIO can be configured by software as:

- Multiple functions: I2S, SPI, I2C, UART, PWM, USB, SWGPIO…up to 24 functions
- InputEnable/OutputEnable: input, output, high impedance (ie=0, oe=0)
- PullUp/PullDown: pull up, pull down, float (pu=0, pd=0)
- drive strength: four gears of 0, 1, 2, and 3. The larger the value, the stronger the drive ability
- smt trigger: smt enable, smt disable, to prevent jitter near the trigger threshold

When the GPIO multiplexing function is configured as SWGPIO input, you can also configure the interrupt trigger mode for it, and each GPIO can have two interrupt modes, and both interrupt modes can be effective separately/simultaneously:

- Interrupt mode 1: rising edge trigger, falling edge trigger, high level trigger, low level trigger
- Interrupt mode 2: rising edge trigger, falling edge trigger, high level trigger, low level trigger

The basic block diagram of the GPIO module is shown in the figure.

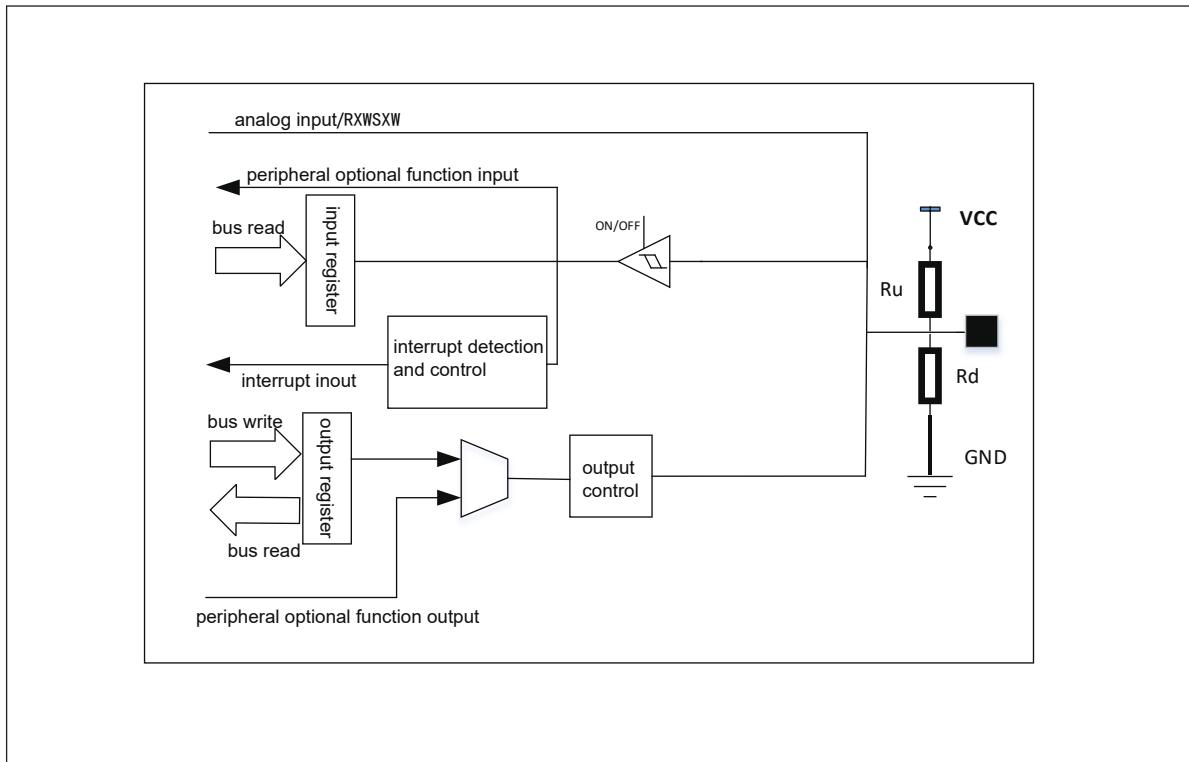


Fig. 3.1: Basic block diagram of GPIO

3.2.8 GPIO function setting

The function of GPIO is set through the GPIO_CFGCTL register group. The main setting items include:

- func_sel: select GPIO function
- pu: Choose whether to pull up
- pd: Choose whether to drop down
- drv: set drive capability
- smt: Choose whether to enable Schmitt trigger
- ie: set input enable
- oe: set output enable

The functions that GPIO can be set include:

- Flash/QSPI: Set GPIO as QSPI function, which can be connected to Flash as a program storage/running medium
- SPI: Set GPIO as SPI function
- I2C: Set GPIO as I2C function
- UART: Set GPIO as UART function
- PWM: set GPIO as PWM function
- ANA: Set GPIO as Analog function
- SWGPIO: set GPIO as general IO function
- JTAG: Set GPIO as JTAG function
- Other multiplexing functions

In order to meet customer needs as much as possible, each GPIO can basically select the above optional functions. When an optional function is selected, GPIO and the corresponding function signal are shown in the following table:

Table 3.2: GPIO function table 1

GPIO	CLK_OUT	Flash_PSRAM	I2S	SPI0	I2C	UART	PWM
GPIO0	clk_out[0]		I2S0_BCLK	SPI_0_MOSI	I2C0_SCL	UART_SIG0	PWM[0]
GPIO1	clk_out[1]		I2S0_FS	SPI_0_MISO	I2C0_SDA	UART_SIG1	PWM[1]
GPIO2	clk_out[0]		I2S0_DIO/I2S0_DO	SPI_0_SS	I2C0_SCL	UART_SIG2	PWM[2]
GPIO3	clk_out[1]		I2S0_RCLK_O/I2S0_DI	SPI_0_SCLK	I2C0_SDA	UART_SIG3	PWM[3]
GPIO4	clk_out[0]		I2S0_BCLK	SPI_0_MOSI	I2C0_SCL	UART_SIG4	PWM[4]
GPIO5	clk_out[1]		I2S0_FS	SPI_0_MISO	I2C0_SDA	UART_SIG5	PWM[0]
GPIO6	clk_out[0]		I2S0_DIO/I2S0_DO	SPI_0_SS	I2C0_SCL	UART_SIG6	PWM[1]
GPIO7	clk_out[1]		I2S0_RCLK_O/I2S0_DI	SPI_0_SCLK	I2C0_SDA	UART_SIG7	PWM[2]
GPIO8	clk_out[0]		I2S0_BCLK	SPI_0_MOSI	I2C0_SCL	UART_SIG0	PWM[3]
GPIO9	clk_out[1]		I2S0_FS	SPI_0_MISO	I2C0_SDA	UART_SIG1	PWM[4]
GPIO10	clk_out[0]		I2S0_DIO/I2S0_DO	SPI_0_SS	I2C0_SCL	UART_SIG2	PWM[0]
GPIO11	clk_out[1]		I2S0_RCLK_O/I2S0_DI	SPI_0_SCLK	I2C0_SDA	UART_SIG3	PWM[1]
GPIO12	clk_out[0]		I2S0_BCLK	SPI_0_MOSI	I2C0_SCL	UART_SIG4	PWM[2]
GPIO13	clk_out[1]		I2S0_FS	SPI_0_MISO	I2C0_SDA	UART_SIG5	PWM[3]
GPIO14	clk_out[0]		I2S0_DIO/I2S0_DO	SPI_0_SS	I2C0_SCL	UART_SIG6	PWM[4]
GPIO15	clk_out[1]		I2S0_RCLK_O/I2S0_DI	SPI_0_SCLK	I2C0_SDA	UART_SIG7	PWM[0]
GPIO16	clk_out[0]		I2S0_BCLK	SPI_0_MOSI	I2C0_SCL	UART_SIG0	PWM[1]
GPIO17	clk_out[1]	SF_IO_0/SF2_CS2	I2S0_FS	SPI_0_MISO	I2C0_SDA	UART_SIG1	PWM[2]
GPIO18	clk_out[0]	SF_IO_1	I2S0_DIO/I2S0_DO	SPI_0_SS	I2C0_SCL	UART_SIG2	PWM[3]
GPIO19	clk_out[1]	SF_CS	I2S0_RCLK_O/I2S0_DI	SPI_0_SCLK	I2C0_SDA	UART_SIG3	PWM[4]
GPIO20	clk_out[0]	SF_IO_3	I2S0_BCLK	SPI_0_MOSI	I2C0_SCL	UART_SIG4	PWM[0]
GPIO21	clk_out[1]	SF_CLK	I2S0_FS	SPI_0_MISO	I2C0_SDA	UART_SIG5	PWM[1]
GPIO22	clk_out[0]	SF_IO_2	I2S0_DIO/I2S0_DO	SPI_0_SS	I2C0_SCL	UART_SIG6	PWM[2]
GPIO23	clk_out[1]	SF2_IO_2	I2S0_RCLK_O/I2S0_DI	SPI_0_SCLK	I2C0_SDA	UART_SIG7	PWM[3]
GPIO24	clk_out[0]	SF2_IO_1	I2S0_BCLK	SPI_0_MOSI	I2C0_SCL	UART_SIG0	PWM[4]
GPIO25	clk_out[1]	SF2_CS	I2S0_FS	SPI_0_MISO	I2C0_SDA	UART_SIG1	PWM[0]
GPIO26	clk_out[0]	SF2_IO_3	I2S0_DIO/I2S0_DO	SPI_0_SS	I2C0_SCL	UART_SIG2	PWM[1]
GPIO27	clk_out[1]	SF2_CLK	I2S0_RCLK_O/I2S0_DI	SPI_0_SCLK	I2C0_SDA	UART_SIG3	PWM[2]
GPIO28	clk_out[0]	SF2_IO_0	I2S0_BCLK	SPI_0_MOSI	I2C0_SCL	UART_SIG4	PWM[3]
GPIO29	clk_out[1]		I2S0_FS	SPI_0_MISO	I2C0_SDA	UART_SIG5	PWM[4]
GPIO30	clk_out[0]		I2S0_DIO/I2S0_DO	SPI_0_SS	I2C0_SCL	UART_SIG6	PWM[0]

Table 3.2: GPIO function table 1(continued)

GPIO	CLK_OUT	Flash_PSRAM	I2S	SPI0	I2C	UART	PWM
GPIO31	clk_out[1]		I2S0_RCLK_O/I2S0_DI	SPI_0_SCLK	I2C0_SDA	UART_SIG7	PWM[1]

Table 3.3: GPIO function table 2

GPIO	Analog	SWGPI0	JTAG	Ether_Mac	QDEC
GPIO0		REG_GPIO[0]	E21_TMS/E21_TCK	MII_REF_CLK	qdec0_a
GPIO1		REG_GPIO[1]	E21_TDI/E21_TDO	MII_TXD[0]	qdec0_b
GPIO2		REG_GPIO[2]	E21_TCK/E21_TMS	MII_TXD[1]	qdec0_led
GPIO3		REG_GPIO[3]	E21_TDO/E21_TDI		qdec1_a
GPIO4		REG_GPIO[4]	E21_TMS/E21_TCK		qdec1_b
GPIO5		REG_GPIO[5]	E21_TDI/E21_TDO		qdec1_led
GPIO6		REG_GPIO[6]	E21_TCK/E21_TMS		qdec2_a
GPIO7	USB_DP/gpip_ch[6]/gpdac_vref_ext	REG_GPIO[7]	E21_TDO/E21_TDI	MII_RXD[0]	qdec2_b
GPIO8	USB_DM/gpip_ch[0]	REG_GPIO[8]	E21_TMS/E21_TCK	MII_RXD[1]	qdec2_led
GPIO9	pmip_dc_tp/clkpll_dc_tp/gpip_ch[7]	REG_GPIO[9]	E21_TDI/E21_TDO		qdec0_a
GPIO10	MICBIAS	REG_GPIO[10]	E21_TCK/E21_TMS		qdec0_b
GPIO11	gpip_ch[3]	REG_GPIO[11]	E21_TDO/E21_TDI		qdec0_led
GPIO12	gpip_ch[4]	REG_GPIO[12]	E21_TMS/E21_TCK		qdec1_a
GPIO13		REG_GPIO[13]	E21_TDI/E21_TDO		qdec1_b
GPIO14	gpip_ch[5]/atest_out_0	REG_GPIO[14]	E21_TCK/E21_TMS		qdec1_led
GPIO15	gpip_ch[1]/atest_out_1	REG_GPIO[15]	E21_TDO/E21_TDI		qdec2_a
GPIO16		REG_GPIO[16]	E21_TMS/E21_TCK		qdec2_b
GPIO17	gpip_ch[2]/psw_irrcv	REG_GPIO[17]	E21_TDI/E21_TDO		qdec2_led
GPIO18	gpip_ch[8]	REG_GPIO[18]	E21_TCK/E21_TMS	RMII_MDC	qdec0_a
GPIO19	gpip_ch[9]	REG_GPIO[19]	E21_TDO/E21_TDI	RMII_MDIO	qdec0_b
GPIO20	gpip_ch[10]	REG_GPIO[20]	E21_TMS/E21_TCK	RMII_RXERR	qdec0_led
GPIO21	gpip_ch[11]	REG_GPIO[21]	E21_TDI/E21_TDO	RMII_TX_EN	qdec1_a
GPIO22	leddrv[0]	REG_GPIO[22]	E21_TCK/E21_TMS	RMII_RX_DV	qdec1_b
GPIO23	leddrv[1]/flash_pull_out[0]	REG_GPIO[23]	E21_TDO/E21_TDI		qdec1_led
GPIO24	flash_pull_out[1]	REG_GPIO[24]	E21_TMS/E21_TCK	RMII_MDC	qdec2_a
GPIO25	flash_pull_out[2]	REG_GPIO[25]	E21_TDI/E21_TDO	RMII_MDIO	qdec2_b
GPIO26	flash_pull_out[3]	REG_GPIO[26]	E21_TCK/E21_TMS	RMII_RXERR	qdec2_led
GPIO27	flash_pull_out[4]	REG_GPIO[27]	E21_TDO/E21_TDI	RMII_TX_EN	qdec0_a
GPIO28	flash_pull_out[5]	REG_GPIO[28]	E21_TMS/E21_TCK	RMII_RX_DV	qdec0_b
GPIO29		REG_GPIO[29]	E21_TDI/E21_TDO		qdec0_led

Table 3.3: GPIO function table 2(continued)

GPIO	Analog	SWGPIO	JTAG	Ether_Mac	QDEC
GPIO30		REG_GPIO[30]	E21_TCK/E21_TMS		qdec1_a
GPIO31		REG_GPIO[31]	E21_TDO/E21_TDI		qdec1_b

Table 3.4: GPIO function table 3

GPIO	SWGPIO
GPIO17	pad_irrx_i,irrxgpsl=1
GPIO18	pad_irrx_i,irrxgpsl=2
GPIO19	pad_irrx_i,irrxgpsl=3
GPIO20	pad_irrx_i,irrxgpsl=4
GPIO21	pad_irrx_i,irrxgpsl=5
GPIO22	pad_irrx_i,irrxgpsl=6
GPIO23	pad_irrx_i,irrxgpsl=7
GPIO24	pad_irrx_i,irrxgpsl=8
GPIO25	pad_irrx_i,irrxgpsl=9
GPIO26	pad_irrx_i,irrxgpsl=10
GPIO27	pad_irrx_i,irrxgpsl=11
GPIO28	pad_irrx_i,irrxgpsl=12
GPIO29	pad_irrx_i,irrxgpsl=13
GPIO30	pad_irrx_i,irrxgpsl=14
GPIO31	pad_irrx_i,irrxgpsl=15

When using the IR function, you need to set GPIO as SWGPIO and set the irrxgpsl register (GPIO17-GPIO31 can be used as IR pins)

In the above table, when the UART function is selected, only a signal of the UART is selected, and the specific function of the pin (such as UART TX or UART RX) is not specified. The user needs to further select specific UART signals and corresponding functions through UART_SIGX_SEL (X=0-7).

The signals that can be selected for each UART_SIGX_SEL include:

- 0 : UART0_RTS
- 1 : UART0_CTS
- 2 : UART0_RXD

- 3 : UART0_RXD
- 4 : UART1_RTS
- 5 : UART1_CTS
- 6 : UART1_TXD
- 7 : UART1_RXD

Take GPIO0 as an example, when fun_sel selects UART, GPIO0 selects UART_SIG0. By default, the value of UART_SIG0_SEL is 0, which is UART0_RTS, that is, GPIO is UART0_RTS function. If the application wants to use GPIO as UART1_TXD, then just set UART_SIG0_SEL to 6, then the function of GPIO0 is UART1_TXD.

3.2.9 GPIO output settings

By setting func_sel to SWGPIO, GPIO can be used as the input/output of ordinary GPIO. Set IE to 0 and OE to 1, then GPIO can be configured as an output function, and the output value is set through the GPIO_O register group.

When the corresponding Bit of GPIO_O is set to 0, GPIO outputs low level. When the corresponding Bit of GPIO_O is set to 1, GPIO outputs high level. The output capability can be set through the DRV control bit.

3.2.10 GPIO input settings

By setting func_sel to SWGPIO, setting IE to 1, and OE to 0, users can configure GPIO as an input function. Set whether to enable the Schmitt trigger through the SMT control bit, and set the pull-up and pull-down attributes through the PD and PU control bits.

The value of external input can be obtained by reading the bit corresponding to the GPIO_I register.

3.2.11 GPIO optional function settings

By setting func_sel as the corresponding peripheral function, the connection between the GPIO and the peripheral can be realized, and the input and output of the peripheral can be realized. It can be seen from the basic functional block diagram of GPIO that when selecting optional functions, you need to set IE to 1, and OE to 0, which means that the output control function of ordinary GPIO is disconnected.

In this way, for a peripheral with a fixed input function, the OE signal of the peripheral is always 0, thereby realizing the input function. For a fixed output peripheral, its OE signal is always 1, so that the output is controlled by the peripheral, and the input signal at this time is the output signal, but it will not be collected by the peripheral being output. When the peripheral needs both input and output, the input and output can be realized by controlling the OE signal of the peripheral.

That is: for functions other than swgpio, as the output direction function, the configuration values of IE and OE do not affect the function. But as the input direction, IE must be set and the configuration of OE does not affect the function; when used as swgpio, both IE and OE need to be configured correctly.

3.2.12 GPIO interrupt settings

To use the interrupt function of GPIO, you need to set GPIO to input mode first, and the interrupt trigger mode is set through the GPIO_INT_MODE_SET register group. The interrupt modes that can be set include:

- Falling edge trigger interrupt
- Rising edge trigger interrupt
- Low level trigger interrupt
- High level trigger interrupt

Each GPIO can be set as an interrupt function. Whether to enable a GPIO interrupt can be set through the GPIO_INT_MASK register. When an interrupt is generated, the GPIO pin number that generated the interrupt can be obtained through the GPIO_INT_STAT register in the interrupt function. Clear the corresponding interrupt signal through GPIO_INT_CLR.

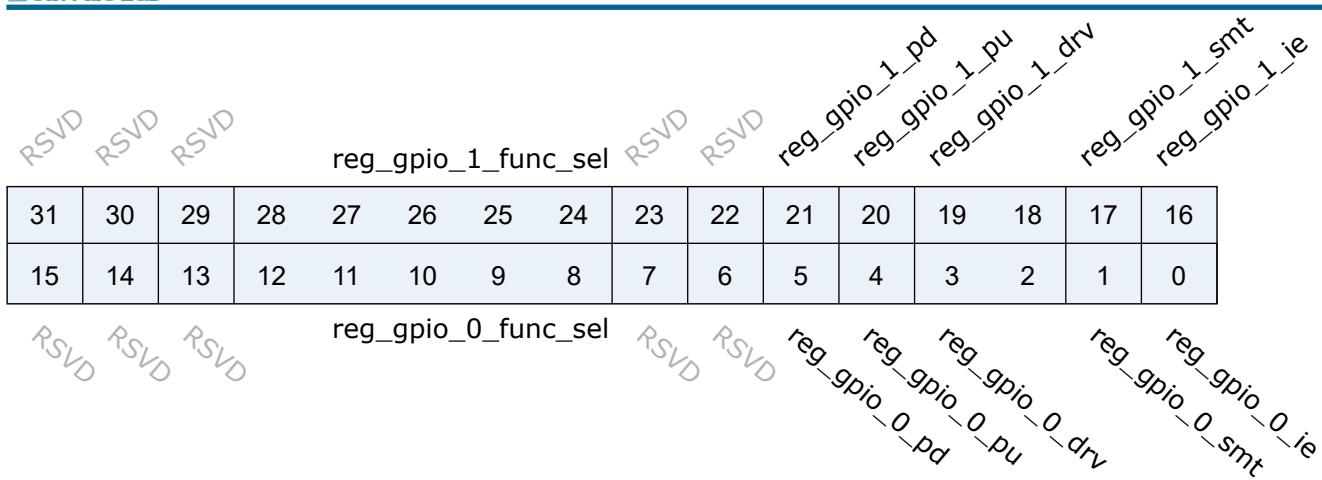
3.3 Register description

Name	Description
GPIO_CFGCTL0	
GPIO_CFGCTL1	
GPIO_CFGCTL2	
GPIO_CFGCTL3	
GPIO_CFGCTL4	
GPIO_CFGCTL5	
GPIO_CFGCTL6	
GPIO_CFGCTL7	
GPIO_CFGCTL8	
GPIO_CFGCTL9	
GPIO_CFGCTL10	
GPIO_CFGCTL11	
GPIO_CFGCTL12	
GPIO_CFGCTL13	
GPIO_CFGCTL14	
GPIO_CFGCTL15	

Name	Description
GPIO_CFGCTL30	
GPIO_CFGCTL32	
GPIO_CFGCTL34	
GPIO_CFGCTL35	
GPIO_INT_MASK1	
GPIO_INT_STAT1	
GPIO_INT_CLR1	
GPIO_INT_MODE_SET1	
GPIO_INT_MODE_SET2	
GPIO_INT_MODE_SET3	
GPIO_INT_MODE_SET4	
GPIO_INT2_MASK1	
GPIO_INT2_STAT1	
GPIO_INT2_CLR1	
GPIO_INT2_MODE_SET1	
GPIO_INT2_MODE_SET2	
GPIO_INT2_MODE_SET3	
GPIO_INT2_MODE_SET4	

3.3.1 GPIO_CFGCTL0

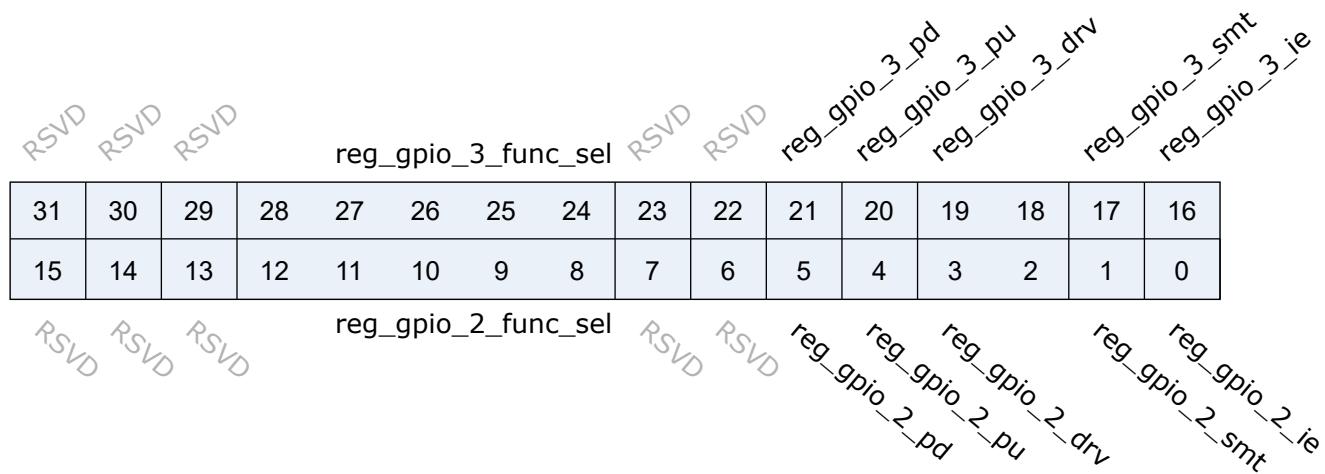
Address: 0x40000100



Bits	Name	Type	Reset	Description							
31:29	RSVD										
28:24	reg_gpio_1_func_sel	r/w	5'hE	GPIO Function Select (Default : JTAG)							
23:22	RSVD										
21	reg_gpio_1_pd	r/w	0	GPIO Pull Down Control							
20	reg_gpio_1_pu	r/w	0	GPIO Pull Up Control							
19:18	reg_gpio_1_drv	r/w	0	GPIO Driving Control							
17	reg_gpio_1_smt	r/w	1	GPIO SMT Control							
16	reg_gpio_1_ie	r/w	1	GPIO Input Enable							
15:13	RSVD										
12:8	reg_gpio_0_func_sel	r/w	5'hE	GPIO Function Select (Default : JTAG)							
7:6	RSVD										
5	reg_gpio_0_pd	r/w	0	GPIO Pull Down Control							
4	reg_gpio_0_pu	r/w	0	GPIO Pull Up Control							
3:2	reg_gpio_0_drv	r/w	0	GPIO Driving Control							
1	reg_gpio_0_smt	r/w	1	GPIO SMT Control							
0	reg_gpio_0_ie	r/w	1	GPIO Input Enable							

3.3.2 GPIO_CFGCTL1

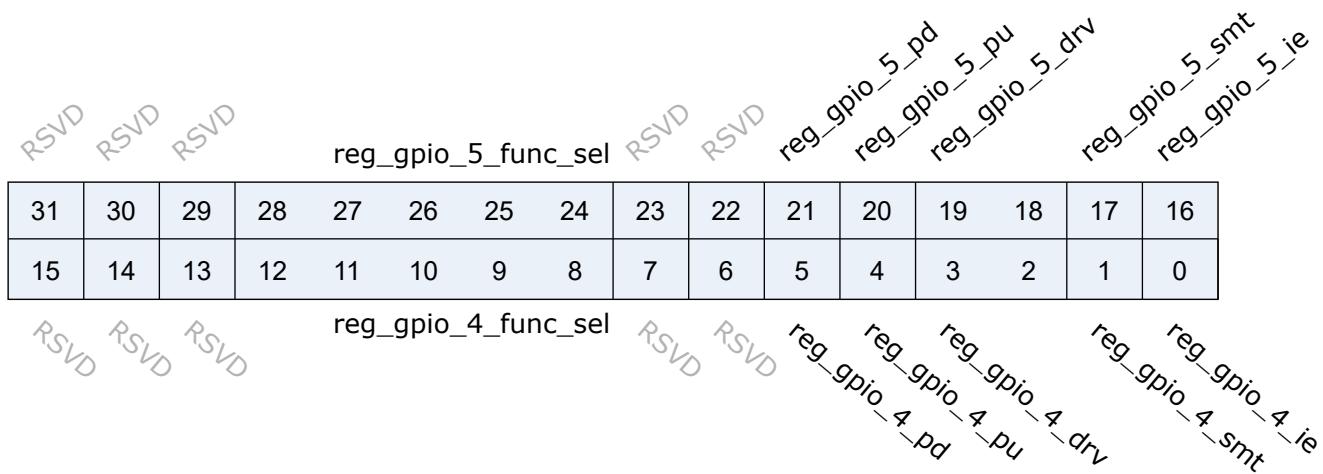
Address: 0x40000104



Bits	Name	Type	Reset	Description			
31:29	RSVD						
28:24	reg_gpio_3_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)			
23:22	RSVD						
21	reg_gpio_3_pd	r/w	0	GPIO Pull Down Control			
20	reg_gpio_3_pu	r/w	0	GPIO Pull Up Control			
19:18	reg_gpio_3_drv	r/w	0	GPIO Driving Control			
17	reg_gpio_3_smt	r/w	1	GPIO SMT Control			
16	reg_gpio_3_ie	r/w	1	GPIO Input Enable			
15:13	RSVD						
12:8	reg_gpio_2_func_sel	r/w	5'hE	GPIO Function Select (Default : JTAG)			
7:6	RSVD						
5	reg_gpio_2_pd	r/w	0	GPIO Pull Down Control			
4	reg_gpio_2_pu	r/w	0	GPIO Pull Up Control			
3:2	reg_gpio_2_drv	r/w	0	GPIO Driving Control			
1	reg_gpio_2_smt	r/w	1	GPIO SMT Control			
0	reg_gpio_2_ie	r/w	1	GPIO Input Enable			

3.3.3 GPIO_CFGCTL2

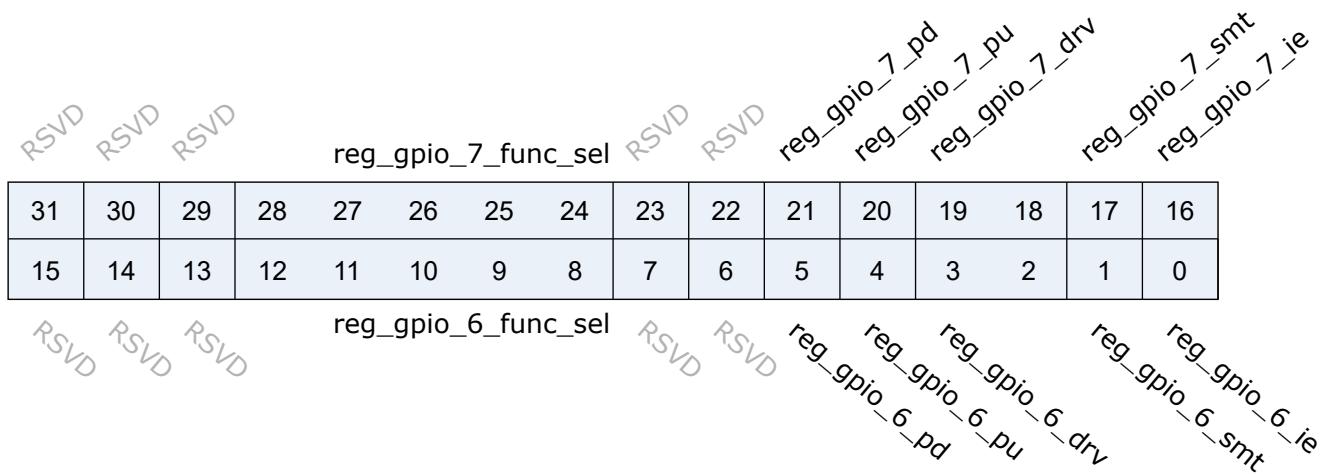
Address: 0x40000108



Bits	Name	Type	Reset	Description			
31:29	RSVD						
28:24	reg_gpio_5_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)			
23:22	RSVD						
21	reg_gpio_5_pd	r/w	0	GPIO Pull Down Control			
20	reg_gpio_5_pu	r/w	0	GPIO Pull Up Control			
19:18	reg_gpio_5_drv	r/w	0	GPIO Driving Control			
17	reg_gpio_5_smt	r/w	1	GPIO SMT Control			
16	reg_gpio_5_ie	r/w	1	GPIO Input Enable			
15:13	RSVD						
12:8	reg_gpio_4_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)			
7:6	RSVD						
5	reg_gpio_4_pd	r/w	0	GPIO Pull Down Control			
4	reg_gpio_4_pu	r/w	0	GPIO Pull Up Control			
3:2	reg_gpio_4_drv	r/w	0	GPIO Driving Control			
1	reg_gpio_4_smt	r/w	1	GPIO SMT Control			
0	reg_gpio_4_ie	r/w	1	GPIO Input Enable			

3.3.4 GPIO_CFGCTL3

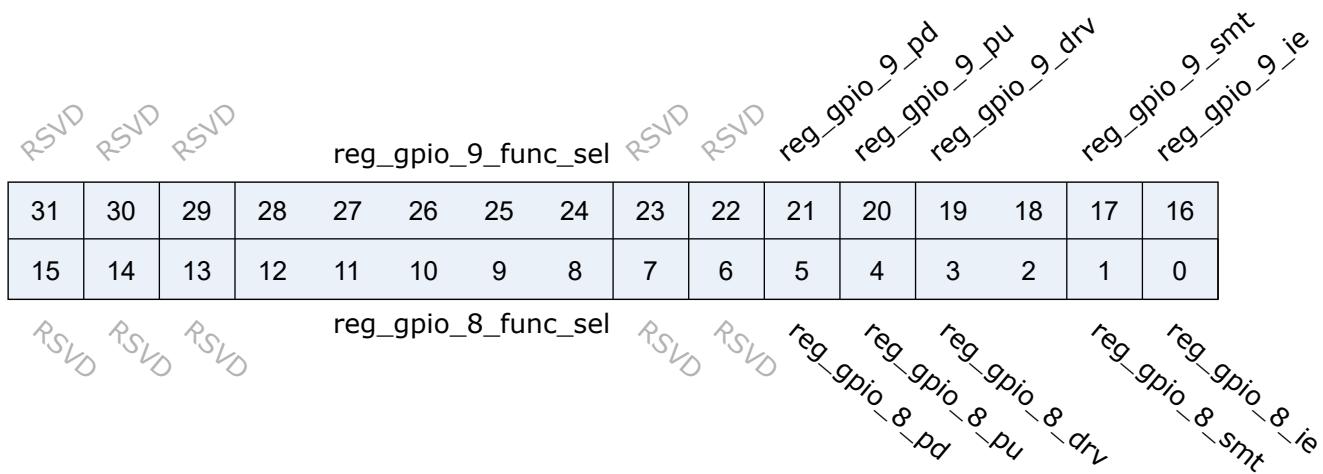
Address: 0x4000010c



Bits	Name	Type	Reset	Description			
31:29	RSVD						
28:24	reg_gpio_7_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)			
23:22	RSVD						
21	reg_gpio_7_pd	r/w	0	GPIO Pull Down Control			
20	reg_gpio_7_pu	r/w	0	GPIO Pull Up Control			
19:18	reg_gpio_7_drv	r/w	0	GPIO Driving Control			
17	reg_gpio_7_smt	r/w	1	GPIO SMT Control			
16	reg_gpio_7_ie	r/w	1	GPIO Input Enable			
15:13	RSVD						
12:8	reg_gpio_6_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)			
7:6	RSVD						
5	reg_gpio_6_pd	r/w	0	GPIO Pull Down Control			
4	reg_gpio_6_pu	r/w	0	GPIO Pull Up Control			
3:2	reg_gpio_6_drv	r/w	0	GPIO Driving Control			
1	reg_gpio_6_smt	r/w	1	GPIO SMT Control			
0	reg_gpio_6_ie	r/w	1	GPIO Input Enable			

3.3.5 GPIO_CFGCTL4

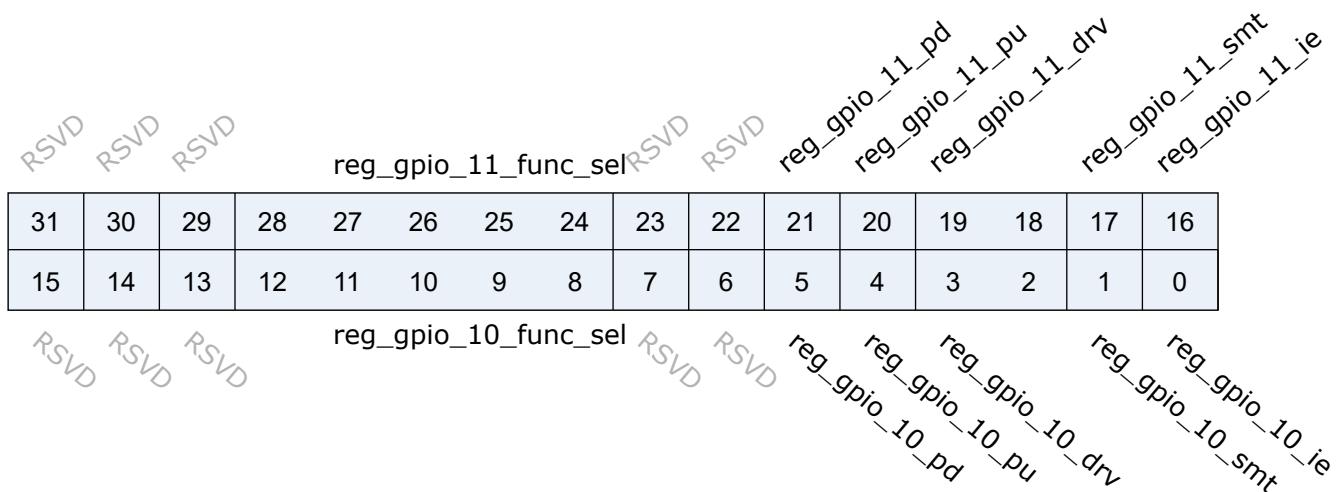
Address: 0x40000110



Bits	Name	Type	Reset	Description							
31:29	RSVD										
28:24	reg_gpio_9_func_sel	r/w	5'hE	GPIO Function Select (Default : JTAG)							
23:22	RSVD										
21	reg_gpio_9_pd	r/w	0	GPIO Pull Down Control (Use this bit if reg_en_hw_pu_pd := 0 (0x4000F014[16]))							
20	reg_gpio_9_pu	r/w	0	GPIO Pull Up Control (Use this bit if reg_en_hw_pu_pd := 0 (0x4000F014[16]))							
19:18	reg_gpio_9_drv	r/w	0	GPIO Driving Control							
17	reg_gpio_9_smt	r/w	1	GPIO SMT Control(Useless, IE depend on reg_aon_pad_ie_smt[0] : 0x4000F014[8])							
16	reg_gpio_9_ie	r/w	0	GPIO Input Enable (Useless, IE depend on reg_aon_pad_ie_smt[0] : 0x4000F014[8])							
15:13	RSVD										
12:8	reg_gpio_8_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)							
7:6	RSVD										
5	reg_gpio_8_pd	r/w	0	GPIO Pull Down Control							
4	reg_gpio_8_pu	r/w	0	GPIO Pull Up Control							
3:2	reg_gpio_8_drv	r/w	0	GPIO Driving Control							
1	reg_gpio_8_smt	r/w	1	GPIO SMT Control							
0	reg_gpio_8_ie	r/w	1	GPIO Input Enable							

3.3.6 GPIO_CFGCTL5

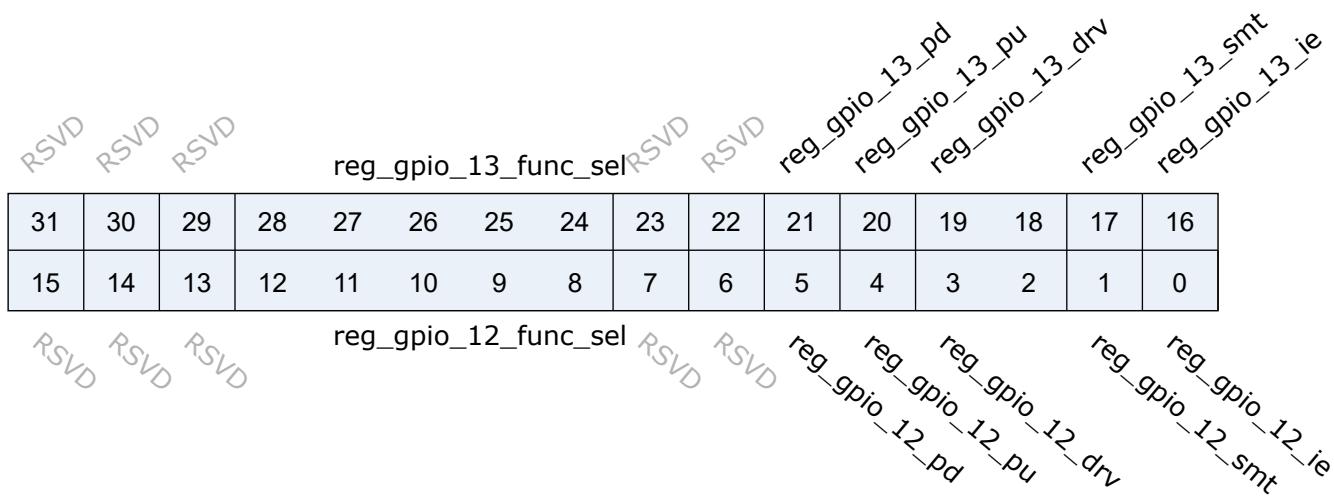
Address: 0x40000114



Bits	Name	Type	Reset	Description
31:29	RSVD			
28:24	reg_gpio_11_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)
23:22	RSVD			
21	reg_gpio_11_pd	r/w	0	GPIO Pull Down Control
20	reg_gpio_11_pu	r/w	0	GPIO Pull Up Control
19:18	reg_gpio_11_drv	r/w	0	GPIO Driving Control
17	reg_gpio_11_smt	r/w	1	GPIO SMT Control(Useless, IE depend on reg_aon_pad_ie_smt[2] : 0x4000F014[10])
16	reg_gpio_11_ie	r/w	1	GPIO Input Enable (Useless, IE depend on reg_aon_pad_ie_smt[2] : 0x4000F014[10])
15:13	RSVD			
12:8	reg_gpio_10_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)
7:6	RSVD			
5	reg_gpio_10_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_10_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_10_drv	r/w	0	GPIO Driving Control
1	reg_gpio_10_smt	r/w	1	GPIO SMT Control(Useless, IE depend on reg_aon_pad_ie_smt[1] : 0x4000F014[9])
0	reg_gpio_10_ie	r/w	1	GPIO Input Enable (Useless, IE depend on reg_aon_pad_ie_smt[1] : 0x4000F014[9])

3.3.7 GPIO_CFGCTL6

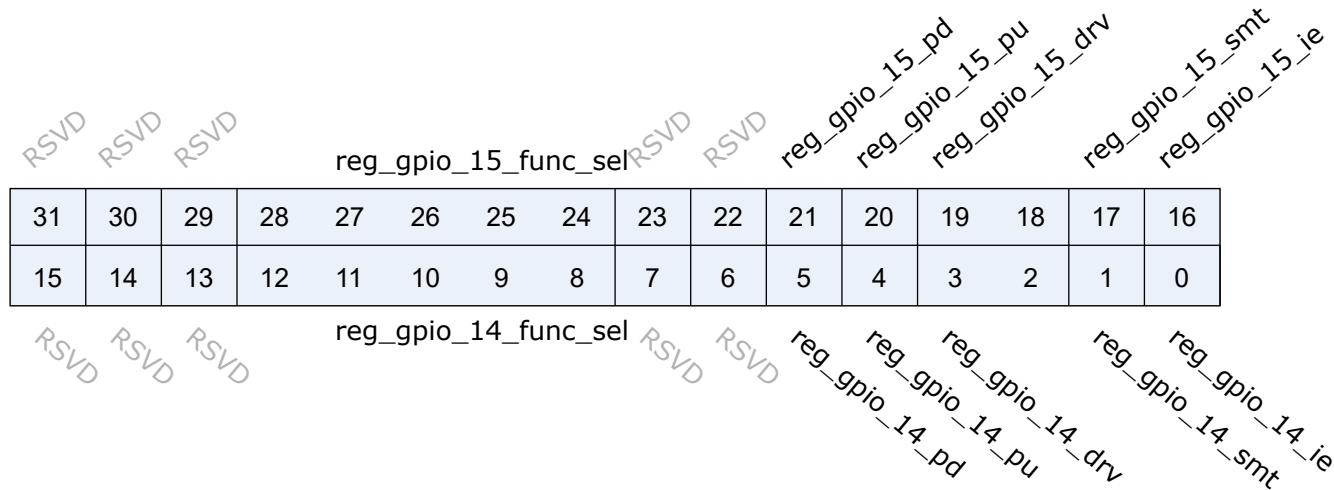
Address: 0x40000118



Bits	Name	Type	Reset	Description
31:29	RSVD			
28:24	reg_gpio_13_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)
23:22	RSVD			
21	reg_gpio_13_pd	r/w	0	GPIO Pull Down Control
20	reg_gpio_13_pu	r/w	0	GPIO Pull Up Control
19:18	reg_gpio_13_drv	r/w	0	GPIO Driving Control
17	reg_gpio_13_smt	r/w	1	GPIO SMT Control(Useless, IE depend on reg_aon_pad_ie_smt[4] : 0x4000F014[12])
16	reg_gpio_13_ie	r/w	1	GPIO Input Enable (Useless, IE depend on reg_aon_pad_ie_smt[4] : 0x4000F014[12])
15:13	RSVD			
12:8	reg_gpio_12_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)
7:6	RSVD			
5	reg_gpio_12_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_12_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_12_drv	r/w	0	GPIO Driving Control
1	reg_gpio_12_smt	r/w	1	GPIO SMT Control(Useless, IE depend on reg_aon_pad_ie_smt[3] : 0x4000F014[11])
0	reg_gpio_12_ie	r/w	1	GPIO Input Enable (Useless, IE depend on reg_aon_pad_ie_smt[3] : 0x4000F014[11])

3.3.8 GPIO_CFGCTL7

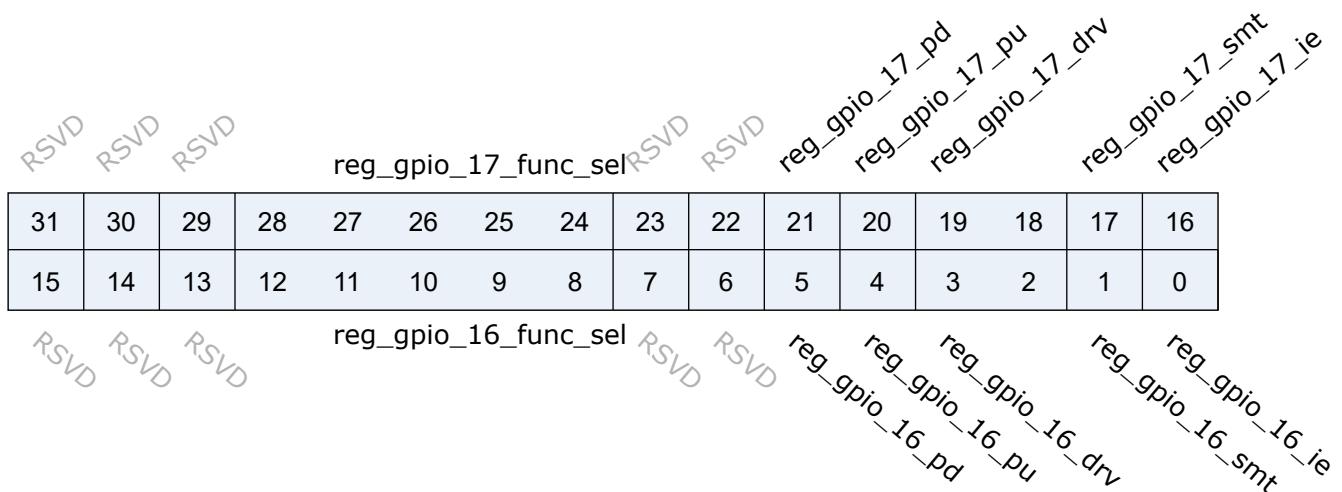
Address: 0x40000011c



Bits	Name	Type	Reset	Description
31:29	RSVD			
28:24	reg_gpio_15_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)
23:22	RSVD			
21	reg_gpio_15_pd	r/w	0	GPIO Pull Down Control
20	reg_gpio_15_pu	r/w	0	GPIO Pull Up Control
19:18	reg_gpio_15_drv	r/w	0	GPIO Driving Control
17	reg_gpio_15_smt	r/w	1	GPIO SMT Control
16	reg_gpio_15_ie	r/w	1	GPIO Input Enable
15:13	RSVD			
12:8	reg_gpio_14_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)
7:6	RSVD			
5	reg_gpio_14_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_14_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_14_drv	r/w	0	GPIO Driving Control
1	reg_gpio_14_smt	r/w	1	GPIO SMT Control
0	reg_gpio_14_ie	r/w	1	GPIO Input Enable

3.3.9 GPIO_CFGCTL8

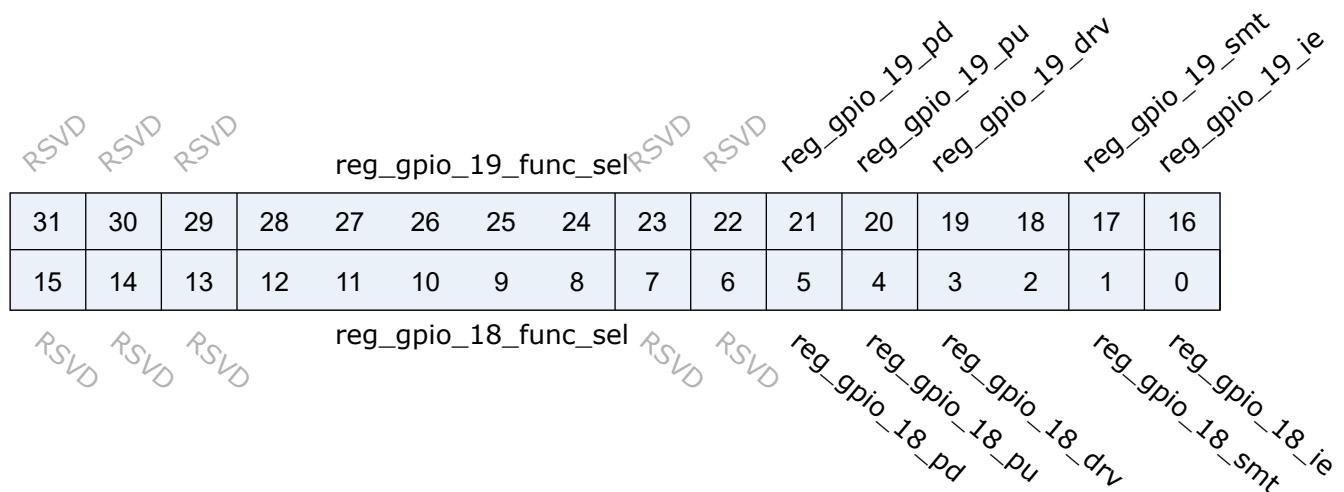
Address: 0x40000120



Bits	Name	Type	Reset	Description
31:29	RSVD			
28:24	reg_gpio_17_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)
23:22	RSVD			
21	reg_gpio_17_pd	r/w	0	GPIO Pull Down Control (Use this bit if reg_en_hw_pu_pd := 0 (0x4000F014[16]))
20	reg_gpio_17_pu	r/w	0	GPIO Pull Up Control (Use this bit if reg_en_hw_pu_pd := 0 (0x4000F014[16]))
19:18	reg_gpio_17_drv	r/w	0	GPIO Driving Control
17	reg_gpio_17_smt	r/w	1	GPIO SMT Control
16	reg_gpio_17_ie	r/w	1	GPIO Input Enable
15:13	RSVD			
12:8	reg_gpio_16_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)
7:6	RSVD			
5	reg_gpio_16_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_16_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_16_drv	r/w	0	GPIO Driving Control
1	reg_gpio_16_smt	r/w	1	GPIO SMT Control
0	reg_gpio_16_ie	r/w	1	GPIO Input Enable

3.3.10 GPIO_CFGCTL9

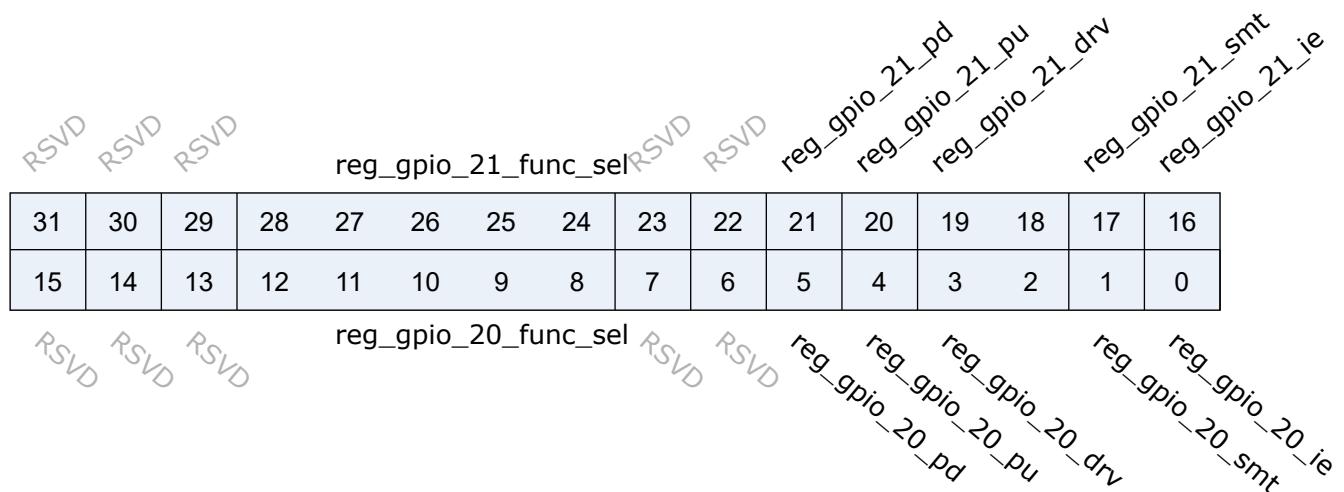
Address: 0x40000124



Bits	Name	Type	Reset	Description
31:29	RSVD			
28:24	reg_gpio_19_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)
23:22	RSVD			
21	reg_gpio_19_pd	r/w	0	GPIO Pull Down Control
20	reg_gpio_19_pu	r/w	0	GPIO Pull Up Control
19:18	reg_gpio_19_drv	r/w	0	GPIO Driving Control
17	reg_gpio_19_smt	r/w	1	GPIO SMT Control
16	reg_gpio_19_ie	r/w	1	GPIO Input Enable
15:13	RSVD			
12:8	reg_gpio_18_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)
7:6	RSVD			
5	reg_gpio_18_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_18_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_18_drv	r/w	0	GPIO Driving Control
1	reg_gpio_18_smt	r/w	1	GPIO SMT Control
0	reg_gpio_18_ie	r/w	1	GPIO Input Enable

3.3.11 GPIO_CFGCTL10

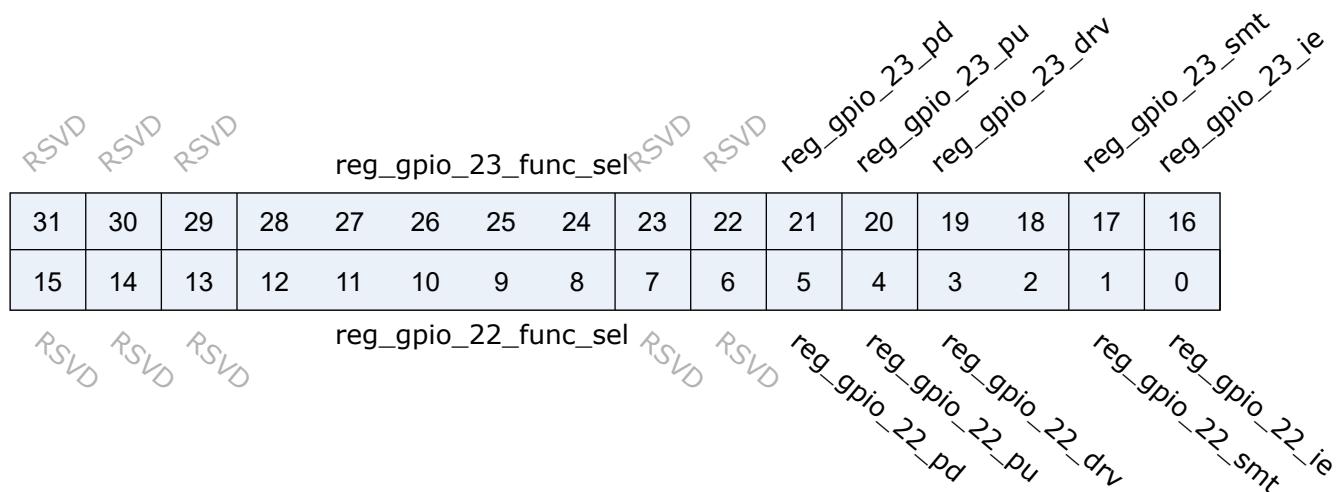
Address: 0x40000128



Bits	Name	Type	Reset	Description
31:29	RSVD			
28:24	reg_gpio_21_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)
23:22	RSVD			
21	reg_gpio_21_pd	r/w	0	GPIO Pull Down Control
20	reg_gpio_21_pu	r/w	0	GPIO Pull Up Control
19:18	reg_gpio_21_drv	r/w	0	GPIO Driving Control
17	reg_gpio_21_smt	r/w	1	GPIO SMT Control
16	reg_gpio_21_ie	r/w	1	GPIO Input Enable
15:13	RSVD			
12:8	reg_gpio_20_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)
7:6	RSVD			
5	reg_gpio_20_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_20_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_20_drv	r/w	0	GPIO Driving Control
1	reg_gpio_20_smt	r/w	1	GPIO SMT Control
0	reg_gpio_20_ie	r/w	1	GPIO Input Enable

3.3.12 GPIO_CFGCTL11

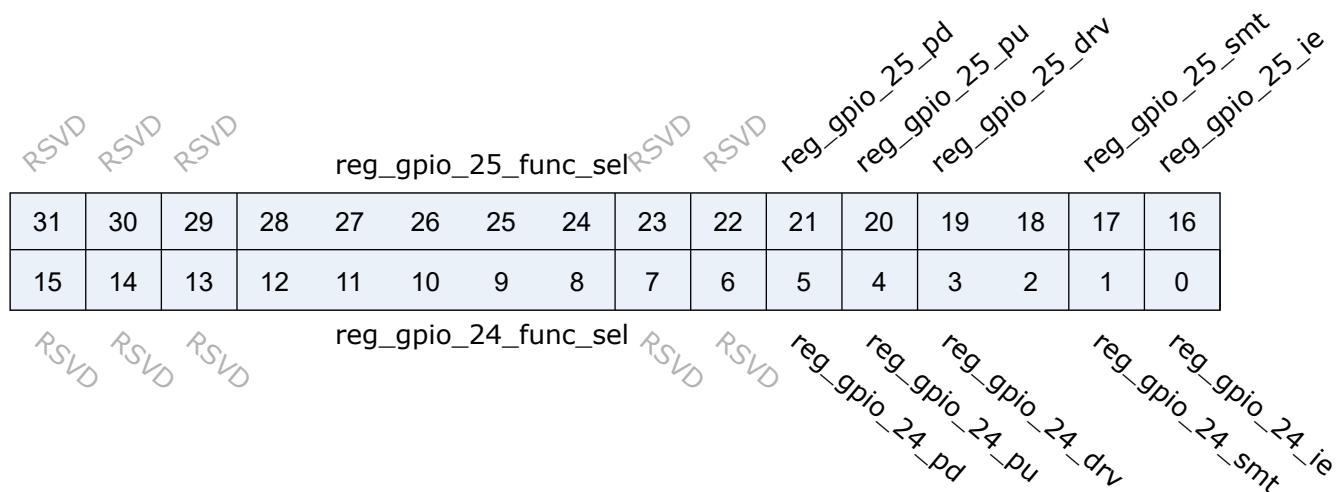
Address: 0x4000012c



Bits	Name	Type	Reset	Description
31:29	RSVD			
28:24	reg_gpio_23_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)
23:22	RSVD			
21	reg_gpio_23_pd	r/w	0	GPIO Pull Down Control
20	reg_gpio_23_pu	r/w	0	GPIO Pull Up Control
19:18	reg_gpio_23_drv	r/w	0	GPIO Driving Control
17	reg_gpio_23_smt	r/w	1	GPIO SMT Control
16	reg_gpio_23_ie	r/w	1	GPIO Input Enable
15:13	RSVD			
12:8	reg_gpio_22_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)
7:6	RSVD			
5	reg_gpio_22_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_22_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_22_drv	r/w	0	GPIO Driving Control
1	reg_gpio_22_smt	r/w	1	GPIO SMT Control
0	reg_gpio_22_ie	r/w	1	GPIO Input Enable

3.3.13 GPIO_CFGCTL12

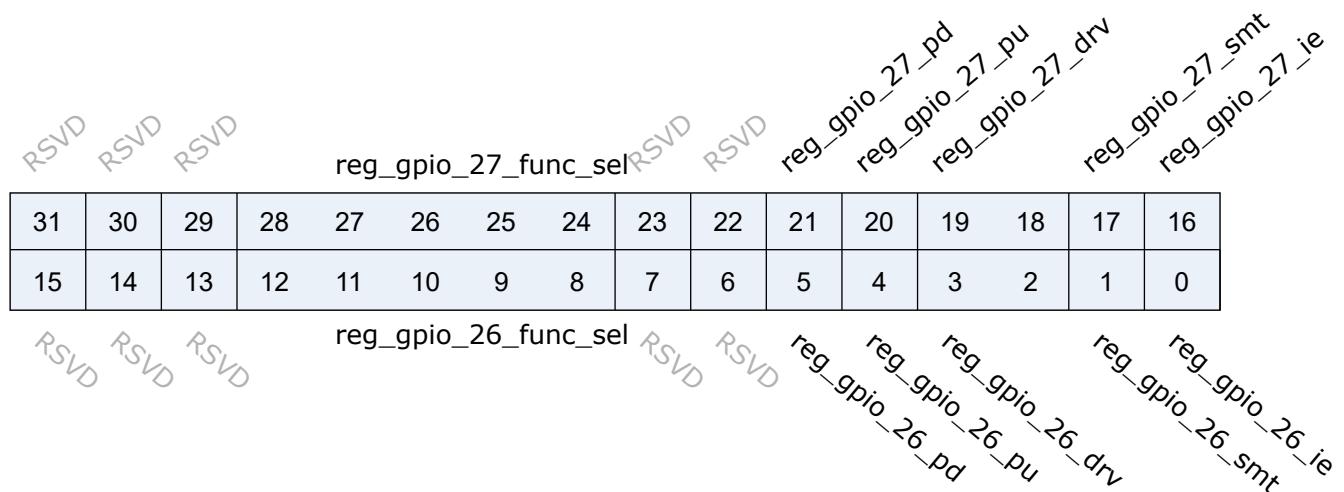
Address: 0x40000130



Bits	Name	Type	Reset	Description
31:29	RSVD			
28:24	reg_gpio_25_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)
23:22	RSVD			
21	reg_gpio_25_pd	r/w	0	GPIO Pull Down Control
20	reg_gpio_25_pu	r/w	0	GPIO Pull Up Control
19:18	reg_gpio_25_drv	r/w	0	GPIO Driving Control
17	reg_gpio_25_smt	r/w	1	GPIO SMT Control
16	reg_gpio_25_ie	r/w	1	GPIO Input Enable
15:13	RSVD			
12:8	reg_gpio_24_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)
7:6	RSVD			
5	reg_gpio_24_pd	r/w	1	GPIO Pull Down Control
4	reg_gpio_24_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_24_drv	r/w	0	GPIO Driving Control
1	reg_gpio_24_smt	r/w	1	GPIO SMT Control
0	reg_gpio_24_ie	r/w	1	GPIO Input Enable

3.3.14 GPIO_CFGCTL13

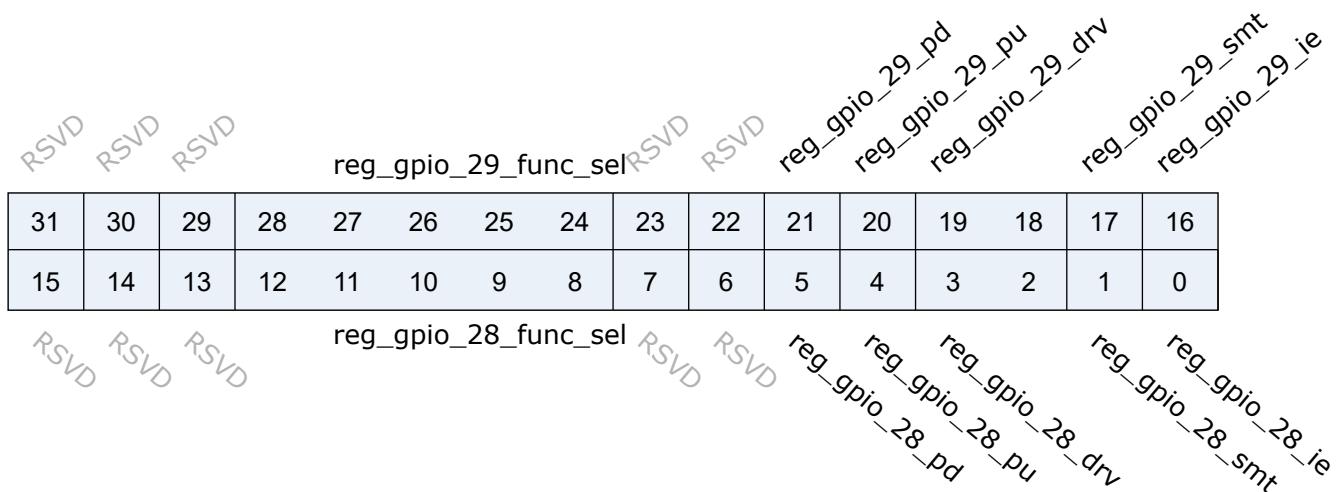
Address: 0x40000134



Bits	Name	Type	Reset	Description
31:29	RSVD			
28:24	reg_gpio_27_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)
23:22	RSVD			
21	reg_gpio_27_pd	r/w	0	GPIO Pull Down Control
20	reg_gpio_27_pu	r/w	0	GPIO Pull Up Control
19:18	reg_gpio_27_drv	r/w	0	GPIO Driving Control
17	reg_gpio_27_smt	r/w	1	GPIO SMT Control
16	reg_gpio_27_ie	r/w	1	GPIO Input Enable
15:13	RSVD			
12:8	reg_gpio_26_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)
7:6	RSVD			
5	reg_gpio_26_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_26_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_26_drv	r/w	0	GPIO Driving Control
1	reg_gpio_26_smt	r/w	1	GPIO SMT Control
0	reg_gpio_26_ie	r/w	1	GPIO Input Enable

3.3.15 GPIO_CFGCTL14

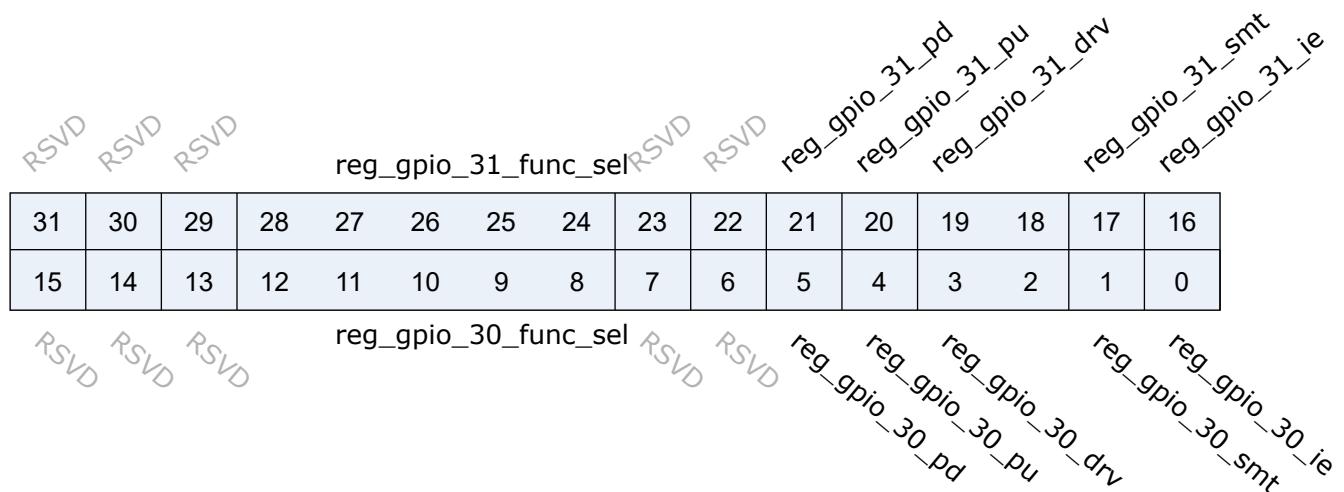
Address: 0x40000138



Bits	Name	Type	Reset	Description
31:29	RSVD			
28:24	reg_gpio_29_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)
23:22	RSVD			
21	reg_gpio_29_pd	r/w	0	GPIO Pull Down Control
20	reg_gpio_29_pu	r/w	0	GPIO Pull Up Control
19:18	reg_gpio_29_drv	r/w	0	GPIO Driving Control
17	reg_gpio_29_smt	r/w	1	GPIO SMT Control
16	reg_gpio_29_ie	r/w	1	GPIO Input Enable
15:13	RSVD			
12:8	reg_gpio_28_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)
7:6	RSVD			
5	reg_gpio_28_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_28_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_28_drv	r/w	0	GPIO Driving Control
1	reg_gpio_28_smt	r/w	1	GPIO SMT Control
0	reg_gpio_28_ie	r/w	1	GPIO Input Enable

3.3.16 GPIO_CFGCTL15

Address: 0x4000013c



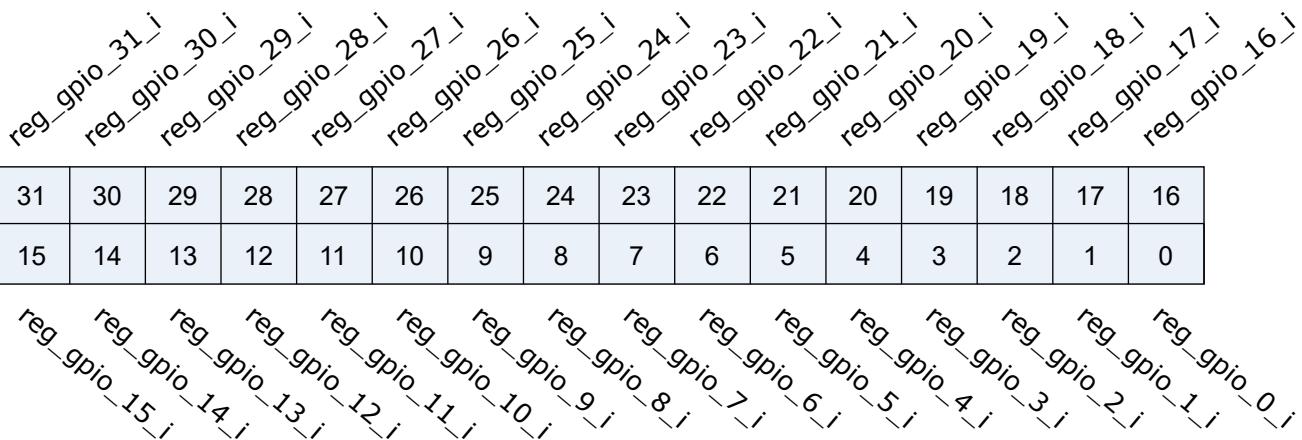
Bits	Name	Type	Reset	Description
31:29	RSVD			
28:24	reg_gpio_31_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)
23:22	RSVD			
21	reg_gpio_31_pd	r/w	0	GPIO Pull Down Control
20	reg_gpio_31_pu	r/w	0	GPIO Pull Up Control
19:18	reg_gpio_31_drv	r/w	0	GPIO Driving Control
17	reg_gpio_31_smt	r/w	1	GPIO SMT Control
16	reg_gpio_31_ie	r/w	1	GPIO Input Enable
15:13	RSVD			
12:8	reg_gpio_30_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)
7:6	RSVD			
5	reg_gpio_30_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_30_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_30_drv	r/w	0	GPIO Driving Control
1	reg_gpio_30_smt	r/w	1	GPIO SMT Control
0	reg_gpio_30_ie	r/w	1	GPIO Input Enable
-1:22	RSVD			
21	reg_gpio_33_pd	r/w	0	GPIO Pull Down Control
20	reg_gpio_33_pu	r/w	0	GPIO Pull Up Control

Bits	Name	Type	Reset	Description
19:18	reg_gpio_33_drv	r/w	0	GPIO Driving Control
17	reg_gpio_33_smt	r/w	1	GPIO SMT Control
16	reg_gpio_33_ie	r/w	1	GPIO Input Enable
15:6	RSVD			
5	reg_gpio_32_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_32_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_32_drv	r/w	0	GPIO Driving Control
1	reg_gpio_32_smt	r/w	1	GPIO SMT Control
0	reg_gpio_32_ie	r/w	1	GPIO Input Enable
-1:22	RSVD			
21	reg_gpio_35_pd	r/w	0	GPIO Pull Down Control
20	reg_gpio_35_pu	r/w	0	GPIO Pull Up Control
19:18	reg_gpio_35_drv	r/w	0	GPIO Driving Control
17	reg_gpio_35_smt	r/w	1	GPIO SMT Control
16	reg_gpio_35_ie	r/w	1	GPIO Input Enable
15:6	RSVD			
5	reg_gpio_34_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_34_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_34_drv	r/w	0	GPIO Driving Control
1	reg_gpio_34_smt	r/w	1	GPIO SMT Control
0	reg_gpio_34_ie	r/w	1	GPIO Input Enable
-1:22	RSVD			
21	reg_gpio_37_pd	r/w	0	GPIO Pull Down Control
20	reg_gpio_37_pu	r/w	0	GPIO Pull Up Control
19:18	reg_gpio_37_drv	r/w	0	GPIO Driving Control
17	reg_gpio_37_smt	r/w	1	GPIO SMT Control
16	reg_gpio_37_ie	r/w	1	GPIO Input Enable
15:6	RSVD			
5	reg_gpio_36_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_36_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_36_drv	r/w	0	GPIO Driving Control

Bits	Name	Type	Reset	Description
1	reg_gpio_36_smt	r/w	1	GPIO SMT Control
0	reg_gpio_36_ie	r/w	1	GPIO Input Enable

3.3.17 GPIO_CFGCTL30

Address: 0x40000180

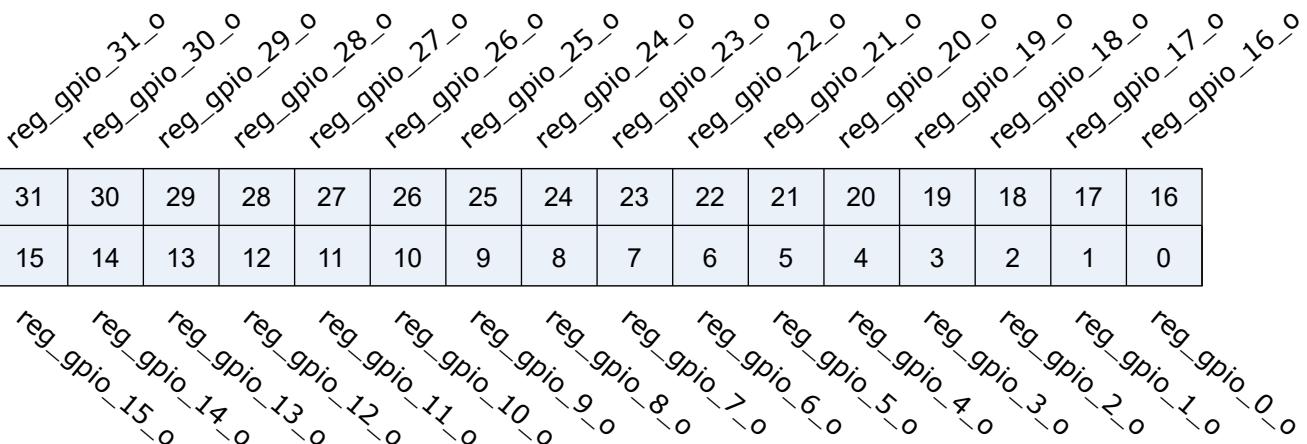


Bits	Name	Type	Reset	Description
31	reg_gpio_31_i	r	0	Register Controlled GPIO Input value
30	reg_gpio_30_i	r	0	Register Controlled GPIO Input value
29	reg_gpio_29_i	r	0	Register Controlled GPIO Input value
28	reg_gpio_28_i	r	0	Register Controlled GPIO Input value
27	reg_gpio_27_i	r	0	Register Controlled GPIO Input value
26	reg_gpio_26_i	r	0	Register Controlled GPIO Input value
25	reg_gpio_25_i	r	0	Register Controlled GPIO Input value
24	reg_gpio_24_i	r	0	Register Controlled GPIO Input value
23	reg_gpio_23_i	r	0	Register Controlled GPIO Input value
22	reg_gpio_22_i	r	0	Register Controlled GPIO Input value
21	reg_gpio_21_i	r	0	Register Controlled GPIO Input value
20	reg_gpio_20_i	r	0	Register Controlled GPIO Input value
19	reg_gpio_19_i	r	0	Register Controlled GPIO Input value
18	reg_gpio_18_i	r	0	Register Controlled GPIO Input value
17	reg_gpio_17_i	r	0	Register Controlled GPIO Input value

Bits	Name	Type	Reset	Description
16	reg_gpio_16_i	r	0	Register Controlled GPIO Input value
15	reg_gpio_15_i	r	0	Register Controlled GPIO Input value
14	reg_gpio_14_i	r	0	Register Controlled GPIO Input value
13	reg_gpio_13_i	r	0	Register Controlled GPIO Input value
12	reg_gpio_12_i	r	0	Register Controlled GPIO Input value
11	reg_gpio_11_i	r	0	Register Controlled GPIO Input value
10	reg_gpio_10_i	r	0	Register Controlled GPIO Input value
9	reg_gpio_9_i	r	0	Register Controlled GPIO Input value
8	reg_gpio_8_i	r	0	Register Controlled GPIO Input value
7	reg_gpio_7_i	r	0	Register Controlled GPIO Input value
6	reg_gpio_6_i	r	0	Register Controlled GPIO Input value
5	reg_gpio_5_i	r	0	Register Controlled GPIO Input value
4	reg_gpio_4_i	r	0	Register Controlled GPIO Input value
3	reg_gpio_3_i	r	0	Register Controlled GPIO Input value
2	reg_gpio_2_i	r	0	Register Controlled GPIO Input value
1	reg_gpio_1_i	r	0	Register Controlled GPIO Input value
0	reg_gpio_0_i	r	0	Register Controlled GPIO Input value

3.3.18 GPIO_CFGCTL32

Address: 0x40000188

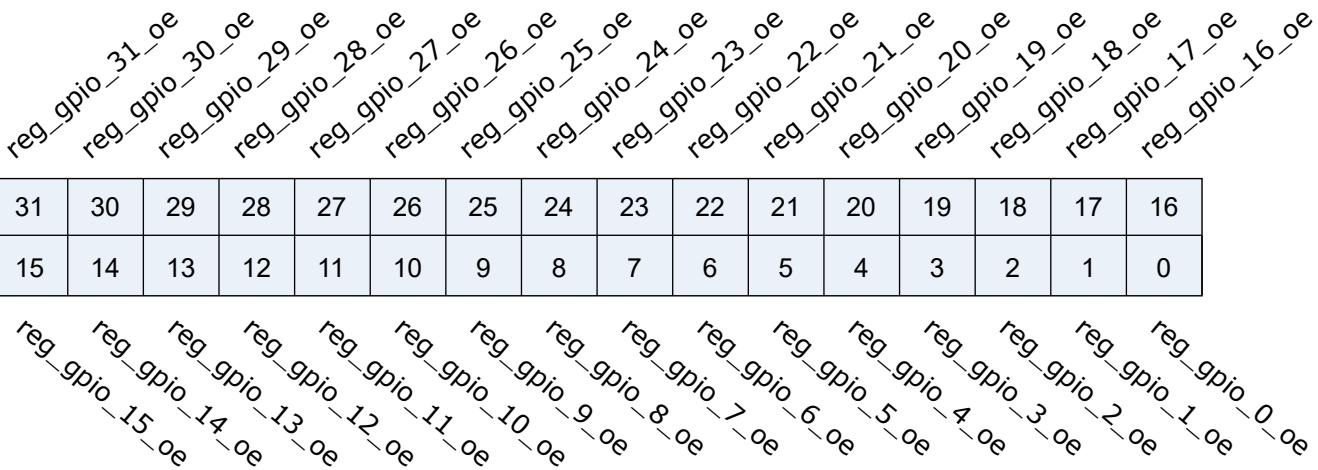


Bits	Name	Type	Reset	Description
31	reg_gpio_31_o	r/w	0	Register Controlled GPIO Output Value
30	reg_gpio_30_o	r/w	0	Register Controlled GPIO Output Value
29	reg_gpio_29_o	r/w	0	Register Controlled GPIO Output Value
28	reg_gpio_28_o	r/w	0	Register Controlled GPIO Output Value
27	reg_gpio_27_o	r/w	0	Register Controlled GPIO Output Value
26	reg_gpio_26_o	r/w	0	Register Controlled GPIO Output Value
25	reg_gpio_25_o	r/w	0	Register Controlled GPIO Output Value
24	reg_gpio_24_o	r/w	0	Register Controlled GPIO Output Value
23	reg_gpio_23_o	r/w	0	Register Controlled GPIO Output Value
22	reg_gpio_22_o	r/w	0	Register Controlled GPIO Output Value
21	reg_gpio_21_o	r/w	0	Register Controlled GPIO Output Value
20	reg_gpio_20_o	r/w	0	Register Controlled GPIO Output Value
19	reg_gpio_19_o	r/w	0	Register Controlled GPIO Output Value
18	reg_gpio_18_o	r/w	0	Register Controlled GPIO Output Value
17	reg_gpio_17_o	r/w	0	Register Controlled GPIO Output Value
16	reg_gpio_16_o	r/w	0	Register Controlled GPIO Output Value
15	reg_gpio_15_o	r/w	0	Register Controlled GPIO Output Value
14	reg_gpio_14_o	r/w	0	Register Controlled GPIO Output Value
13	reg_gpio_13_o	r/w	0	Register Controlled GPIO Output Value
12	reg_gpio_12_o	r/w	0	Register Controlled GPIO Output Value
11	reg_gpio_11_o	r/w	0	Register Controlled GPIO Output Value
10	reg_gpio_10_o	r/w	0	Register Controlled GPIO Output Value
9	reg_gpio_9_o	r/w	0	Register Controlled GPIO Output Value
8	reg_gpio_8_o	r/w	0	Register Controlled GPIO Output Value
7	reg_gpio_7_o	r/w	0	Register Controlled GPIO Output Value
6	reg_gpio_6_o	r/w	0	Register Controlled GPIO Output Value
5	reg_gpio_5_o	r/w	0	Register Controlled GPIO Output Value
4	reg_gpio_4_o	r/w	0	Register Controlled GPIO Output Value
3	reg_gpio_3_o	r/w	0	Register Controlled GPIO Output Value
2	reg_gpio_2_o	r/w	0	Register Controlled GPIO Output Value
1	reg_gpio_1_o	r/w	0	Register Controlled GPIO Output Value

Bits	Name	Type	Reset	Description
0	reg_gpio_0_o	r/w	0	Register Controlled GPIO Output Value

3.3.19 GPIO_CFGCTL34

Address: 0x40000190



Bits	Name	Type	Reset	Description
31	reg_gpio_31_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
30	reg_gpio_30_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
29	reg_gpio_29_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
28	reg_gpio_28_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
27	reg_gpio_27_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
26	reg_gpio_26_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
25	reg_gpio_25_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
24	reg_gpio_24_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
23	reg_gpio_23_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)

Bits	Name	Type	Reset	Description
22	reg_gpio_22_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
21	reg_gpio_21_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
20	reg_gpio_20_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
19	reg_gpio_19_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
18	reg_gpio_18_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
17	reg_gpio_17_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
16	reg_gpio_16_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
15	reg_gpio_15_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
14	reg_gpio_14_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
13	reg_gpio_13_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
12	reg_gpio_12_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
11	reg_gpio_11_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
10	reg_gpio_10_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
9	reg_gpio_9_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
8	reg_gpio_8_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
7	reg_gpio_7_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
6	reg_gpio_6_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
5	reg_gpio_5_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)

Bits	Name	Type	Reset	Description
4	reg_gpio_4_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
3	reg_gpio_3_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
2	reg_gpio_2_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
1	reg_gpio_1_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
0	reg_gpio_0_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)

3.3.20 GPIO_CFGCTL35

Address: 0x40000194

RSVD																
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	

RSVD															
------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------

Bits	Name	Type	Reset	Description
31:0	reg_gpio_int_mask1	r/w	32'hFFFFFFF	Reg_gpio_int_mask[31:0]

3.3.21 GPIO_INT_STAT1

Address: 0x400001a8

gpio_int_stat1

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

gpio_int_stat1

Bits	Name	Type	Reset	Description
31:0	gpio_int_stat1	r	0	gpio_int_stat[31:0]

3.3.22 GPIO_INT_CLR1

Address: 0x400001b0

reg_gpio_int_clr1

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

reg_gpio_int_clr1

Bits	Name	Type	Reset	Description
31:0	reg_gpio_int_clr1	r/w	0	reg_gpio_int_clr[31:0]

3.3.23 GPIO_INT_MODE_SET1

Address: 0x400001c0

RSVD RSVD

reg_gpio_int_mode_set1

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

reg_gpio_int_mode_set1

Bits	Name	Type	Reset	Description
31:30	RSVD			
29:0	reg_gpio_int_mode_set1	r/w	0	reg_gpio9_int_mode[2:0], reg_gpio1_int_mode[2:0], reg_gpio0_int_mode

3.3.24 GPIO_INT_MODE_SET2

Address: 0x400001c4



RSVD RSVD

reg_gpio_int_mode_set2

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

reg_gpio_int_mode_set2

Bits	Name	Type	Reset	Description
31:30	RSVD			
29:0	reg_gpio_int_mode_set2	r/w	0	reg_gpio19_int_mode[2:0], reg_gpio11_int_mode[2:0],reg_gpio10_int_mode

3.3.25 GPIO_INT_MODE_SET3

Address: 0x400001c8

RSVD RSVD

reg_gpio_int_mode_set3

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

reg_gpio_int_mode_set3

Bits	Name	Type	Reset	Description
31:30	RSVD			
29:0	reg_gpio_int_mode_set3	r/w	0	reg_gpio29_int_mode[2:0], reg_gpio21_int_mode[2:0],reg_gpio20_int_mode

3.3.26 GPIO INT MODE SET4

Address: 0x400001cc

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

reg gpio int mode set4

Bits	Name	Type	Reset	Description
31:6	RSVD			
5:0	reg_gpio_int_mode_set4	r/w	0	reg_gpio31_int_mode[2:0],reg_gpio30_int_mode

3.3.27 GPIO_INT2_MASK1

Address: 0x400001d0

reg_gpio_int2_mask1

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

reg_gpio_int2_mask1

Bits	Name	Type	Reset	Description
31:0	reg_gpio_int2_mask1	r/w	32'hFFFFFFF	reg_gpio_int2_mask[31:0]

3.3.28 GPIO_INT2_STAT1

Address: 0x400001d4

gpio_int2_stat1

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

gpio_int2_stat1

Bits	Name	Type	Reset	Description
31:0	gpio_int2_stat1	r	0	gpio_int2_stat[31:0]

3.3.29 GPIO_INT2_CLR1

Address: 0x400001d8

reg_gpio_int2_clr1

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

reg_gpio_int2_clr1

Bits	Name	Type	Reset	Description
31:0	reg_gpio_int2_clr1	r/w	0	reg_gpio_int2_clr[31:0]

3.3.30 GPIO_INT2_MODE_SET1

Address: 0x400001dc

reg_gpio_int2_mode_set1																
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	
reg_gpio_int2_mode_set1																

Bits	Name	Type	Reset	Description
31:30	RSVD			
29:0	reg_gpio_int2_mode_set1	r/w	0	reg_gpio9_int2_mode[2:0], reg_gpio1_int2_- mode[2:0],reg_gpio0_int2_mode

3.3.31 GPIO_INT2_MODE_SET2

Address: 0x400001e0

reg_gpio_int2_mode_set2																
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	
reg_gpio_int2_mode_set2																

Bits	Name	Type	Reset	Description
31:30	RSVD			
29:0	reg_gpio_int2_mode_set2	r/w	0	reg_gpio19_int2_mode[2:0], reg_gpio11_int2_- mode[2:0],reg_gpio10_int2_mode

3.3.32 GPIO_INT2_MODE_SET3

Address: 0x400001e4

reg_gpio_int2_mode_set3															
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

reg_gpio_int2_mode_set3

Bits	Name	Type	Reset	Description
31:30	RSVD			
29:0	reg_gpio_int2_mode_set3	r/w	0	reg_gpio29_int2_mode[2:0], reg_gpio21_int2_mode[2:0], reg_gpio20_int2_mode

3.3.33 GPIO_INT2_MODE_SET4

Address: 0x400001e8

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
reg_gpio_int2_mode_set4															

Bits	Name	Type	Reset	Description
31:6	RSVD			
5:0	reg_gpio_int2_mode_set4	r/w	0	reg_gpio31_int2_mode[2:0], reg_gpio30_int2_mode
-1:12	RSVD			
11	reg_usb_use_ctrl	r/w	1'b1	1: USB XCVR use on-chip usb controller ; 0: USB XCVR use off-chip usb controller
10:0	RSVD			

4.1 ADC introduction

The chip contains a 12-bit successive approximation analog-to-digital converter (ADC), which supports 12 external analog inputs and several internal analog signal selections.

ADC supports the following four modes: single-shot single-channel conversion, continuous single-channel conversion, single-shot multi-channel conversion, and continuous multi-channel conversion mode. The conversion result is 12/14/16bits left-justified mode. The ADC has a depth of 32 FIFOs and supports multiple interrupts and DMA operations. In addition to ordinary analog signal measurement, the ADC can also be used to measure the supply voltage. In addition, the ADC can also be used for temperature detection by measuring the internal/external diode voltage.

4.2 ADC main features

- High performance
 - 12/14/16bits conversion result output
 - ADC maximum working clock is 2MHZ
 - 2.0V,3.2V optional reference voltage
 - DMA support
 - Supports four modes: single-shot single-channel conversion, continuous single-channel conversion, single-shot multi-channel conversion, and continuous multi-channel conversion mode
 - Two input modes: single-ended and differential
 - Support jitter compensation
 - User can set conversion result offset value
- Analog channels
 - 12 external analog channels

- 2 DAC internal channels
- 1 VBAT / 2 channel
- 1 TSEN channel

4.3 ADC functional description

The basic block diagram of the ADC is shown below.

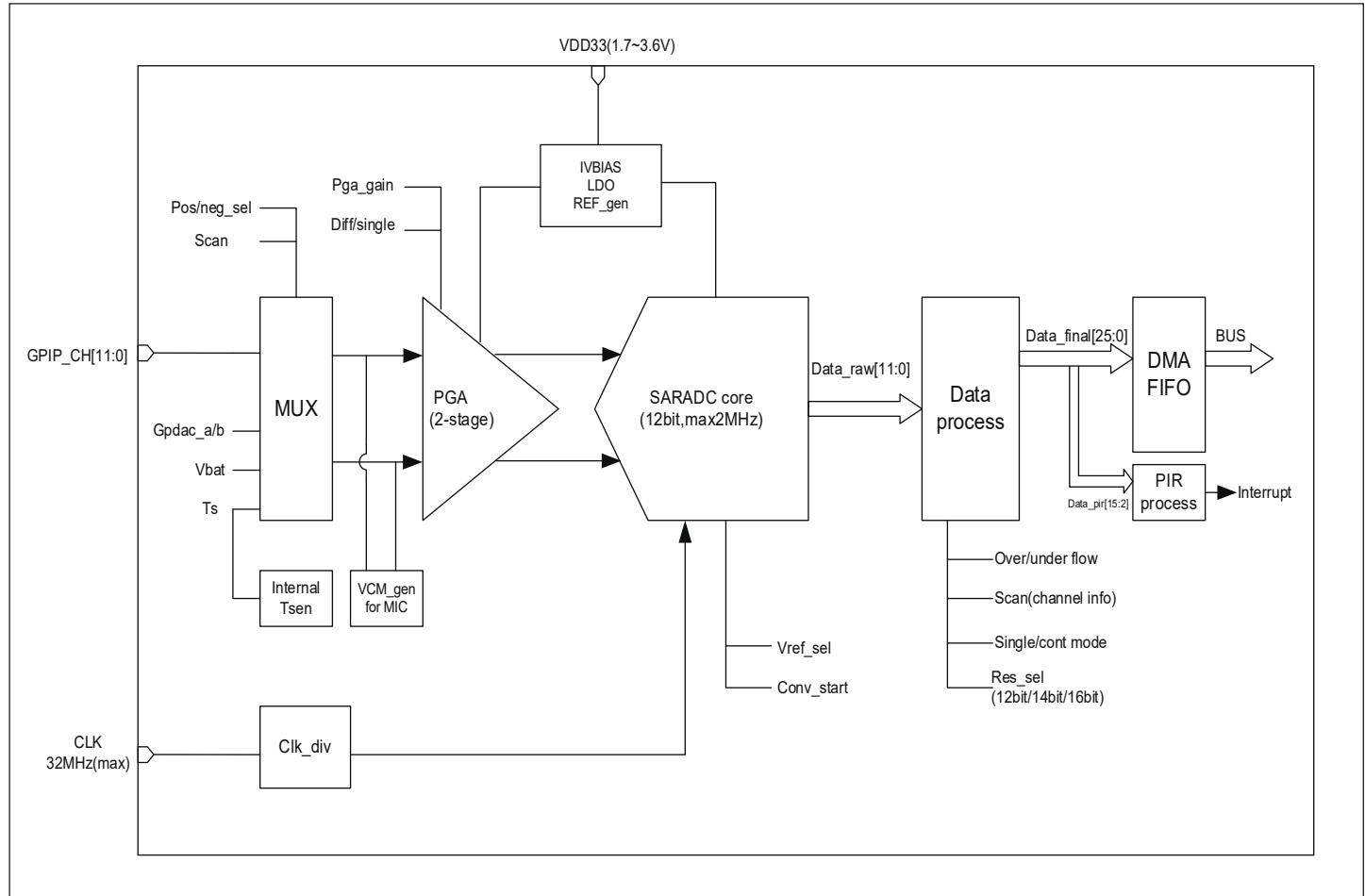


Fig. 4.1: ADC block diagram

The ADC consists of five parts: front-end input channel selector, program-controlled amplifier, ADC sampling module, data processing module, and FIFO.

The input channel selector is used to select the channel to be sampled. It contains both external analog signals and internal analog signals. The program-controlled amplifier is used to further process the input signal. It can be set according to the characteristics of the input signal, such as DC and AC. In order to get more accurate conversion values.

The ADC sampling module is the most important function module. It obtains the conversion from analog signals to digital signals through successive comparisons. The conversion result is 12bit. The data processing module is

responsible for further processing the conversion results, including adding channel information. The resulting data is pushed into the FIFO.

4.3.1 ADC pins and internal signals

Table 4.1: ADC internal signals

internal signals	Signal type	Description
VBAT/2	Input	Voltage signal divided from the power pin
TSEN	Input	Internal temperature sensor output voltage
VREF	Input	Internal analog module reference voltage
DACOUTA	Input	DAC module output
DACOUTB	Input	DAC module output

Table 4.2: ADC external pins

external pins	Signal type	Description
VDDA	Input	Analog power supply and positive reference voltage for the ADC
VSSA	Input	Ground for analog power supply
ADC_CHX	Input	12 analog input channels

4.3.2 ADC channel

The channels that can be selected by the ADC include the input signals of external analog pins and the optional signals inside the chip:

- ADC CH0
- ADC CH1
- ADC CH2
- ADC CH3
- ADC CH4
- ADC CH5
- ADC CH6
- ADC CH7
- ADC CH8

- ADC CH9
- ADC CH10
- ADC CH11
- VSSA
- DAC OUTA
- DAC OUTB
- VBAT/2
- TSEN
- VREF
- GND

It should be noted that if VBAT/2 or TSEN is selected as the input signal to be acquired, gpadc_vbat_en or gpadc_ts_en needs to be set.

The ADC module can support single-ended input or differential input. If it is single-ended input mode, the negative input channel needs to select GND.

4.3.3 ADC clock

The working clock source of the ADC module is shown in the following figure:

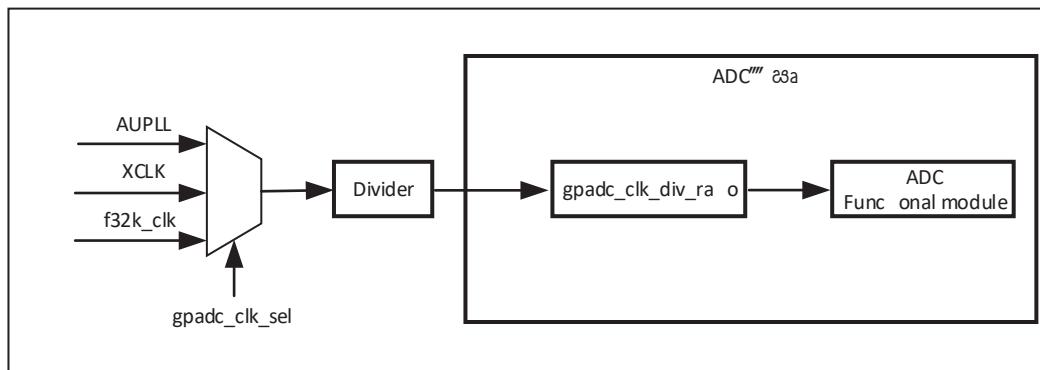


Fig. 4.2: ADC Clock

The ADC clock source can select Audio PLL, XCLK or f32k. The clock source selection is set in the GLB module. While selecting, the clock source can be divided by the frequency divider.

In general voltage measurement applications, users can choose XCLK as the clock source. In audio applications, users can use AUPLL to generate common sampling clocks such as 8KHZ and 44.1KHZ. f32k is a low-frequency clock, which provides a wake-up clock when the MCU sleeps.

Inside the ADC module, a clock divider is provided, which can divide the input clock by 1/4/8/12/16/20/24/32. Users can adjust the ADC clock source and various frequency division coefficients according to actual sampling requirements. Note that the maximum input clock of ADC is 2MHZ.

The width of the gpadc_32m_clk_div divider register is 6bits, the maximum divider is 64, and the divider formula is $f_{out} = f_{source} / (gpadc_32m_clk_div + 1)$.

The gpadc_clk_div_ratio divider register is located inside the ADC module, with a width of 3bits, and its divider value is defined as follows:

```
ADC_CLK_DIV_1,          /*!< ADC clock:on 32M clock is 32M */
ADC_CLK_DIV_4,          /*!< ADC clock:on 32M clock is 8M */
ADC_CLK_DIV_8,          /*!< ADC clock:on 32M clock is 4M */
ADC_CLK_DIV_12,         /*!< ADC clock:on 32M clock is 2.666M */
ADC_CLK_DIV_16,         /*!< ADC clock:on 32M clock is 2M */
ADC_CLK_DIV_20,         /*!< ADC clock:on 32M clock is 1.6M */
ADC_CLK_DIV_24,         /*!< ADC clock:on 32M clock is 1.333M */
ADC_CLK_DIV_32,         /*!< ADC clock:on 32M clock is 1M */
```

If the user wants to adjust the ADC input clock, there are four ways.

1. Switch the clock source, XTAL defaults to 32MHZ, Audio PLL (can be configured to 11.288MHZ or 11.2896MHZ).
2. Use a frequency divider with a length of 6BITS in the clock module.
3. Using the frequency divider in the ADC module, the optional frequency division is 1/4/8/12/16/20/24/32.
4. By configuring the gpadc_res_sel register, change the value of OSR to achieve the frequency effect. If OSR=256, the actual equivalent ADC input clock is divided by 256.

Assuming that the clock source selection is Audio PLL=11.2896MHZ, the GLB frequency division selection configuration is 1, the ADC internal frequency divider is selected ADC_CLK_DIV_4, OSR=128, then the final clock output is $f_{out} = 11289600 / (1 + 1) / 4/128 = 11025\text{Hz}$

4.3.4 ADC conversion mode

The ADC supports two conversion modes: single-channel conversion mode and scan mode.

In single-channel conversion mode, the user needs to select the positive input channel through gpadc_pos_sel, select the negative input channel through gpadc_neg_sel, and set the gpadc_cont_conv_en control bit to 0, which means single-channel conversion, and then set the gpadc_conv_start control bit to start the conversion.

In scan conversion mode, the gpadc_cont_conv_en control bit needs to be set to 1, and the number of conversion channels set by the ADC according to the gpadc_scan_length control bit. According to the channel order set by the

gpadc_reg_scn_posX (X = 1, 2) and gpadc_reg_scn_negX (X = 1, 2) registers, the conversion is performed one by one, and the result of the conversion is automatically pushed into the ADC FIFO. The channels set by the gpadc_reg_scn_posX (X = 1, 2) and gpadc_reg_scn_negX (X = 1, 2) registers can be the same, which means that users can implement multiple sampling conversions on a channel.

ADC conversion results are generally placed in the FIFO. The ADC module does not provide conversion completion interrupts. Users need to set the FIFO receive data threshold interrupt based on the actual number of conversion channels. The FIFO threshold interrupt is used as the ADC conversion completion interrupt.

4.3.5 ADC consequence

The gpadc_raw_data register stores the raw result of the ADC. In single-ended mode, the data valid bit is 12bits, unsigned bit. In differential mode, the highest bit is the sign bit. The remaining 11bits represent the result of the conversion.

The gpadc_data_out register stores the ADC result. This result contains the ADC result, sign bit and channel information. The data format is as follows:

Table 4.3: ADC conversion result format

BitS	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
meaning	Positive channel number					Negative channel number					Conversion result															

bit21-bit25 of the conversion result is the positive channel number, bit16-bit20 is the negative channel number, and bit0-bit15 is the converted value.

The gpadc_res_sel control bit can set the number of bits of the conversion result, which are 12 bits, 14 bits, and 16 bits, respectively. Among them, 14 bits and 16 bits are the results obtained by multiple sampling to improve the accuracy.

The values that can be set are as follows:

- 3' b000 12bit 2MS/s, OSR=1
- 3' b001 14bit 125kS/s, OSR=16
- 3' b010 14bit 31.25kS/s, OSR=64
- 3' b011 16bit 15.625KS/s, OSR=128
- 3' b100 16bit 7.8125KS/s, OSR=256

The ADC conversion result is left-justified. When 12 bits are selected, bit15-bit4 of the conversion result is valid. When 14 bits are selected, bit15-bit2 of the conversion result is valid. When 16 bits are selected, bit15-bit0 of the conversion result is valid.

Similarly, in the differential mode, the highest is the sign, that is, when 14 bits are selected, bit15 is the sign bit, bit14-bit2 is the conversion result, and bit14 is the MSB.

In single-ended mode, there is no sign bit, that is, when 12 bits are selected, bit15-bit4 is the conversion result and bit15 is the MSB.

In actual use, the results of the ADC are generally placed in the FIFO, which is particularly important in the multi-channel scan mode. Therefore, users generally obtain conversion results from the ADC FIFO. The data format of the ADC FIFO is the same in the gpadc_data_out register.

4.3.6 ADC interrupt

The ADC module can generate interrupts when the positive sampling is saturated and the negative sampling is saturated. The respective interrupts can be masked by gpadc_pos_satur_mask, gpadc_neg_satur_mask.

When the interrupt is generated, the interrupt status can be queried by the gpadc_pos_satur, and gpadc_neg_satur registers, and the interrupt can be cleared by gpadc_pos_satur_clr and gpadc_neg_satur_clr. This function can be used to determine whether the input voltage is abnormal.

4.3.7 ADC FIFO

The ADC module has a FIFO with a depth of 32 and a data width of 26Bits. After the ADC completes the conversion, it will automatically push the result into the FIFO. The ADC's FIFO has the following status and interrupt management functions:

- FIFO Overrun interrupt
- FIFO Underrun interrupt
- FIFO threshold interrupt

When the FIFO is full, but the user does not read the value through DMA or direct access to the register, and data enters the FIFO again, the module will generate a FIFO Overrun interrupt at this time.

When the FIFO is empty, but the user still requests data from the FIFO, the module will generate a FIFO Underrun interrupt at this time.

The user can configure the FIFO threshold register gpadc_fifo_thl, select the threshold for FIFO to generate interrupts, and choose 1, 4, 8, and 16. If the number of ADC FIFOs reaches the set threshold, a threshold interrupt will be generated.

When an interrupt occurs, the interrupt flag can be cleared by the corresponding clear bit.

Using the ADC's FIFO, users can implement three modes of data acquisition: query mode, interrupt mode, and DMA mode.

Query mode

The CPU polls the length of the ADC FIFO. When the length of the FIFO is not empty, it indicates that there are valid

data in the FIFO, and the CPU can read these data from the FIFO.

Interrupt mode

Using the threshold interrupt of the FIFO, when the interrupt is generated and the number of ADC data reaches the threshold, the CPU can read the length of the ADC FIFO in the interrupt service function and read it all.

DMA mode

The user sets the dmaen control bit, which can cooperate with the DMA to complete the transfer of the converted data to the memory. When using the DMA mode, set the threshold of the number of data sent by the ADC FIFO through the fifothl. When the DMA receives the request, it will automatically follow the user settings. The specified parameters transfer the specified number of results from the FIFO to the corresponding memory.

4.3.8 ADC configuration process

Setting the ADC clock

According to the ADC conversion speed requirements, determine the working clock of the ADC, set the ADC clock source and frequency division of the GLB module, and combine with clkdvr to determine the final working module's clock frequency.

Set GPIO according to the channel used

According to the analog pin used, determine the channel number used, initialize the corresponding GPIO as an analog function. It should be noted that when setting the GPIO as an analog input, do not set the GPIO pull-up or pull-down, you need to set it to float.

Set the channel to be converted

Set the corresponding channel register according to the analog channel and conversion mode used.

For single-channel conversion, set the converted channel information in the possel and negsel registers.

For multi-channel scanning mode, set sclen, scpX and scnX according to the number of scanning channels and scanning order.

Set the data reading method

According to the way of reading data introduced by ADC FIFO, select the mode to use and set the corresponding register. If you use DMA, you also need to configure a channel of DMA to cooperate with the ADC FIFO to complete the data transfer.

Start conversion

Finally set ressel to select the precision of the data conversion result. Finally set gben = 1 and cvst = 1 to start the ADC to start conversion.

When the conversion is complete and needs to be converted again, cvst needs to be set to 0 and then set to 1 in order to trigger the conversion again.

4.3.9 VBAT measurement

The VBAT/2 measurement is the voltage of the chip VDD33, not the voltage of an external battery such as a lithium battery. If you need to measure the voltage of a power supply head such as a lithium battery, you can divide the voltage and then input it to the ADC's GPIO analog channel. Measuring the voltage of VDD33 can reduce the use of GPIO.

The VBAT/2 voltage measured by the ADC module is after a partial pressure. The actual input voltage to the ADC module is half of VDD33, that is, $VBAT/2 = VDD33/2$. Because the voltage is divided, in order to obtain higher accuracy, it is recommended that the reference voltage of the ADC is 2V, single-ended mode is used, the positive input voltage is VBAT/2, the negative input voltage is GND, and vbaten is set to 1 to start.

After conversion, multiply the corresponding conversion result by 2 to get the VDD33 voltage.

4.3.10 TSEN measurement

The ADC can measure the internal diode or external diode voltage value, and the voltage difference between the diode and temperature is related, so by measuring the voltage of the diode, the ambient temperature can be calculated. We call it Temperature Sensor, referred to as TSEN.

The test principle of TSEN is to generate a fitted curve by measuring the voltage difference ΔV generated by two different currents on a diode with temperature.

Regardless of the measurement of the external or internal diode, the final output value is related to temperature, which can be expressed as $\Delta(ADC_out) = 7.753T + X$. When we know the voltage value, we also know the temperature T. Here X is an offset value that can be used as a standard value. Before actual use, we need to determine X. The chip manufacturer will measure $\Delta(ADC_out)$ at a standard temperature, such as 25 degrees at room temperature, before the chip leaves the factory to get X.

When the user uses it, as long as the formula $T = [\Delta(ADC_out) - X]/7.753$, the temperature T can be obtained.

When using TSEN, it is recommended to set the ADC to 16Bits mode, reduce the error by multiple sampling, and select 2V as the reference voltage to improve accuracy. Set tsen to 1 to enable the TSEN function. If the internal diode is selected, tsxten = 0. External diode, tsxten = 1, select the forward input channel according to the actual situation.

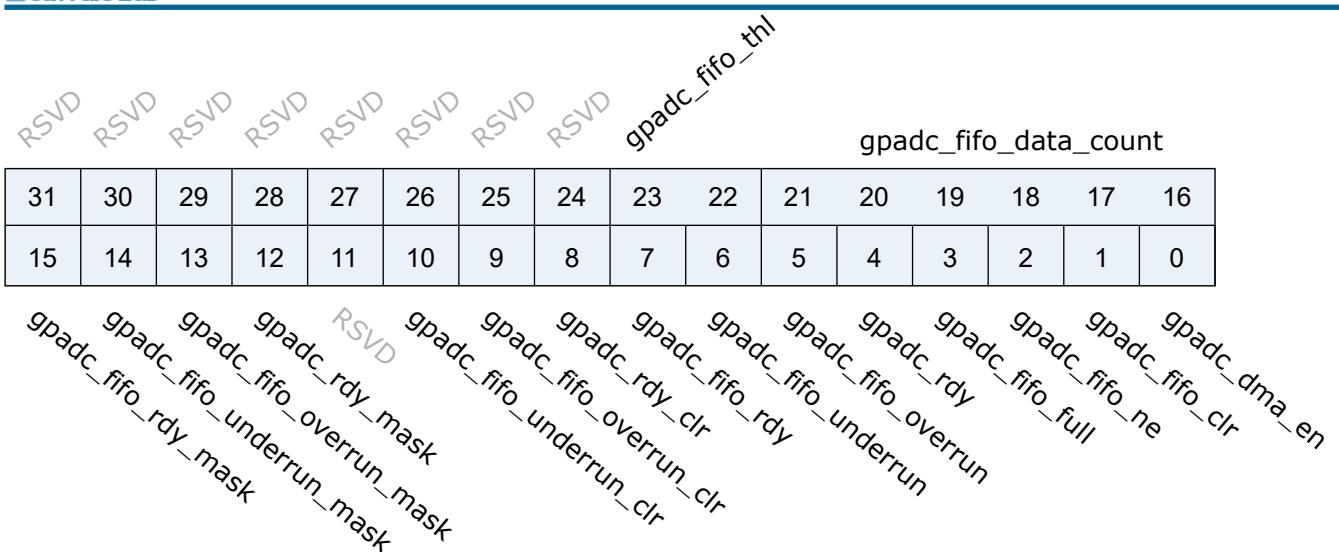
If it is an internal diode, select the TSEN channel. If it is external, select the corresponding analog GPIO channel. Select the negative input terminal as GND. After the above settings are completed, set tsdc = 0 to start the measurement and get the measurement result V0, then set tsdc = 1 to start the measurement and get the measurement result V1, $\Delta(ADC_out) = V1 - V0$, according to the formula $T = [\Delta(ADC_out) - X] / 7.753$ to obtain the temperature T.

4.4 Register description

Name	Description
gpadc_config	GPADC configuration
gpadc_dma_rdata	GPADC DMA read data
gpdac_config	GPDAC configuration
gpdac_dma_config	GPDAC dma configuration
gpdac_dma_wdata	GPDAC dma write data
gpadc_reg_cmd	GPADC register configuration 0
gpadc_reg_config1	GPADC register configuration 1
gpadc_reg_config2	GPADC register configuration 2
gpadc_reg_scn_pos1	adc converation sequence 1
gpadc_reg_scn_pos2	adc converation sequence 2
gpadc_reg_scn_neg1	adc converation sequence 3
gpadc_reg_scn_neg2	adc converation sequence 4
gpadc_reg_status	GPADC register status
gpadc_reg_isr	GPADC register operation
gpadc_reg_raw_result	GPADC register raw result
gpadc_reg_define	GPADC register define

4.4.1 gpadc_config

Address: 0x40002000

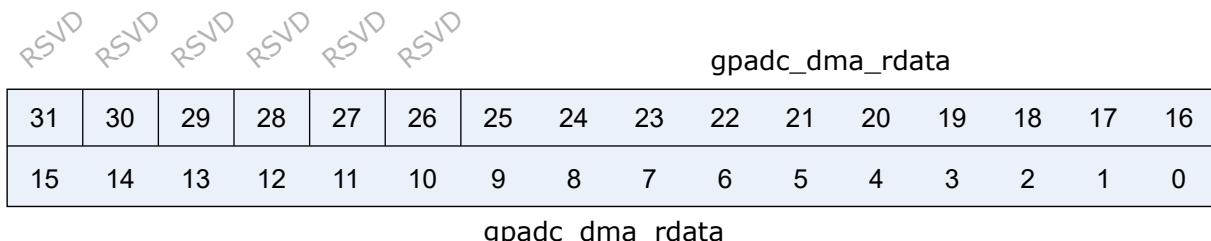


Bits	Name	Type	Reset	Description
31:24	RSVD			
23:22	gpadc_fifo_thl	r/w	2'd0	fifo threshold 2'b00: 1 data 2'b01: 4 data 2'b10: 8 data 2'b11: 16 data
21:16	gpadc_fifo_data_count	r	6'd0	fifo data number
15	gpadc_fifo_rdy_mask	r/w	1'b1	write 1 mask
14	gpadc_fifo_underrun_mask	r/w	1'b0	write 1 mask
13	gpadc_fifo_overrun_mask	r/w	1'b0	write 1 mask
12	gpadc_rdy_mask	r/w	1'b0	write 1 mask
11	RSVD			
10	gpadc_fifo_underrun_clr	w1c	1'b0	Write 1 to clear flag
9	gpadc_fifo_overrun_clr	w1c	1'b0	Write 1 to clear flag
8	gpadc_rdy_clr	w1c	1'b0	Write 1 to clear flag
7	gpadc_fifo_rdy	r	1'b0	FIFO ready interrupt flag
6	gpadc_fifo_underrun	r	1'b0	FIFO underrun interrupt flag
5	gpadc_fifo_overrun	r	1'b0	FIFO overrun interrupt flag
4	gpadc_rdy	r	1'b0	Conversion data ready interrupt flag
3	gpadc_fifo_full	r	1'b0	FIFO full flag
2	gpadc_fifo_ne	r	1'b0	FIFO not empty flag
1	gpadc_fifo_clr	w1c	1'b0	FIFO clear signal

Bits	Name	Type	Reset	Description
0	gpadc_dma_en	r/w	1'b0	GPADC DMA enable

4.4.2 gpadc_dma_rdata

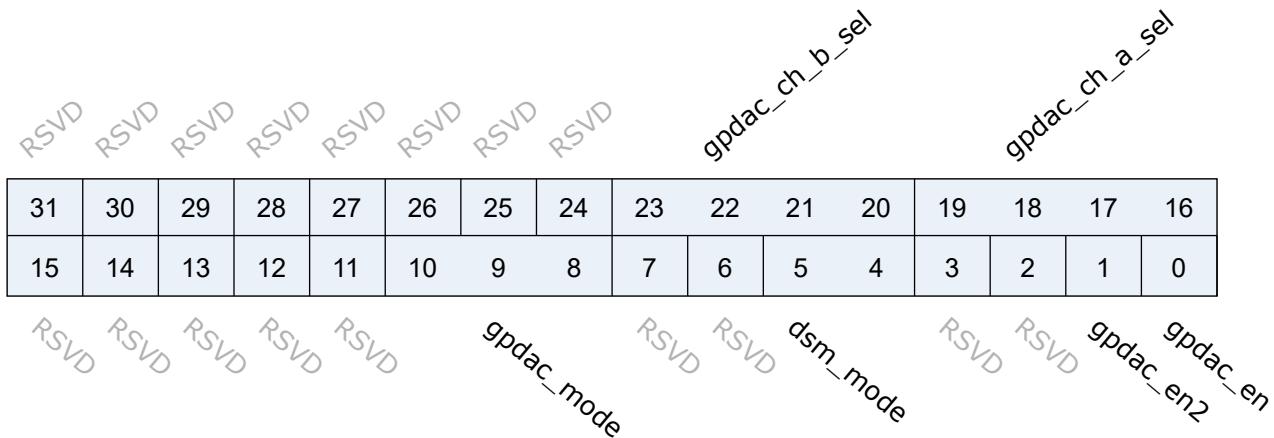
Address: 0x40002004



Bits	Name	Type	Reset	Description
31:26	RSVD			
25:0	gpadc_dma_rdata	r	26'd0	GPADC finial conversion result stored in the FIFO

4.4.3 gpdac_config

Address: 0x40002040

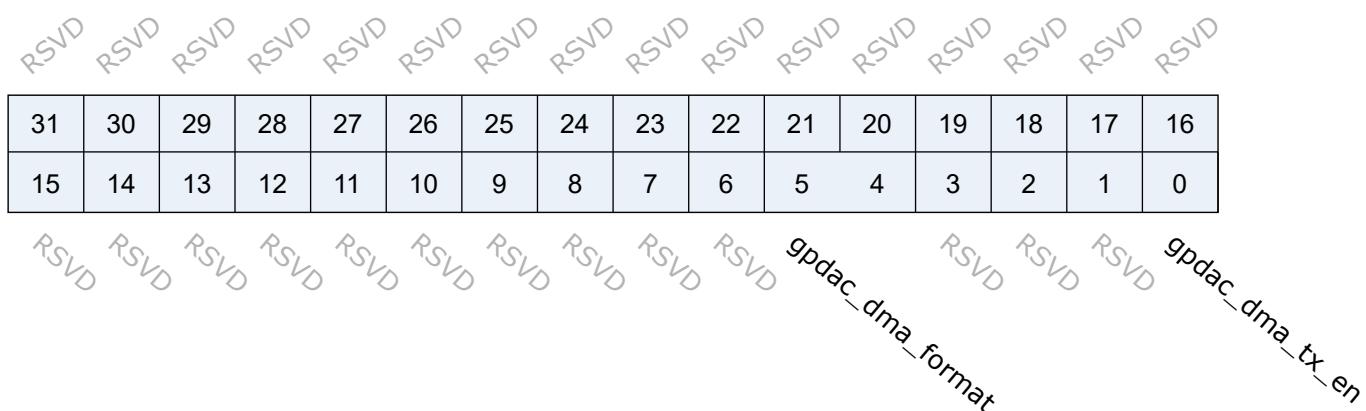


Bits	Name	Type	Reset	Description
31:24	RSVD			

Bits	Name	Type	Reset	Description
23:20	gpdac_ch_b_sel	r/w	0	Channel B Source Select 0: Reg 1: DMA 2: DMA + Filter 3: Sin Gen 4: A (The same as channel A) 5: A (Inverse of channel A)
19:16	gpdac_ch_a_sel	r/w	0	Channel A Source Select 0: Reg 1: DMA 2: DMA + Filter 3: Sin Gen
15:11	RSVD			
10:8	gpdac_mode	r/w	0	0:32k, 1:16k, 3:8k, 4:512k(for DMA only)
7:6	RSVD			
5:4	dsm_mode	r/w	0	0:bypass, 1:dsm order=1, 2: dsm order=2
3:2	RSVD			
1	gpdac_en2	r/w	0	GPDAC enable 2 (for B channel)
0	gpdac_en	r/w	0	GPDAC enable

4.4.4 gpdac_dma_config

Address: 0x40002044



Bits	Name	Type	Reset	Description
31:6	RSVD			

Bits	Name	Type	Reset	Description
5:4	gpdac_dma_format	r/w	0	DMA TX format (Data 12-bit) 0: A0, A1, A2... 1: B0,A0, B1,A1, B2,A2... 2: A1,A0, A3,A2, A5,A4... (Note: 20'h0,[11:0] or 4'h0,[27:16],4'h0,[11:0])
3:1	RSVD			
0	gpdac_dma_tx_en	r/w	0	GPDAC DMA TX enable

4.4.5 gpdac_dma_wdata

Address: 0x40002048

gpdac_dma_wdata

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

gpdac_dma_wdata

Bits	Name	Type	Reset	Description
31:0	gpdac_dma_wdata	w	x	GPDAC DMA TX data

4.4.6 gpadc_reg_cmd

Address: 0x4000f90c

RSVD	gpadc_sen_test_en	gpadc_sen_sel	RSVD	gpadc_chip_sen_pu	RSVD	RSVD	gpadc_micboost_32db_en	gpadc_mic_pga2_gain	gpadc_mic1_diff	gpadc_mic2_diff	gpadc_dwa_en	RSVD	gpadc_byp_micboost		
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
gpadc_micbias_en	gpadc_neg_gnd	gpadc_pos_sel	gpadc_neg_sel	gpadc_soft_rst	gpadc_global_en	gpadc_conv_start									
gpadc_micpga_en															

Bits	Name	Type	Reset	Description
31	RSVD			
30	gpadc_sen_test_en	r/w	1'b0	enable sensor dc test mux
29:28	gpadc_sen_sel	r/w	2'h0	selected output current channel and measurement channel 2'h0: 1st channel 2'h1: 2nd channel 2'h2: 3rd channel 2'h3: 4th channel
27	gpadc_chip_sen_pu	r/w	1'b0	enable chip sensor test 1'b0: disable 1'b1: enable
26:24	RSVD			
23	gpadc_micboost_32db_en	r/w	1'b0	micboost 32db enable 1'b0: 16dB 1'b1: 32dB
22:21	gpadc_mic_pga2_gain	r/w	2'h0	mic_pga2_gain 2'h0: 0dB 2'h1: 6dB 2'h2: -6dB 2'h3: 12dB
20	gpadc_mic1_diff	r/w	1'b0	mic1 diff enable 1'b0: single 1'b1: diff
19	gpadc_mic2_diff	r/w	1'b0	mic2 diff enable 1'b0: single 1'b1: diff
18	gpadc_dwa_en	r/w	1'b0	dwa enable 1'b0: dwa disable 1'b1: dwa enable
17	RSVD			
16	gpadc_byp_micboost	r/w	1'b0	micboost amp bypass 1'b0: not bypass 1'b1: bypass
15	gpadc_micpga_en	r/w	1'b0	micpga enable 1'b0: micpga disable 1'b1: miapga enable
14	gpadc_micbias_en	r/w	1'b0	enable micbias 1'b0: micbias power down 1'b1: miabias power on

Bits	Name	Type	Reset	Description
13	gpadc_neg_gnd	r/w	1'b0	set negative input of adc to ground 1'b0: disable 1'b1: enable
12:8	gpadc_pos_sel	r/w	5'hf	select adc positive input in none-scan mode 5 'h0 gpio0 5'h1 gpio1 5'h2 gpio2 5 'h3 gpio3 5'h4 gpio4 5'h5 gpio5 5 'h6 gpio6 5'h7 gpio7 5'h8 gpio8 5 'h9 gpio9 5'h10 gpio10 5'h11 gpio11 5 'h12 daca 5'h13 dacb 5'h14 temp_p 5 'h15 temp_n 5'h16 vref 5'h17 atest 5 'h18 vbat/2 5'h19 vp3_diode 5'h20 vp2_diode 5 'h21 vp1_diode 5'h22 vp0_diode 5'h23 31 avss

Bits	Name	Type	Reset	Description
7:3	gpadc_neg_sel	r/w	5'hf	select adc positive input in none-scan mode 5 'h0 gpio0 5'h1 gpio1 5'h2 gpio2 5 'h3 gpio3 5'h4 gpio4 5'h5 gpio5 5 'h6 gpio6 5'h7 gpio7 5'h8 gpio8 5 'h9 gpio9 5'h10 gpio10 5'h11 gpio11 5 'h12 daca 5'h13 dacb 5'h14 temp_p 5 'h15 temp_n 5'h16 vref 5'h17 atest 5 'h18 vbat/2 5'h19 vn3_diode 5'h20 vn2_diode 5 'h21 vn1_diode 5'h22 vn0_diode 5'h23 31 avss
2	gpadc_soft_RST	r/w	1'b0	user reset the whole block 1'h0: not reset 1'h1: reset
1	gpadc_conv_start	r/w	1'b0	1'h0: stop converation 1'h1: start converation
0	gpadc_global_en	r/w	1'b0	1'h0: disable ADC 1'h1: enable ADC

4.4.7 gpadc_reg_config1

Address: 0x4000f910

RSVD	RSVD	RSVD	RSVD	RSVD	gpadc_dither_en	gpadc_scan_en	gpadc_scan_length	gpadc_clk_div_ratio	gpadc_clk_ana_inv	RSVD
31	30	29	28	27	26	25	24	23	22	21
15	14	13	12	11	10	9	8	7	6	5
										4
										3
										2
										1
										0
RSVD	RSVD	RSVD	RSVD	RSVD	gpadc_lowv_det_en	gpadc_vcm_hyst_sel	RSVD	RSVD	gpadc_res_sel	gpadc_cal_os_en
RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	gpadc_cont_conv_en	RSVD

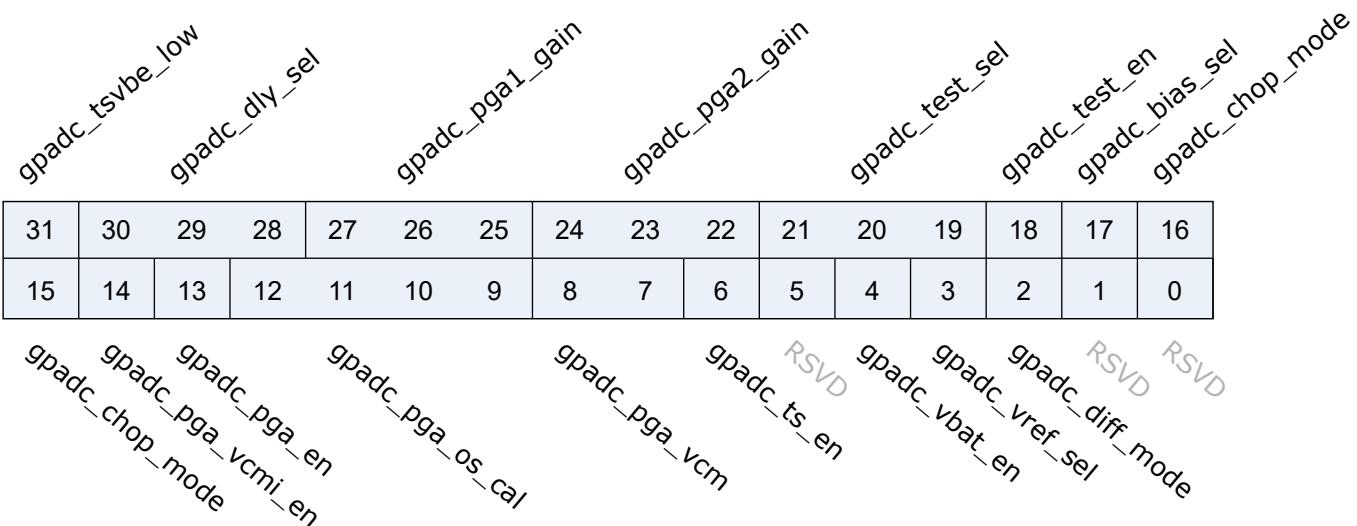
Bits	Name	Type	Reset	Description
31	RSVD			
30:29	gpadc_v18_sel	r/w	2'h0	internal vdd18 select
28:27	gpadc_v11_sel	r/w	2'h0	internal vdd11 select
26	gpadc_dither_en	r/w	1'h0	Dither compensation enable
25	gpadc_scan_en	r/w	1'h0	select scan mode enable: 0: select gpadc_pos/neg_sel;1: select : select gpadc_scan_pos_x and gpadc_scan_neg_x

Bits	Name	Type	Reset	Description
24:21	gpadc_scan_length	r/w	4'h0	select scan mode length 4'b0000 : select gpadc_scan_pos_0 and gpadc_scan_neg_0 4'b0001 : select gpadc_scan_pos_1 and gpadc_scan_neg_1 4'b0010 : select gpadc_scan_pos_2 and gpadc_scan_neg_2 4'b0011 : select gpadc_scan_pos_3 and gpadc_scan_neg_3 4'b0100 : select gpadc_scan_pos_4 and gpadc_scan_neg_4 4'b0101 : select gpadc_scan_pos_5 and gpadc_scan_neg_5 4'b0110 : select gpadc_scan_pos_6 and gpadc_scan_neg_6 4'b0111 : select gpadc_scan_pos_7 and gpadc_scan_neg_7 4'b1000 : select gpadc_scan_pos_8 and gpadc_scan_neg_8 4'b1001 : select gpadc_scan_pos_9 and gpadc_scan_neg_9 4'b1010 : select gpadc_scan_pos_10 and gpadc_scan_neg_10 4'b1011 : select gpadc_scan_pos_11 and gpadc_scan_neg_11
20:18	gpadc_clk_div_ratio	r/w	3'h3	analog 32M clock division ratio 3'b000: div=1 3'b001: div=4 3'b010: div=8 3'b011: div=12 3'b100: div=16 3'b101: div=20 3'b110: div=24 3'b111: div=32
17	gpadc_clk_ana_inv	r/w	1'b0	analog clock 2M inverted
16:11	RSVD			
10	gpadc_lowv_det_en	r/w	1'b0	Low power supply detected enable
9	gpadc_vcm_hyst_sel	r/w	1'b0	pga vcm hystersis select when vcm_sel_en is enabled
8	gpadc_vcm_sel_en	r/w	1'b0	pga vcm selected when lowv_det_en is enable
7:5	RSVD			

Bits	Name	Type	Reset	Description
4:2	gpadc_res_sel	r/w	3'h0	adc resolution/over-sample rate select 3'b000 12bit 2MS/s, OSR=1 3'b001 14bit 125kS/s, OSR=16 3'b010 14bit 31.25kS/s, OSR=64 3'b011 16bit 15.625KS/s, OSR=128 (voice mode 16KS/s) 3'b100 16bit 7.8125KS/s, OSR=256 (voice mode 8KS/s)
1	gpadc_cont_conv_en	r/w	1'b1	To enable continuous conversion 1'h0: one shot conversion 1'h1: continuous conversion
0	gpadc_cal_os_en	r/w	1'b0	offset calibration enable

4.4.8 gpadc_reg_config2

Address: 0x4000f914

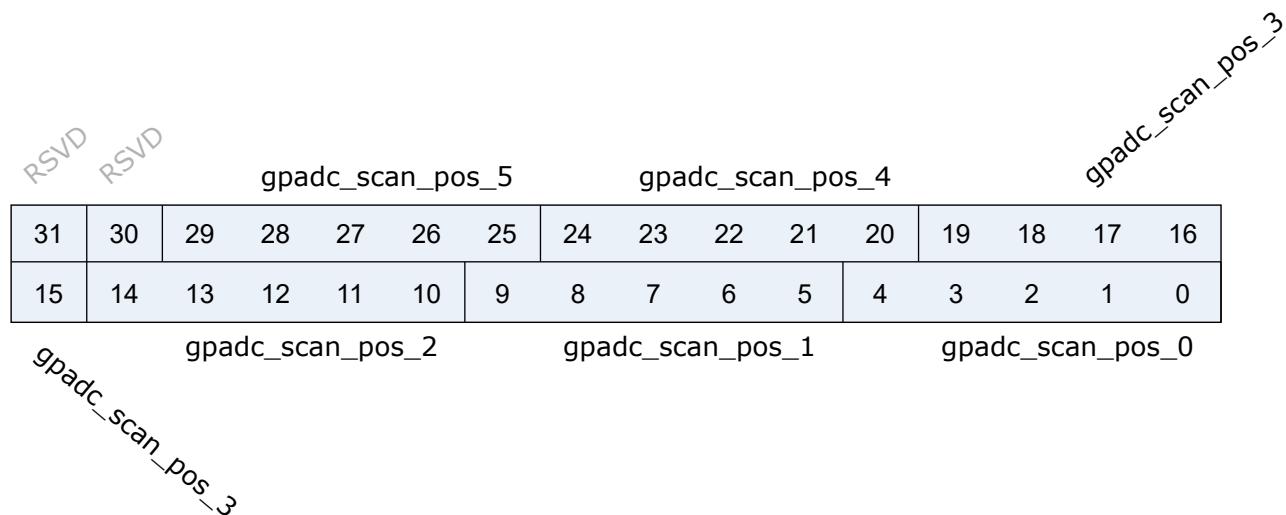


Bits	Name	Type	Reset	Description
31	gpadc_tsvbe_low	r/w	1'b0	tsen diode current
30:28	gpadc_dly_sel	r/w	3'h0	adc conversion speed
27:25	gpadc_pga1_gain	r/w	3'h0	3'h0: disable 3'h1: gain=1 3'h2: gain=2 3'h3: gain=4 3'h4: gain=8 3'h5: gain=16 3'h6: gain=32 3'h7: gain=32

Bits	Name	Type	Reset	Description
24:22	gpadc_pga2_gain	r/w	3'h0	3'h0: disable 3'h1: gain=1 3'h2: gain=2 3'h3: gain=4 3'h4: gain=8 3'h5: gain=16 3'h6: gain=32 3'h7: gain=32
21:19	gpadc_test_sel	r/w	3'h0	select test point 0 7
18	gpadc_test_en	r/w	1'b0	Analog test enable.
17	gpadc_bias_sel	r/w	1'b0	adc analog portion low power mode select 1'h0: bandgap system 1'h1:aon bandgap
16:15	gpadc_chop_mode	r/w	2'b3	2'b11 all off 2'b11 Vref AZ on 2'b11 Vref AZ and PGA chop on 2'b11 Vref AZ and PGA chop+RPC on
14	gpadc_pga_vcmi_en	r/w	1'b0	enable pga input vcm bias
13	gpadc_pga_en	r/w	1'b0	1'h0: disable PGA 1'h1 enable PGA
12:9	gpadc_pga_os_cal	r/w	4'h8	pga offset calibration
8:7	gpadc_pga_vcm	r/w	2'h2	Audio PGA output common mode control 2'b00: cm=1.3V 2'b11: cm=1.4V 2'b11: cm=1.5V 2'b11: cm=1.6V
6	gpadc_ts_en	r/w	1'b0	1'h0: disable temperature sensor 1'h1: enable temperature sensor
5	gpadc_tsext_sel	r/w	1'b0	1'h0: internal diode mode 1'h1: external diode mode
4	gpadc_vbat_en	r/w	1'b0	1'h0: disable VBAT sensor 1'h1 enable VBAT sensor
3	gpadc_vref_sel	r/w	1'b0	ADC reference select 1'h0 3.2V 1'h1 2.0V
2	gpadc_diff_mode	r/w	1'b0	1'h0 single-ended 1'h1 differential
1:0	RSVD			

4.4.9 gpadc_reg_scn_pos1

Address: 0x4000f918



Bits	Name	Type	Reset	Description
31:30	RSVD			
29:25	gpadc_scan_pos_5	r/w	5'hf	definition is the same as adc_reg_cmd.adc_pos_sel
24:20	gpadc_scan_pos_4	r/w	5'hf	definition is the same as adc_reg_cmd.adc_pos_sel
19:15	gpadc_scan_pos_3	r/w	5'hf	definition is the same as adc_reg_cmd.adc_pos_sel
14:10	gpadc_scan_pos_2	r/w	5'hf	definition is the same as adc_reg_cmd.adc_pos_sel
9:5	gpadc_scan_pos_1	r/w	5'hf	definition is the same as adc_reg_cmd.adc_pos_sel
4:0	gpadc_scan_pos_0	r/w	5'hf	definition is the same as adc_reg_cmd.adc_pos_sel

4.4.10 gpadc_reg_scn_pos2

Address: 0x4000f91c

RSVD		RSVD		gpadc_scan_pos_11					gpadc_scan_pos_10					gpadc_scan_pos_9			
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			
gpadc_scan_pos_8								gpadc_scan_pos_7					gpadc_scan_pos_6				

Bits	Name	Type	Reset	Description
31:30	RSVD			
29:25	gpadc_scan_pos_11	r/w	5'hf	definition is the same as adc_reg_cmd.adc_pos_sel
24:20	gpadc_scan_pos_10	r/w	5'hf	definition is the same as adc_reg_cmd.adc_pos_sel
19:15	gpadc_scan_pos_9	r/w	5'hf	definition is the same as adc_reg_cmd.adc_pos_sel
14:10	gpadc_scan_pos_8	r/w	5'hf	definition is the same as adc_reg_cmd.adc_pos_sel
9:5	gpadc_scan_pos_7	r/w	5'hf	definition is the same as adc_reg_cmd.adc_pos_sel
4:0	gpadc_scan_pos_6	r/w	5'hf	definition is the same as adc_reg_cmd.adc_pos_sel

4.4.11 gpadc_reg_scn_neg1

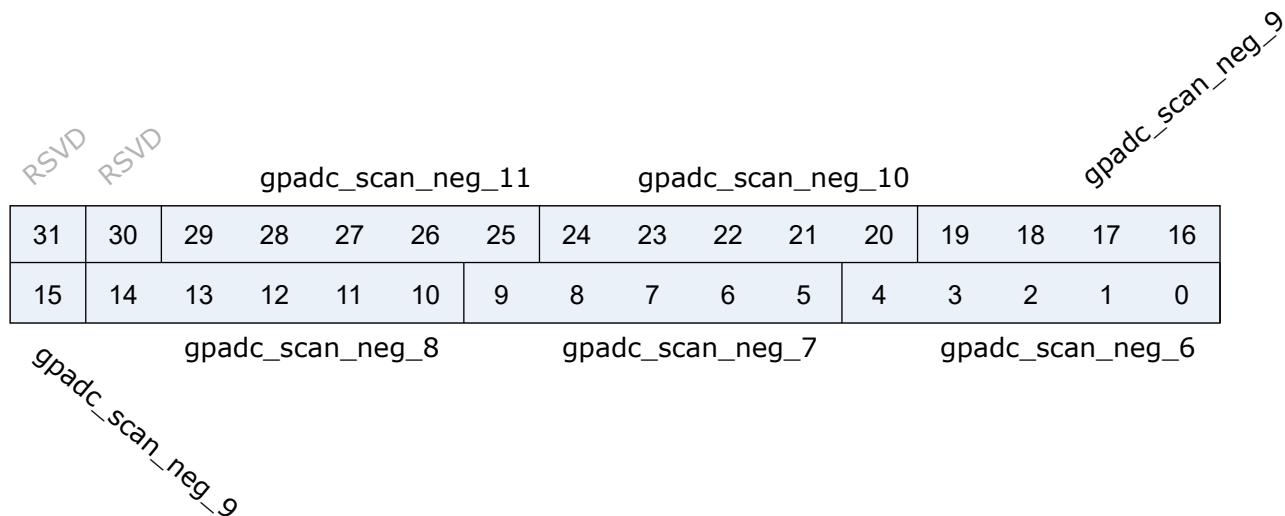
Address: 0x4000f920

RSVD		RSVD		gpadc_scan_neg_5					gpadc_scan_neg_4					gpadc_scan_neg_3			
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			
gpadc_scan_neg_2								gpadc_scan_neg_1					gpadc_scan_neg_0				

Bits	Name	Type	Reset	Description
31:30	RSVD			
29:25	gpadc_scan_neg_5	r/w	5'hf	definition is the same as adc_reg_cmd.adc_neg_sel
24:20	gpadc_scan_neg_4	r/w	5'hf	definition is the same as adc_reg_cmd.adc_neg_sel
19:15	gpadc_scan_neg_3	r/w	5'hf	definition is the same as adc_reg_cmd.adc_neg_sel
14:10	gpadc_scan_neg_2	r/w	5'hf	definition is the same as adc_reg_cmd.adc_neg_sel
9:5	gpadc_scan_neg_1	r/w	5'hf	definition is the same as adc_reg_cmd.adc_neg_sel
4:0	gpadc_scan_neg_0	r/w	5'hf	definition is the same as adc_reg_cmd.adc_neg_sel

4.4.12 gpadc_reg_scn_neg2

Address: 0x4000f924



Bits	Name	Type	Reset	Description
31:30	RSVD			
29:25	gpadc_scan_neg_11	r/w	5'hf	definition is the same as adc_reg_cmd.adc_neg_sel
24:20	gpadc_scan_neg_10	r/w	5'hf	definition is the same as adc_reg_cmd.adc_neg_sel
19:15	gpadc_scan_neg_9	r/w	5'hf	definition is the same as adc_reg_cmd.adc_neg_sel
14:10	gpadc_scan_neg_8	r/w	5'hf	definition is the same as adc_reg_cmd.adc_neg_sel
9:5	gpadc_scan_neg_7	r/w	5'hf	definition is the same as adc_reg_cmd.adc_neg_sel
4:0	gpadc_scan_neg_6	r/w	5'hf	definition is the same as adc_reg_cmd.adc_neg_sel



4.4.13 gpadc_reg_status

Address: 0x4000f928

Bits	Name	Type	Reset	Description
31:16	gpadc_reserved	r/w	16'h0	
15:1	RSVD			
0	gpadc_data_rdy	r	1'b0	ADC final conversion data ready

4.4.14 gpadc_reg_isr

Address: 0x4000f92c

Bits	Name	Type	Reset	Description
31:10	RSVD			
9	gpadc_pos_satur_mask	r/w	1'h0	write 1 mask
8	gpadc_neg_satur_mask	r/w	1'h0	write 1 mask
7:6	RSVD			
5	gpadc_pos_satur_clr	r/w	1'b0	Write 1 to clear flag

Bits	Name	Type	Reset	Description
4	gpadc_neg_satur_clr	r/w	1'b0	Write 1 to clear flag
3:2	RSVD			
1	gpadc_pos_satur	r	1'b0	ADC data positive side saturation interrupt flag
0	gpadc_neg_satur	r	1'b0	ADC data negative side saturation interrupt flag

4.4.15 gpadc_reg_raw_result

Address: 0x4000f934

RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD
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	
gpadc_raw_data																

Bits	Name	Type	Reset	Description
31:12	RSVD			
11:0	gpadc_raw_data	r	12'h0	ADC Raw data

4.4.16 gpadc_reg_define

Address: 0x4000f938

RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD
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	
gpadc_os_cal_data																

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	gpadc_os_cal_data	r/w	16'h0	User defined or self calculated offset data 16-bit signed

5.1 DAC introduction

The chip has a built-in 10bits digital-to-analog converter (DAC) with a FIFO depth of 1, and supports 2 DAC modulation outputs. Can be used for audio playback, conventional analog signal modulation.

5.2 DAC main feature

- DAC modulation accuracy is 10bits
- DAC input clock can be selected as 32M or Audio PLL
- Support DMA to transfer memory to DAC modulation register
- Support dual channel playback DMA transport mode
- The output pin of DAC is fixed to ChannelA as GPIO11, ChannelB as GPIO17
- DAC reference voltage can be selected internally

5.3 DAC function description

The basic block diagram of the DAC module is shown in the figure.

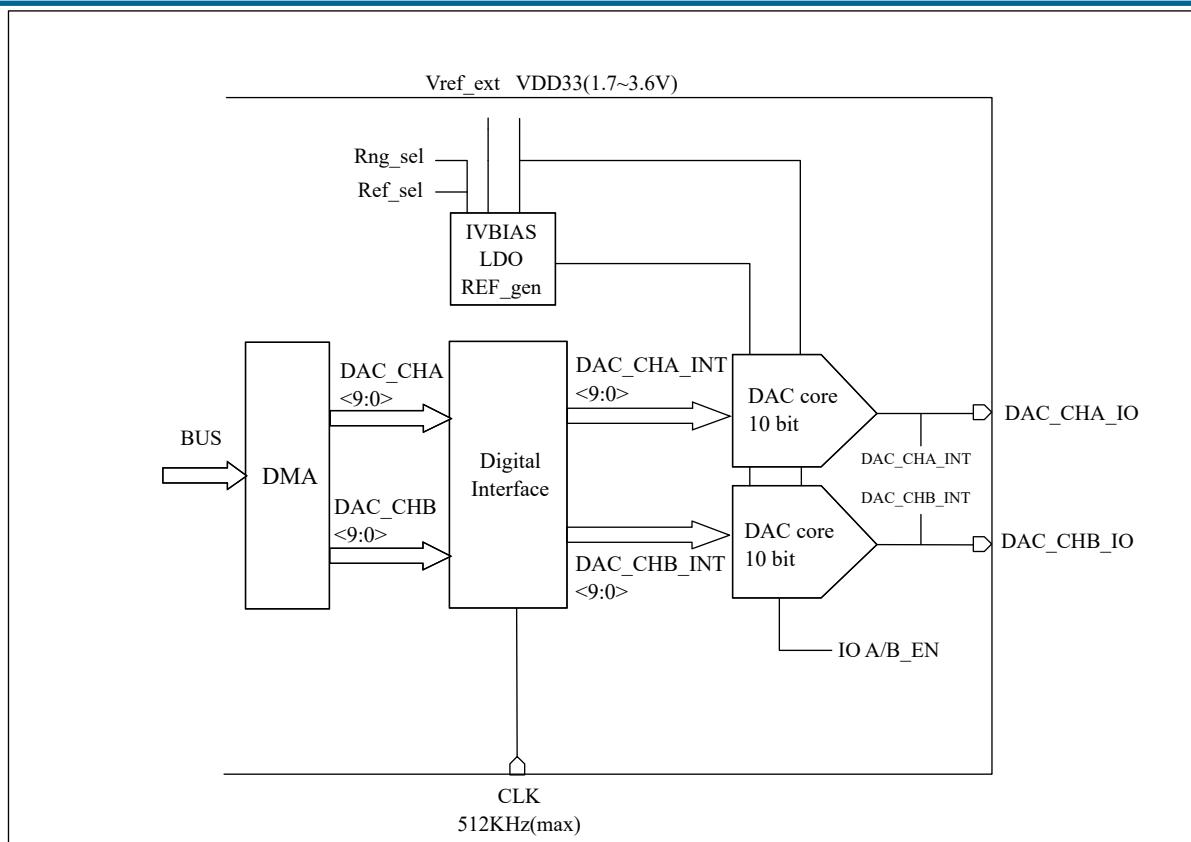


Fig. 5.1: DAC basic block diagram

The DAC module contains two DAC modulation circuits and a power supply circuit related to modulating analog signals. The user can use Ref_Sel to select whether the DAC reference voltage is external/internal, and Ref_Rng to select the internal reference voltage source.

The modulation data of the DAC can be directly written into the DAC modulation register (GLB_GPDAC_A_DATA, GLB_GPDAC_B_DATA in 0x40000314) by the CPU, or it can be transferred to the gpdac_dma_wdata (0x40002048) register by the DMA.

DAC data writing method

The CPU directly writes the GLB_GPDAC_A_DATA, GLB_GPDAC_B_DATA registers to complete the modulation, or uses DMA to transfer the data that needs to be modulated to gpdac_dma_wdata.

DMA handling mode

gpdac_dma_wdata (0x40002048) is a 32BITS register. The default meaning is that the 32BITS values are all modulated on the ChannelA pin in order. It can also be configured as the high 16 bits which correspond to the analog voltage output of Channel B by default, and the low 16 bits correspond to the analog voltage output of Channel A.

Note that whether it is 32/16-bit modulation, only the lower 10 bits are valid, because the maximum modulation accuracy of the DAC is 10BITS. The user can modify the meaning of the high and low bytes transported by the DMA by configuring the gpdac_dma_format register.

If gpdac_dma_format is 0, the data transferred by DMA into gpdac_dma_wdata are all modulated in Channel A in sequence, and the modulation order is {A0},{A1},{A2},...

If gpdac_dma_format is 1, the high 16 bits of the data transferred into gpdac_dma_wdata by DMA are modulated in Channel B, and the low 16 bits are modulated in Channel A. The modulation sequence is {B0,A0},{B1,A1},{B2,A2},... . This feature is very useful in stereo playback.

If gpdac_dma_format is 2, the data transferred by DMA into gpdac_dma_wdata is all modulated in Channel A, but the order of modulation is {A1,A0},{A3,A2},{A5,A4},...

DAC external reference voltage selection

The user can select external reference voltage or internal reference voltage by configuring gpdac_ref_sel (0x40000308[8]).

If the internal reference voltage is selected, the configuration is shown in the following table.

Table 5.1: Internal reference voltage

gpdac_a_rng	gpdac_ref_sel	Output range
00	0	0.2-1
01/10	0	0.225-1.425
11	0	0.2-1.8

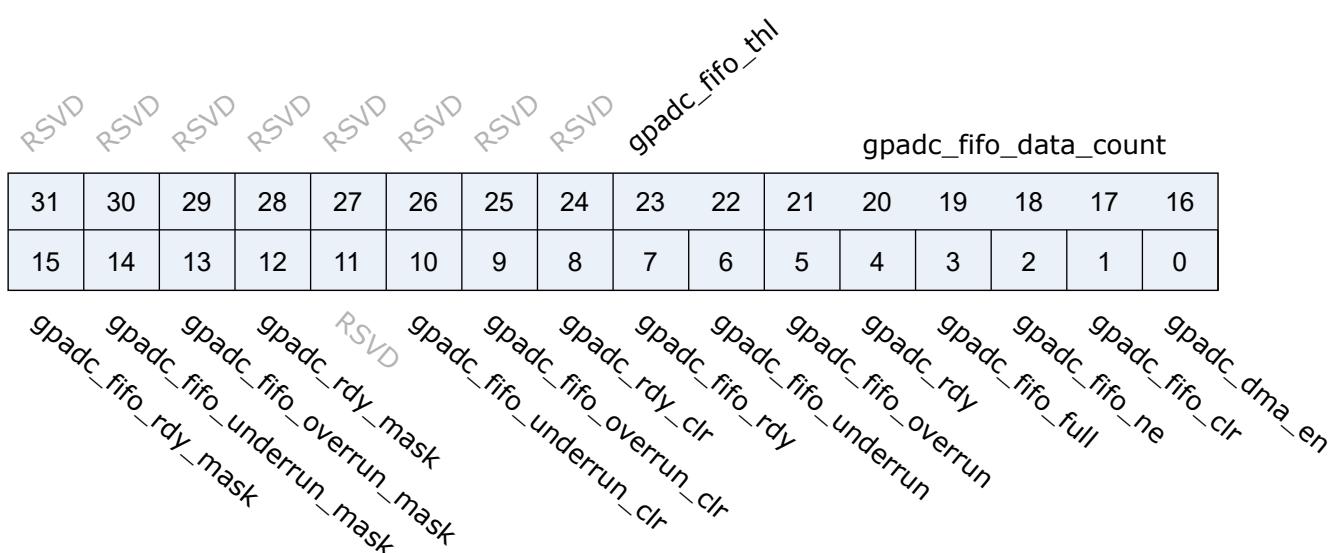
If you choose an external reference voltage, please connect the external voltage to the fixed GPIO7.

5.4 Register description

Name	Description
gpadc_config	GPADC configuration
gpadc_dma_rdata	GPADC DMA read data
gpdac_config	GPDAC configuration
gpdac_dma_config	GPDAC dma configuration
gpdac_dma_wdata	GPDAC dma write data

5.4.1 gpadc_config

Address: 0x40002000



Bits	Name	Type	Reset	Description
31:24	RSVD			
23:22	gpadc_fifo_thl	r/w	2'd0	fifo threshold 2'b00: 1 data 2'b01: 4 data 2'b10: 8 data 2'b11: 16 data
21:16	gpadc_fifo_data_count	r	6'd0	fifo data number
15	gpadc_fifo_rdy_mask	r/w	1'b1	write 1 mask
14	gpadc_fifo_underrun_mask	r/w	1'b0	write 1 mask
13	gpadc_fifo_overrun_mask	r/w	1'b0	write 1 mask
12	gpadc_rdy_mask	r/w	1'b0	write 1 mask
11	RSVD			
10	gpadc_fifo_underrun_clr	w1c	1'b0	Write 1 to clear flag
9	gpadc_fifo_overrun_clr	w1c	1'b0	Write 1 to clear flag
8	gpadc_rdy_clr	w1c	1'b0	Write 1 to clear flag
7	gpadc_fifo_rdy	r	1'b0	FIFO ready interrupt flag
6	gpadc_fifo_underrun	r	1'b0	FIFO underrun interrupt flag
5	gpadc_fifo_overrun	r	1'b0	FIFO overrun interrupt flag
4	gpadc_rdy	r	1'b0	Conversion data ready interrupt flag

Bits	Name	Type	Reset	Description
3	gpadc_fifo_full	r	1'b0	FIFO full flag
2	gpadc_fifo_ne	r	1'b0	FIFO not empty flag
1	gpadc_fifo_clr	w1c	1'b0	FIFO clear signal
0	gpadc_dma_en	r/w	1'b0	GPADC DMA enable

5.4.2 gpadc_dma_rdata

Address: 0x40002004

gpadc_dma_rdata															
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
gpadc_dma_rdata															

Bits	Name	Type	Reset	Description
31:26	RSVD			
25:0	gpadc_dma_rdata	r	26'd0	GPADC final conversion result stored in the FIFO

5.4.3 gpdac_config

Address: 0x40002040

gpdac_config															
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
gpdac_config															
RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD
gpdac_mode				RSVD				dsm_mode				RSVD			
RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD
gpdac_en2				gpdac_en				gpdac_ch_a_sel				gpdac_ch_b_sel			

Bits	Name	Type	Reset	Description
31:24	RSVD			
23:20	gpdac_ch_b_sel	r/w	0	Channel B Source Select 0: Reg 1: DMA 2: DMA + Filter 3: Sin Gen 4: A (The same as channel A) 5: A (Inverse of channel A)
19:16	gpdac_ch_a_sel	r/w	0	Channel A Source Select 0: Reg 1: DMA 2: DMA + Filter 3: Sin Gen
15:11	RSVD			
10:8	gpdac_mode	r/w	0	0:32k, 1:16k, 3:8k, 4:512k(for DMA only)
7:6	RSVD			
5:4	dsm_mode	r/w	0	0:bypass, 1:dsm order=1, 2: dsm order=2
3:2	RSVD			
1	gpdac_en2	r/w	0	GPDAC enable 2 (for B channel)
0	gpdac_en	r/w	0	GPDAC enable

5.4.4 gpdac_dma_config

Address: 0x40002044

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | | | |

RSVD RSVD RSVD

gpdac_dma_format gpdac_dma_tx_en

Bits	Name	Type	Reset	Description
31:6	RSVD			
5:4	gpdac_dma_format	r/w	0	DMA TX format (Data 12-bit) 0: A0, A1, A2... 1: B0,A0, B1,A1, B2,A2... 2: A1,A0, A3,A2, A5,A4... (Note: 20'h0,[11:0] or 4'h0,[27:16],4'h0,[11:0])
3:1	RSVD			
0	gpdac_dma_tx_en	r/w	0	GPDAC DMA TX enable

5.4.5 gpdac_dma_wdata

Address: 0x40002048

gpdac_dma_wdata

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

gpdac_dma_wdata

Bits	Name	Type	Reset	Description
31:0	gpdac_dma_wdata	w	x	GPDAC DMA TX data

6.1 DMA Introduction

DMA (Direct Memory Access) is a memory access technology that can independently read and write system memory directly without processor intervention. Under the same degree of processor load, DMA is a fast data transfer method. The DMA controller has 8 channels, which manage the data transfer between peripheral devices and memory to improve bus efficiency.

There are four main types of transfers: memory to memory, memory to peripheral, peripheral to peripheral and peripheral to memory. And support LLI link list function. Use the software to configure the transmission data size, data source address, and destination address.

6.2 DMA main features

- 8 independently configurable channels (requests) on DMA
- Independent control of source and destination access width (single-byte, double-byte, four-byte)
- Each channel acts as a read-write cache independently
- Each channel can be triggered by independent peripheral hardware or software
- Support peripherals including UART、I2C、SPI、ADC、I2S、DAC。
- 4 kinds of process control
 - DMA flow control, source memory, target memory
 - DMA flow control, source memory, target peripheral
 - DMA flow control, source peripheral, target memory
 - DMA flow control, source peripheral, target peripheral
- Support LLI linked list function to improve DMA efficiency

6.3 DMA functional description

6.3.1 Working principle

When a device attempts to transfer data (usually a large amount of data) directly to another device via the bus, it will first send a DMA request signal to the CPU. The peripheral device makes a bus request to the CPU to take over the bus control right through the DMA. After the CPU receives the signal, after the current bus cycle ends, it will respond to the DMA signal according to the priority of the DMA signal and the order of the DMA request.

When the CPU responds to a DMA request to a device interface, it will give up bus control.

Therefore, under the management of the DMA controller, the peripherals and the memory directly exchange data without CPU intervention. After the data transfer is complete, the device sends a DMA end signal to the CPU, returning the bus control.

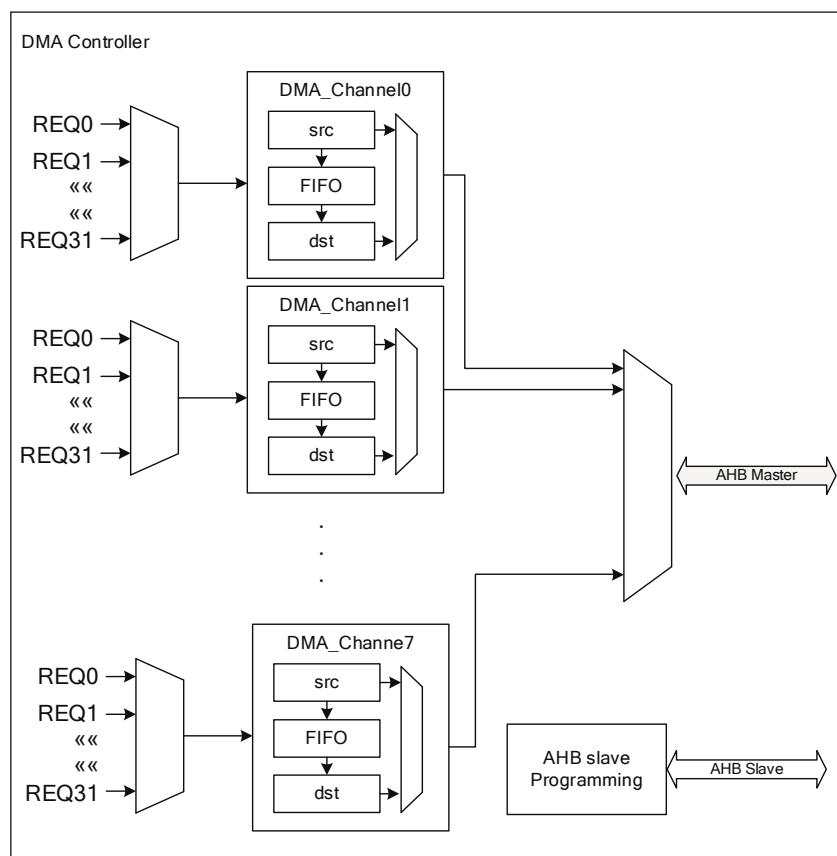


Fig. 6.1: DMA architecture

The DMA includes a set of AHB Master interfaces and a set of AHB Slave interfaces. The AHB Master interface actively accesses memory or peripherals through the system bus according to the current configuration requirements, as a port for data movement. The AHB Slave interface is used to configure the DMA interface and only supports 32-bit access.

6.3.2 DMA channel configuration

DMA supports 8 channels in total, each channel does not interfere with each other and can run at the same time. The following is the configuration process of DMA channel x:

1. Set 32-bit source address in DMA_C0SrcAddr register
2. Set the 32-bit target address in the DMA_C0DstAddr register
3. Configure SI (source) and DI (destination) in the DMA_C0Control register to set whether to enable the automatic address accumulation mode. When set to 1, enable the automatic address accumulation mode
4. Set the transmission data width by configuring the STW (source) and DTW (destination) bits in the DMA_C0Control register. The width options are single-byte, double-self-knot, and four-byte
5. Burst type, which can be set by configuring the SBS (source) and DBS (destination) bits in the DMA_C0Control register. The configuration options are Single, INCR4, INCR8, INCR16
6. Special attention should be paid to the configured combination. A single burst cannot exceed 16 bytes
7. Set the data transmission length range: 0-4095

6.3.3 Peripheral support

The SrcPeripheral (source) and DstPeripheral (destination) are configured to determine the peripherals that the current DMA cooperates with. The relationship is 0-5 : UART / 6-9 : I2C / 10-13 : SPI / 18-21 : I2S / 22 : ADC / 23 : DAC

UART uses DMA to transfer data

UART sends data packets, using DMA method can greatly reduce CPU processing time, so that its CPU resources are not wasted a lot. Especially when the UART sends and receives a large number of data packets (such as high-frequency sending and receiving instructions) has obvious advantages.

Taking UART0 transmission as an example, the configuration process is as follows:

1. Set the value of the register DMA_C0Config [SRCPH] bit to 1, that is, set the Source peripheral to UART_TX
2. Set the value of the DMA_C0Config [DSTPH] bit to 0, that is, set the Destination peripheral to UART_RX

I2C uses DMA to transfer data

The configuration is as follows:

1. Set the value of the register DMA_C0Config [SRCPH] bit to 7, that is, set the Source peripheral to I2C_TX
2. Set the value of the DMA_C0Config [DSTPH] bit to 6, that is, set the Destination peripheral to I2C_RX

SPI uses DMA to transfer data

The configuration is as follows:

1. Set the value of the DMA_C0Config [SRCPH] bit to 11, that is, set the Source peripheral to SPI_TX

2. Set the value of the DMA_C0Config [DSTPH] bit to 10, that is, set the Destination peripheral to SPI_RX

ADC uses DMA to transfer data

The configuration is as follows:

1. Set the value of the DMA_C0Config [SRCPH] bit to 22, that is, set the Source peripheral to GPADC

DAC uses DMA to transfer data

The configuration is as follows:

1. Set the value of the DMA_C0Config [SRCPH] bit to 23, that is, set the Source peripheral to GPDAC

6.3.4 Linked List Mode

DMA supports linked list operation mode. When performing a DMA read or write operation, you can fill the data in the next linked list. After completing the data transfer of the current linked list, read the DMA_C0LLI register to obtain the start address of the next linked list, and directly transfer the data in the next linked list.

Ensure continuous and uninterrupted work during DMA transfer, and improve the efficiency of CPU and DMA.

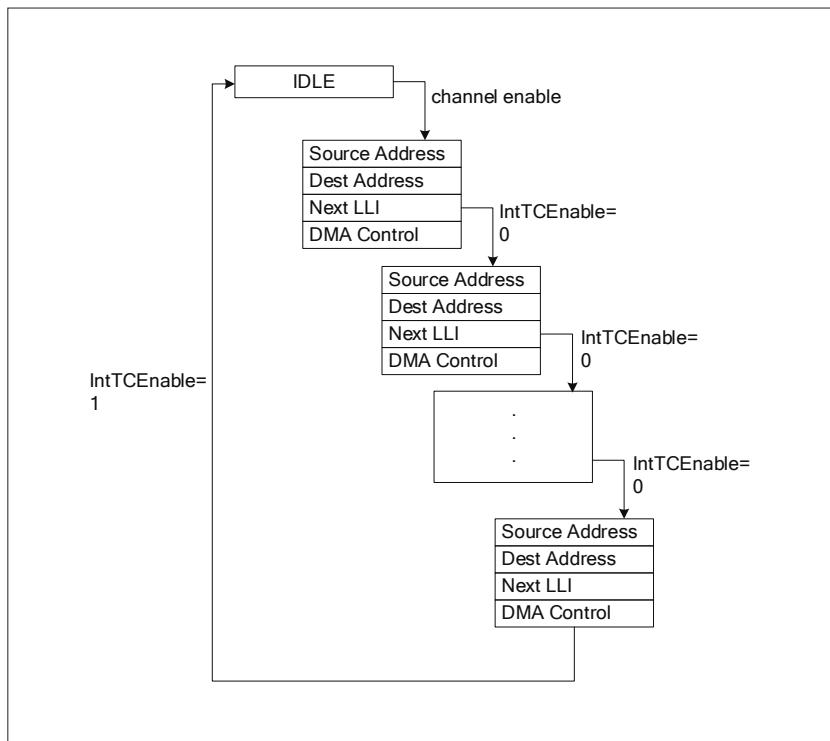


Fig. 6.2: LLI architecture

6.3.5 DMA interrupt

- DMA_INT_TCOMPLETED
 - Data transmission completed interrupt. When a data transmission is completed, this interrupt will be entered.
- DMA_INT_ERR
 - Data transmission error interrupt, when an error occurs during data transmission, this interrupt will be entered

6.4 Transmission mode

6.4.1 Memory to memory

After this mode is started, the DMA will move the data from the source address to the destination address according to the set transfer size. After the transfer, the DMA controller will automatically return to the idle state and wait for the next transfer.

The specific configuration process is as follows:

1. Set the value of the register DMA_C0SrcAddr to the memory address of the source
2. Set the value of the register DMA_C0DstAddr to the target memory address
3. Select the transmission mode and set the value of the DMA_C0Config [FLOWCTRL] bit to 0, that is, select the memory-to-memory mode
4. Set the value of the corresponding bit in the DMA_C0Control register: set the DI and SI bits to 1 to enable the automatic address accumulation mode, the DTW and STW bits set the transmission width of the source and destination, and the DBS and SBS bits set the burst type of the source and destination
5. Select the appropriate channel, enable DMA, and complete the data transfer

6.4.2 Memory to peripheral

In this working mode, the DMA will move data from the source to the internal cache according to the set transfer size (TransferSize). When the cache space is insufficient, the DMA will automatically suspend it. When there is sufficient cache space, continue to transfer until it reaches Set the moving quantity.

On the other hand, when the target peripheral request triggers, it will burst the target configuration to the target address until it reaches the set number of moves and automatically returns to the idle state, waiting for the next startup.

The specific configuration process is as follows:

1. Set the value of the register DMA_C0SrcAddr to the memory address of the source
2. Set the value of the register DMA_C0DstAddr to the target peripheral address
3. Select the transfer mode and set the value of the DMA_C0Config [FLOWCTRL] bit to 1 to select the memory-to-

peripheral mode

4. Set the value of the corresponding bit in the DMA_C0Control register: the SI bit is set to 1 to enable the address auto-accumulation mode, and the DI bit is set to 0 to disable the address auto-accumulation mode, the DTW and STW bits set the transmission width of the source and destination, and the DBS and SBS bits set the burst type of the source and destination
5. Select the appropriate channel, enable DMA, and complete the data transfer

6.4.3 Peripheral to memory

In this working mode, when the source peripheral request is triggered, the source configuration is burst to the buffer until the set number of moves reaches the stop. On the other hand, when the internal cache is enough for the target burst number once, the DMA will automatically move the cached content to the target address until it reaches the set number of moves and automatically returns to the idle state, waiting for the next startup

The specific configuration process is as follows:

1. Set the value of the register DMA_C0SrcAddr to the source peripheral address
2. Set the value of the register DMA_C0DstAddr to the target memory address
3. Select the transfer mode and set the value of the DMA_C0Config [FLOWCTRL] bit to 2 to select the Peripheral-to-memory mode
4. Set the value of the corresponding bit in the DMA_C0Control register: the DI bit is set to 1 to enable the address auto-accumulation mode, and the SI bit is set to 0 to disable the address auto-accumulation mode , the DTW and STW bits set the transmission width of the source and destination respectively, and the DBS and SBS bits set the burst type of the source and destination respectively
5. Select the appropriate channel, enable DMA, and complete the data transfer

6.4.4 Peripheral to peripheral

In this working mode, when the source peripheral requests a trigger, the source configuration burst will be stored in the buffer, and it will stop until the set number of moves is reached. On the other hand, when the internal cache is enough for the target burst number, DMA will automatically move the cached content to the target address until the set number of transfers is reached and automatically return to the idle state, waiting for the next start

The specific configuration process is as follows:

1. Set the value of the register DMA_C0SrcAddr to the peripheral address of the source
2. Set the value of the register DMA_C0DstAddr to the target peripheral address
3. Select the transfer mode, set the value of the register DMA_C0Config[FLOWCTRL] bit to 3, that is, select the Peripheral-to-Peripheral mode
4. Set the value of the corresponding bit in the DMA_C0Control register: DI and SI bits are set to 0, the address

automatic accumulation mode is disabled, the STW and DTW bits respectively set the source and target transfer widths, and the SBS and DBS bits respectively set the source and target bursts type.

5. Select the appropriate channel, enable DMA, and complete the data transfer

6.5 Register description

Name	Description
DMA_IntStatus	Interrupt status
DMA_IntTCStatus	Interrupt terminal count request status
DMA_IntTCClear	Terminal count request clear
DMA_IntErrorStatus	Interrupt error status
DMA_IntErrClr	Interrupt error clear
DMA_RawIntTCStatus	Status of the terminal count interrupt prior to masking
DMA_RawIntErrorStatus	Status of the error interrupt prior to masking
DMA_EnbldChns	Channel enable status
DMA_SoftBReq	Software burst request
DMA_SoftSReq	Software single request
DMA_SoftLBReq	Software last burst request
DMA_SoftLSReq	Software last single request
DMA_Config	DMA general configuration
DMA_Sync	DMA request asynchronous setting
DMA_C0SrcAddr	Channel DMA source address
DMA_C0DstAddr	Channel DMA Destination address
DMA_COLLI	Channel DMA link list
DMA_C0Control	Channel DMA bus control
DMA_C0Config	Channel DMA configuration
DMA_C1SrcAddr	Channel DMA source address
DMA_C1DstAddr	Channel DMA Destination address
DMA_C1LLI	Channel DMA link list
DMA_C1Control	Channel DMA bus control
DMA_C1Config	Channel DMA configuration

Name	Description
DMA_C2SrcAddr	Channel DMA source address
DMA_C2DstAddr	Channel DMA Destination address
DMA_C2LLI	Channel DMA link list
DMA_C2Control	Channel DMA bus contro
DMA_C2Config	Channel DMA configuration
DMA_C3SrcAddr	Channel DMA source address
DMA_C3DstAddr	Channel DMA Destination address
DMA_C3LLI	Channel DMA link list
DMA_C3Control	Channel DMA bus control
DMA_C3Config	Channel DMA configuration
DMA_C4SrcAddr	Channel DMA source address
DMA_C4DstAddr	Channel DMA Destination address
DMA_C4LLI	Channel DMA link list
DMA_C4Control	Channel DMA bus control
DMA_C4Config	Channel DMA configuration
DMA_C5SrcAddr	Channel DMA source address
DMA_C5DstAddr	Channel DMA Destination address
DMA_C5LLI	Channel DMA link list
DMA_C5Control	Channel DMA bus control
DMA_C5Config	Channel DMA configuration
DMA_C6SrcAddr	Channel DMA source address
DMA_C6DstAddr	Channel DMA Destination address
DMA_C6LLI	Channel DMA link list
DMA_C6Control	Channel DMA bus control
DMA_C6Config	Channel DMA configuration
DMA_C7SrcAddr	Channel DMA source address
DMA_C7DstAddr	Channel DMA Destination address
DMA_C7LLI	Channel DMA link list
DMA_C7Control	Channel DMA bus control

Name	Description
DMA_C7Config	Channel DMA configuration

6.5.1 DMA_IntStatus

Address: 0x4000c000

RSVD																	
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		

IntStatus

Bits	Name	Type	Reset	Description
31:8	RSVD			
7:0	IntStatus	r	0	Status of the DMA interrupts after masking

6.5.2 DMA_IntTCStatus

Address: 0x4000c004

RSVD																	
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		

IntTCStatus

Bits	Name	Type	Reset	Description
31:8	RSVD			
7:0	IntTCStatus	r	0	Interrupt terminal count request status

6.5.3 DMA_IntTCClear

Address: 0x4000c008

RSVD															
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

IntTCClear

Bits	Name	Type	Reset	Description
31:8	RSVD			
7:0	IntTCClear	w	0	Terminal count request clear

6.5.4 DMA_IntErrorStatus

Address: 0x4000c00c

RSVD															
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

IntErrorStatus

Bits	Name	Type	Reset	Description
31:8	RSVD			
7:0	IntErrorStatus	r	0	Interrupt error status

6.5.5 DMA_IntErrClr

Address: 0x4000c010

RSVD															
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

RSVD RSVD RSVD RSVD RSVD RSVD RSVD RSVD IntErrClr

Bits	Name	Type	Reset	Description
31:8	RSVD			
7:0	IntErrClr	w	0	Interrupt error clear

6.5.6 DMA_RawIntTCStatus

Address: 0x4000c014

RSVD															
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

RSVD RSVD RSVD RSVD RSVD RSVD RSVD RawIntTCStatus

Bits	Name	Type	Reset	Description
31:8	RSVD			
7:0	RawIntTCStatus	r	0	Status of the terminal count interrupt prior to masking

6.5.7 DMA_RawIntErrorStatus

Address: 0x4000c018

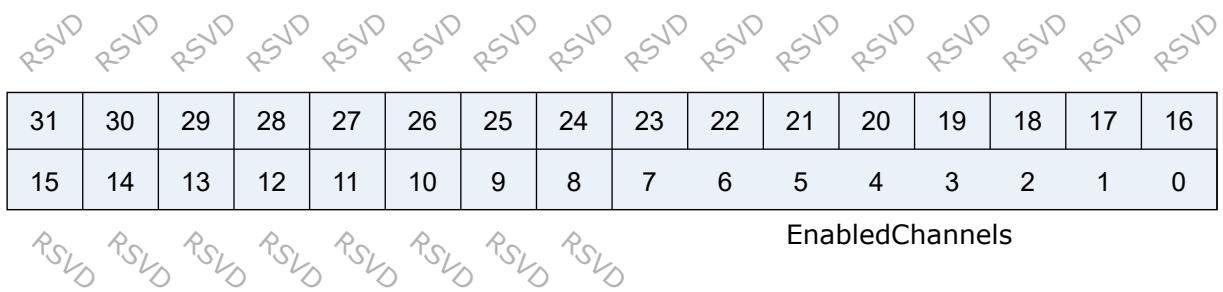
RSVD															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

RSVD RSVD RSVD RSVD RSVD RSVD RSVD RawIntErrorStatus

Bits	Name	Type	Reset	Description
31:8	RSVD			
7:0	RawIntErrorStatus	r	0	Status of the error interrupt prior to masking

6.5.8 DMA_EnbldChns

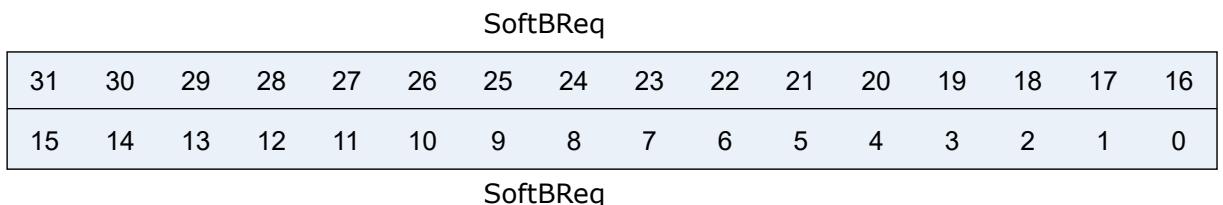
Address: 0x4000c01c



Bits	Name	Type	Reset	Description
31:8	RSVD			
7:0	EnabledChannels	r	0	Channel enable status

6.5.9 DMA_SoftBReq

Address: 0x4000c020



Bits	Name	Type	Reset	Description
31:0	SoftBReq	r/w	0	Software burst request

6.5.10 DMA_SoftSReq

Address: 0x4000c024

SoftSReq

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

SoftSReq

Bits	Name	Type	Reset	Description
31:0	SoftSReq	r/w	0	Software single request

6.5.11 DMA_SoftLBReq

Address: 0x4000c028

SoftLBReq

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

SoftLBReq

Bits	Name	Type	Reset	Description
31:0	SoftLBReq	r/w	0	Software last burst request

6.5.12 DMA_SoftLSReq

Address: 0x4000c02c

SoftLSReq

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

SoftLSReq

Bits	Name	Type	Reset	Description
31:0	SoftLSReq	r/w	0	Software last single request

6.5.13 DMA_Config

Address: 0x4000c030

RSVD	M	E															
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		

Bits	Name	Type	Reset	Description
31:2	RSVD			
1	M	r/w	0	AHB Master endianness configuration: 0 = little-endian, 1 = big-endian
0	E	r/w	0	SMDMA Enable.

6.5.14 DMA_Sync

Address: 0x4000c034

DMA_Sync																	
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		

DMA_Sync

Bits	Name	Type	Reset	Description
31:0	DMA_Sync	r/w	0	DMA synchronization logic for DMA request signals: 0 = enable, 1 = disable

6.5.15 DMA_C0SrcAddr

Address: 0x4000c100

SrcAddr																	
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		

SrcAddr

Bits	Name	Type	Reset	Description
31:0	SrcAddr	r/w	0	DMA source address

6.5.16 DMA_C0DstAddr

Address: 0x4000c104

DstAddr

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

DstAddr

Bits	Name	Type	Reset	Description
31:0	DstAddr	r/w	0	DMA Destination address

6.5.17 DMA_C0LLI

Address: 0x4000c108

LLI

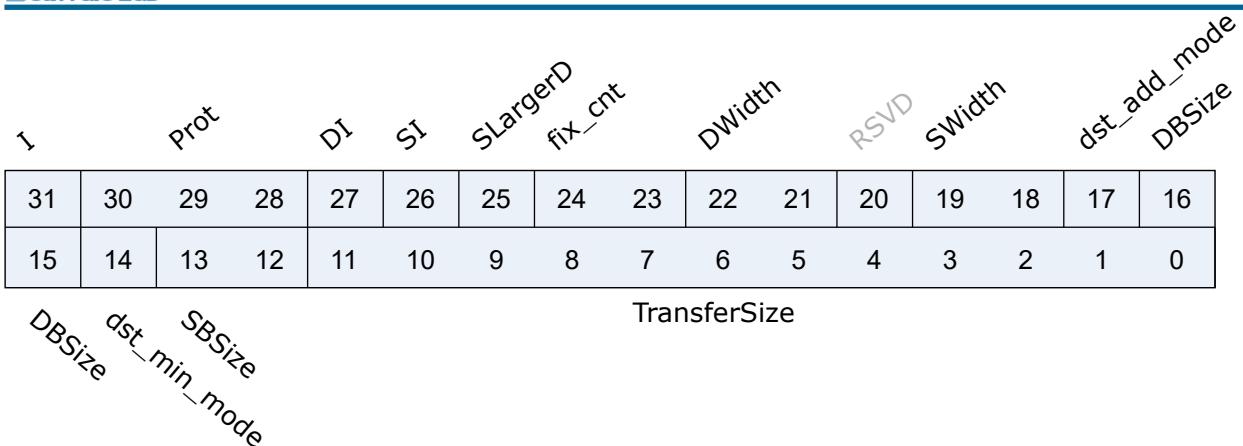
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

LLI

Bits	Name	Type	Reset	Description
31:0	LLI	r/w	0	First linked list item. Bits [1:0] must be 0.

6.5.18 DMA_C0Control

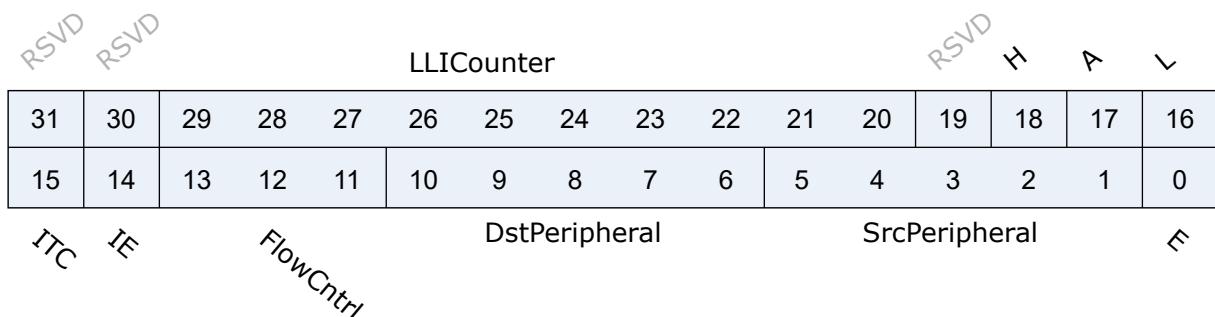
Address: 0x4000c10c



Bits	Name	Type	Reset	Description
31	I	r/w	0	Terminal count interrupt enable bit. It controls whether the current LLI is expected to trigger the terminal count interrupt.
30:28	Prot	r/w	0	Protection.
27	DI	r/w	1	Destination increment. When set, the Destination address is incremented after each transfer.
26	SI	r/w	1	Source increment. When set, the source address is incremented after each transfer.
25	SLargerD	r/w	0	In Memory-to-memory mode, Set this bit high when Src data size is larger than Dst.
24:23	fix_cnt	r/w	2'd0	Only effect when dst_min_mode = 1 Destination transfer cnt = (total src byte cnt - (fix_cnt«DWidth))«DWidth
22:21	DWidth	r/w	2'b10	Destination transfer width: 8/16/32
20	RSVD			
19:18	SWidth	r/w	2'b10	Source transfer width: 8/16/32
17	dst_add_mode	r/w	1'b0	Add mode : issue remain destination traffic
16:15	DBSize	r/w	2'b01	Destination burst size: 1/4/8/16
14	dst_min_mode	r/w	1'b0	Minus mode : Not issue all destination traffic
13:12	SBSIZE	r/w	2'b01	Source burst size: 1/4/8/16. Note CH FIFO Size is 16Bytes and SBSIZE*Swidth should <= 16B
11:0	TransferSize	r/w	0	Transfer size: 0 4095. Number of data transfers left to complete when the SMDMA is the flow controller.

6.5.19 DMA_C0Config

Address: 0x4000c110



Bits	Name	Type	Reset	Description
31:30	RSVD			
29:20	LLICounter	r	0	LLI counter. Increased 1 each LLI run. Cleared 0 when config Control.
19	RSVD			
18	H	r/w	0	Halt: 0 = enable DMA requests, 1 = ignore subsequent source DMA requests.
17	A	r	0	Active: 0 = no data in FIFO of the channel, 1 = FIFO of the channel has data.
16	L	r/w	0	Lock.
15	ITC	r/w	0	Terminal count interrupt mask.
14	IE	r/w	0	Interrupt error mask.
13:11	FlowCntrl	r/w	0	000: Memory-to-memory (DMA) 001: Memory-to-peripheral (DMA) 010: Peripheral-to-memory (DMA) 011: Source peripheral-to-Destination peripheral (DMA) 100: Source peripheral-to-Destination peripheral (Destination peripheral) 101: Memory-to-peripheral (peripheral) 110: Peripheral-to-memory (peripheral) 111: Source peripheral-to-Destination peripheral (Source peripheral)

Bits	Name	Type	Reset	Description
10:6	DstPeripheral	r/w	0	Destination peripheral. [23:22] GPADC [21:18] I2S [17:14] PDM [13:10] SSP [9: 6] I2C [5: 0] UART
5:1	SrcPeripheral	r/w	0	Source peripheral.
0	E	r/w	0	Channel enable.

6.5.20 DMA_C1SrcAddr

Address: 0x4000c200

SrcAddr

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

SrcAddr

Bits	Name	Type	Reset	Description
31:0	SrcAddr	r/w	0	

6.5.21 DMA_C1DstAddr

Address: 0x4000c204

DstAddr

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

DstAddr

Bits	Name	Type	Reset	Description
31:0	DstAddr	r/w	0	

6.5.22 DMA_C1LLI

Address: 0x4000c208

LLI																
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	
LLI																
RSVD RSVD																

Bits	Name				Type	Reset	Description									
31:2	LLI				r/w	0										
1:0	RSVD															

6.5.23 DMA_C1Control

Address: 0x4000c20c

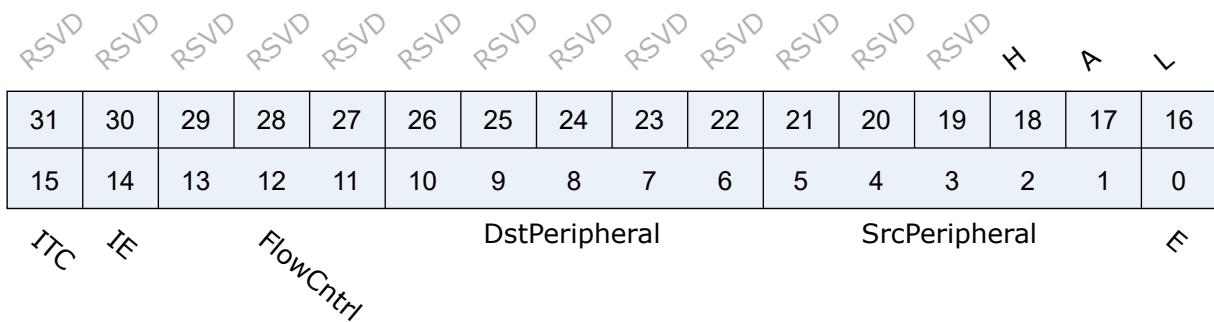
1	Prot	DI	SI	RSVD	fix_cnt	DWidth	SWidth	dst_add_mode	DBSize							
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
TransferSize																
DBSize dst_min_mode SBSIZE																

Bits	Name				Type	Reset	Description									
31	I				r/w	0										
30:28	Prot				r/w	0										
27	DI				r/w	1										
26	SI				r/w	1										
25	RSVD															
24:23	fix_cnt				r/w	2'd0										
22:24	RSVD															
23:21	DWidth				r/w	3'b010										

Bits	Name	Type	Reset	Description
20:18	SWidth	r/w	3'b010	
17	dst_add_mode	r/w	1'b0	
16:15	DBSize	r/w	3'b001	
14	dst_min_mode	r/w	1'b0	
13:12	SBSIZE	r/w	3'b001	
11:0	TransferSize	r/w	0	

6.5.24 DMA_C1Config

Address: 0x4000c210



Bits	Name	Type	Reset	Description
31:19	RSVD			
18	H	r/w	0	
17	A	r	0	
16	L	r/w	0	
15	ITC	r/w	0	
14	IE	r/w	0	
13:11	FlowCntrl	r/w	0	
10:6	DstPeripheral	r/w	0	
5:1	SrcPeripheral	r/w	0	
0	E	r/w	0	

6.5.25 DMA_C2SrcAddr

Address: 0x4000c300

SrcAddr

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

SrcAddr

Bits	Name	Type	Reset	Description
31:0	SrcAddr	r/w	0	

6.5.26 DMA_C2DstAddr

Address: 0x4000c304

DstAddr

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

DstAddr

Bits	Name	Type	Reset	Description
31:0	DstAddr	r/w	0	

6.5.27 DMA_C2LLI

Address: 0x4000c308

LLI

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

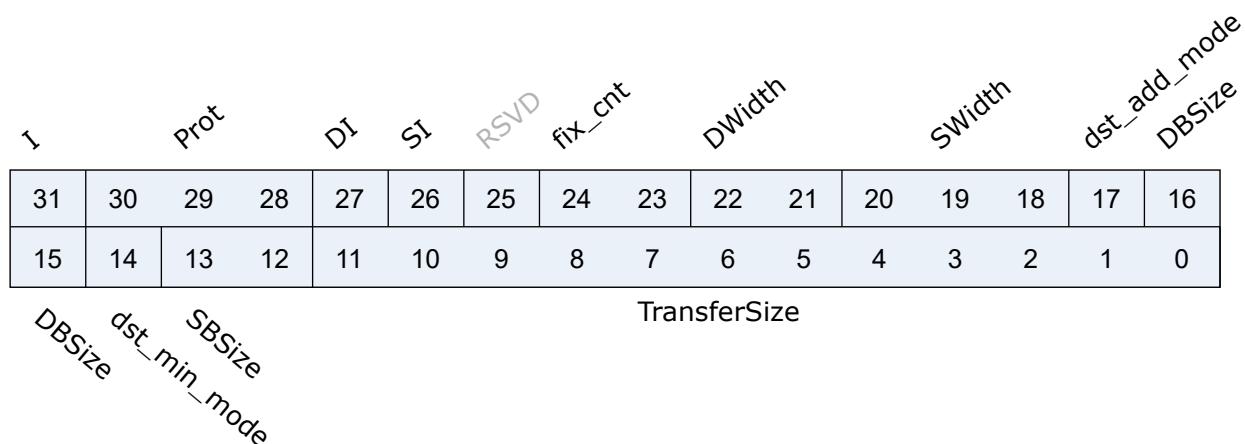
LLI

RSVD RSVD

Bits	Name	Type	Reset	Description
31:2	LLI	r/w	0	
1:0	RSVD			

6.5.28 DMA_C2Control

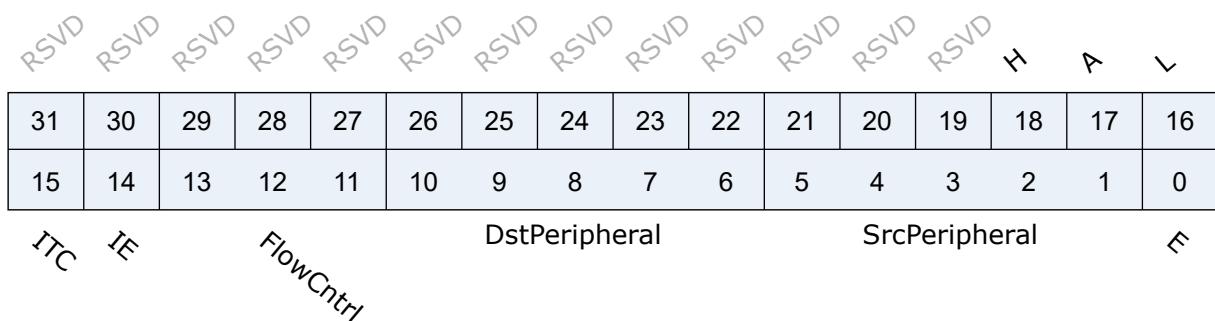
Address: 0x4000c30c



Bits	Name	Type	Reset	Description
31	I	r/w	0	
30:28	Prot	r/w	0	
27	DI	r/w	1	
26	SI	r/w	1	
25	RSVD			
24:23	fix_cnt	r/w	2'd0	
22:24	RSVD			
23:21	DWidth	r/w	3'b010	
20:18	SWidth	r/w	3'b010	
17	dst_add_mode	r/w	1'b0	
16:15	DBSize	r/w	3'b001	
14	dst_min_mode	r/w	1'b0	
13:12	SBSIZE	r/w	3'b001	
11:0	TransferSize	r/w	0	

6.5.29 DMA_C2Config

Address: 0x4000c310



Bits	Name	Type	Reset	Description
31:19	RSVD			
18	H	r/w	0	
17	A	r	0	
16	L	r/w	0	
15	ITC	r/w	0	
14	IE	r/w	0	
13:11	FlowCntrl	r/w	0	
10:6	DstPeripheral	r/w	0	
5:1	SrcPeripheral	r/w	0	
0	E	r/w	0	

6.5.30 DMA_C3SrcAddr

Address: 0x4000c400

SrcAddr															
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

SrcAddr

Bits	Name	Type	Reset	Description
31:0	SrcAddr	r/w	0	

6.5.31 DMA_C3DstAddr

Address: 0x4000c404

DstAddr

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

DstAddr

Bits	Name	Type	Reset	Description
31:0	DstAddr	r/w	0	

6.5.32 DMA_C3LLI

Address: 0x4000c408

LLI

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

LLI

RSVD RSVD

Bits	Name	Type	Reset	Description
31:2	LLI	r/w	0	
1:0	RSVD			

6.5.33 DMA_C3Control

Address: 0x4000c40c

1	Prot	DI	SI	RSVD	fix_cnt	DWidth	SWidth	dst_add_mode	DBSize
31	30	29	28	27	26	25	24	23	22
15	14	13	12	11	10	9	8	7	6

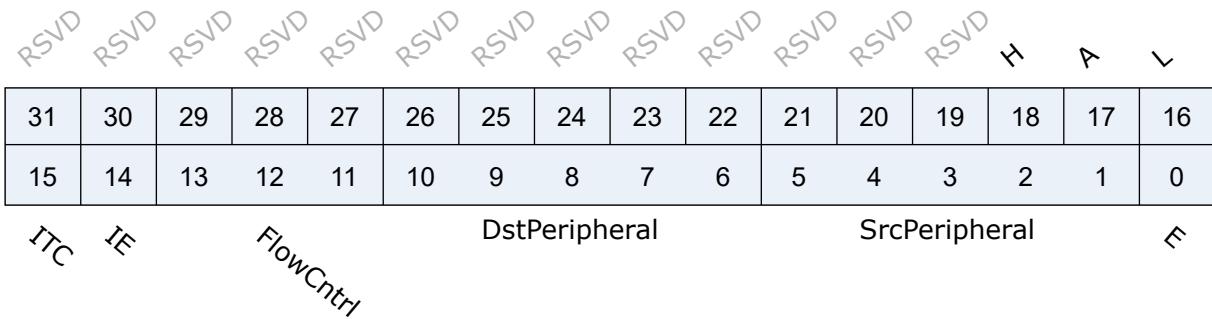
TransferSize

DBSize dst_min_mode SBSIZE

Bits	Name	Type	Reset	Description
31	I	r/w	0	
30:28	Prot	r/w	0	
27	DI	r/w	1	
26	SI	r/w	1	
25	RSVD			
24:23	fix_cnt	r/w	2'd0	
22:24	RSVD			
23:21	DWidth	r/w	3'b010	
20:18	SWidth	r/w	3'b010	
17	dst_add_mode	r/w	1'b0	
16:15	DBSize	r/w	3'b001	
14	dst_min_mode	r/w	1'b0	
13:12	SBSIZE	r/w	3'b001	
11:0	TransferSize	r/w	0	

6.5.34 DMA_C3Config

Address: 0x4000c410



Bits	Name	Type	Reset	Description
31:19	RSVD			
18	H	r/w	0	
17	A	r	0	
16	L	r/w	0	
15	ITC	r/w	0	

Bits	Name	Type	Reset	Description
14	IE	r/w	0	
13:11	FlowCntrl	r/w	0	
10:6	DstPeripheral	r/w	0	
5:1	SrcPeripheral	r/w	0	
0	E	r/w	0	

6.5.35 DMA_C4SrcAddr

Address: 0x4000c500

SrcAddr

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

SrcAddr

Bits	Name	Type	Reset	Description
31:0	SrcAddr	r/w	0	

6.5.36 DMA_C4DstAddr

Address: 0x4000c504

DstAddr

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

DstAddr

Bits	Name	Type	Reset	Description
31:0	DstAddr	r/w	0	

6.5.37 DMA_C4LLI

Address: 0x4000c508

LLI																
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	
LLI																
RSVD RSVD																

Bits	Name				Type	Reset	Description									
31:2	LLI				r/w	0										
1:0	RSVD															

6.5.38 DMA_C4Control

Address: 0x4000c50c

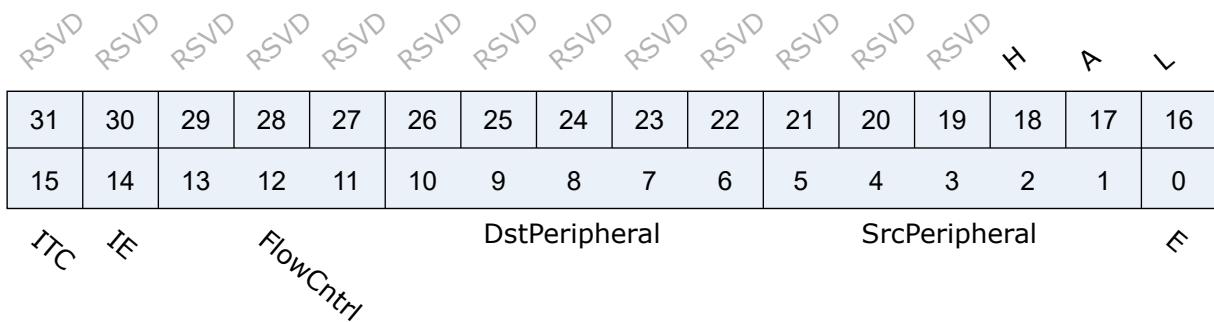
1	Prot	DI	SI	RSVD	fix_cnt	DWidth	SWidth	dst_add_mode	DBSize							
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
TransferSize																
DBSize dst_min_mode SBSIZE																

Bits	Name				Type	Reset	Description									
31	I				r/w	0										
30:28	Prot				r/w	0										
27	DI				r/w	1										
26	SI				r/w	1										
25	RSVD															
24:23	fix_cnt				r/w	2'd0										
22:24	RSVD															
23:21	DWidth				r/w	3'b010										

Bits	Name	Type	Reset	Description
20:18	SWidth	r/w	3'b010	
17	dst_add_mode	r/w	1'b0	
16:15	DBSize	r/w	3'b001	
14	dst_min_mode	r/w	1'b0	
13:12	SBSIZE	r/w	3'b001	
11:0	TransferSize	r/w	0	

6.5.39 DMA_C4Config

Address: 0x4000c510



Bits	Name	Type	Reset	Description
31:19	RSVD			
18	H	r/w	0	
17	A	r	0	
16	L	r/w	0	
15	ITC	r/w	0	
14	IE	r/w	0	
13:11	FlowCntrl	r/w	0	
10:6	DstPeripheral	r/w	0	
5:1	SrcPeripheral	r/w	0	
0	E	r/w	0	

6.5.40 DMA_C5SrcAddr

Address: 0x4000c600

SrcAddr

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

SrcAddr

Bits	Name	Type	Reset	Description
31:0	SrcAddr	r/w	0	

6.5.41 DMA_C5DstAddr

Address: 0x4000c604

DstAddr

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

DstAddr

Bits	Name	Type	Reset	Description
31:0	DstAddr	r/w	0	

6.5.42 DMA_C5LLI

Address: 0x4000c608

LLI

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

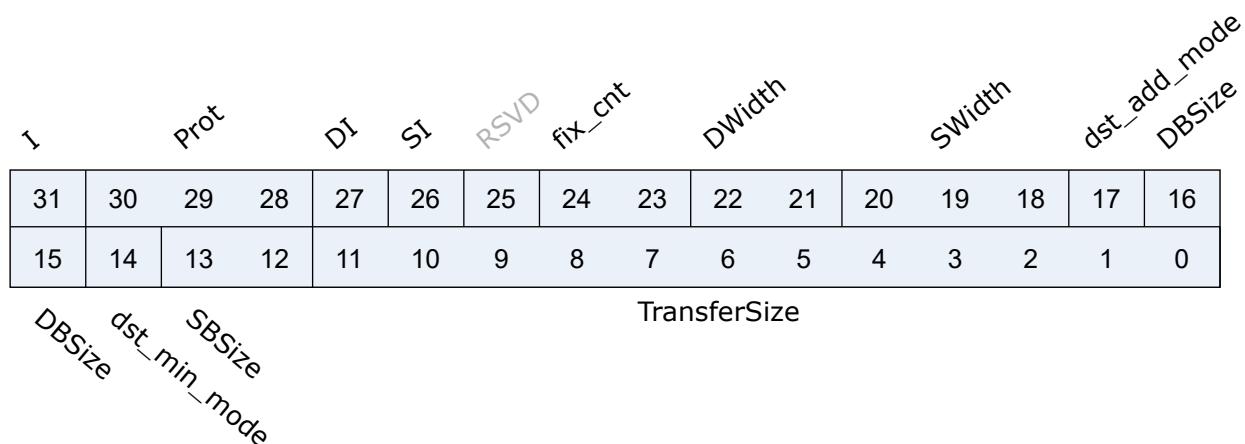
LLI

RSVD RSVD

Bits	Name	Type	Reset	Description
31:2	LLI	r/w	0	
1:0	RSVD			

6.5.43 DMA_C5Control

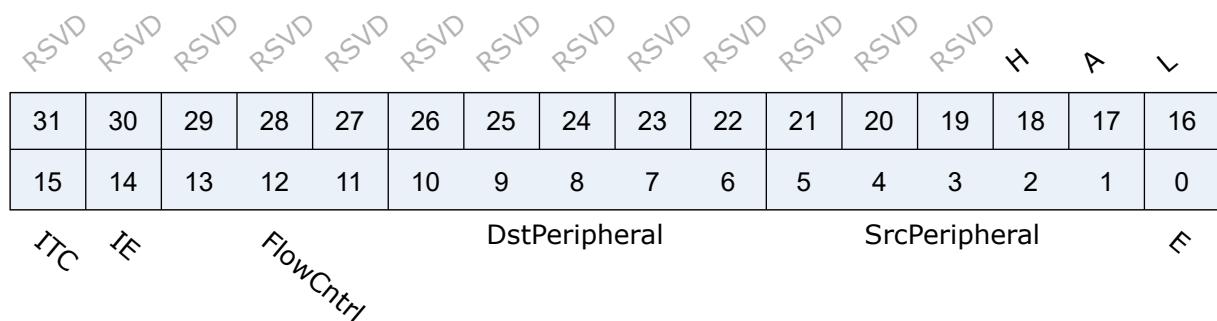
Address: 0x4000c60c



Bits	Name	Type	Reset	Description
31	I	r/w	0	
30:28	Prot	r/w	0	
27	DI	r/w	1	
26	SI	r/w	1	
25	RSVD			
24:23	fix_cnt	r/w	2'd0	
22:24	RSVD			
23:21	DWidth	r/w	3'b010	
20:18	SWidth	r/w	3'b010	
17	dst_add_mode	r/w	1'b0	
16:15	DBSize	r/w	3'b001	
14	dst_min_mode	r/w	1'b0	
13:12	SBSIZE	r/w	3'b001	
11:0	TransferSize	r/w	0	

6.5.44 DMA_C5Config

Address: 0x4000c610



Bits	Name	Type	Reset	Description
31:19	RSVD			
18	H	r/w	0	
17	A	r	0	
16	L	r/w	0	
15	ITC	r/w	0	
14	IE	r/w	0	
13:11	FlowCntrl	r/w	0	
10:6	DstPeripheral	r/w	0	
5:1	SrcPeripheral	r/w	0	
0	E	r/w	0	

6.5.45 DMA_C6SrcAddr

Address: 0x4000c700

SrcAddr

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

SrcAddr

Bits	Name	Type	Reset	Description
31:0	SrcAddr	r/w	0	

6.5.46 DMA_C6DstAddr

Address: 0x4000c704

DstAddr

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

DstAddr

Bits	Name	Type	Reset	Description
31:0	DstAddr	r/w	0	

6.5.47 DMA_C6LLI

Address: 0x4000c708

LLI

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

LLI

RSVD RSVD

Bits	Name	Type	Reset	Description
31:2	LLI	r/w	0	
1:0	RSVD			

6.5.48 DMA_C6Control

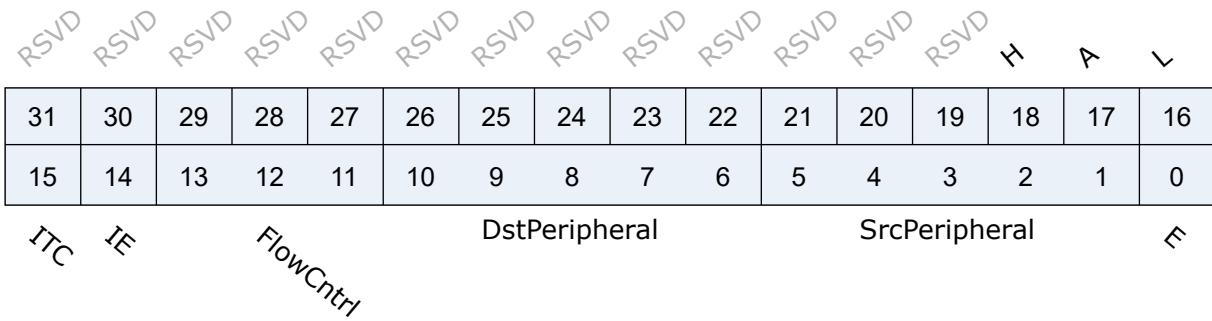
Address: 0x4000c70c

1	Prot	DI	SI	RSVD	fix_cnt	DWidth	SWidth	dst_add_mode	DBSize
31	30	29	28	27	26	25	24	23	22
15	14	13	12	11	10	9	8	7	6
									TransferSize
									DBSize dst_min_mode SBSIZE

Bits	Name	Type	Reset	Description
31	I	r/w	0	
30:28	Prot	r/w	0	
27	DI	r/w	1	
26	SI	r/w	1	
25	RSVD			
24:23	fix_cnt	r/w	2'd0	
22:24	RSVD			
23:21	DWidth	r/w	3'b010	
20:18	SWidth	r/w	3'b010	
17	dst_add_mode	r/w	1'b0	
16:15	DBSize	r/w	3'b001	
14	dst_min_mode	r/w	1'b0	
13:12	SBSIZE	r/w	3'b001	
11:0	TransferSize	r/w	0	

6.5.49 DMA_C6Config

Address: 0x4000c710



Bits	Name	Type	Reset	Description
31:19	RSVD			
18	H	r/w	0	
17	A	r	0	
16	L	r/w	0	
15	ITC	r/w	0	

Bits	Name	Type	Reset	Description
14	IE	r/w	0	
13:11	FlowCntrl	r/w	0	
10:6	DstPeripheral	r/w	0	
5:1	SrcPeripheral	r/w	0	
0	E	r/w	0	

6.5.50 DMA_C7SrcAddr

Address: 0x4000c800

SrcAddr

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

SrcAddr

Bits	Name	Type	Reset	Description
31:0	SrcAddr	r/w	0	

6.5.51 DMA_C7DstAddr

Address: 0x4000c804

DstAddr

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

DstAddr

Bits	Name	Type	Reset	Description
31:0	DstAddr	r/w	0	

6.5.52 DMA_C7LLI

Address: 0x4000c808

LLI																
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	
LLI																
RSVD RSVD																

Bits	Name				Type	Reset	Description									
31:2	LLI				r/w	0										
1:0	RSVD															

6.5.53 DMA_C7Control

Address: 0x4000c80c

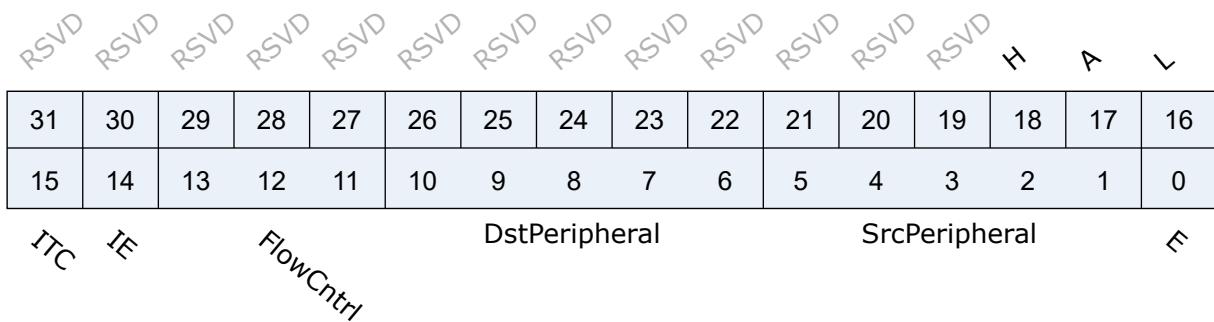
1	Prot	DI	SI	RSVD	fix_cnt	DWidth	SWidth	dst_add_mode	DBSize							
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
TransferSize																
DBSize dst_min_mode SBSIZE																

Bits	Name				Type	Reset	Description									
31	I				r/w	0										
30:28	Prot				r/w	0										
27	DI				r/w	1										
26	SI				r/w	1										
25	RSVD															
24:23	fix_cnt				r/w	2'd0										
22:24	RSVD															
23:21	DWidth				r/w	3'b010										

Bits	Name	Type	Reset	Description
20:18	SWidth	r/w	3'b010	
17	dst_add_mode	r/w	1'b0	
16:15	DBSize	r/w	3'b001	
14	dst_min_mode	r/w	1'b0	
13:12	SBSIZE	r/w	3'b001	
11:0	TransferSize	r/w	0	

6.5.54 DMA_C7Config

Address: 0x4000c810



Bits	Name	Type	Reset	Description
31:19	RSVD			
18	H	r/w	0	
17	A	r	0	
16	L	r/w	0	
15	ITC	r/w	0	
14	IE	r/w	0	
13:11	FlowCntrl	r/w	0	
10:6	DstPeripheral	r/w	0	
5:1	SrcPeripheral	r/w	0	
0	E	r/w	0	

7.1 L1C introduction

L1 Cache Controller is a unit module located outside the processor, used to manage the code or data buffer on Flash/pSRAM and improve the speed of CPU access to Flash/pSRAM. The architecture is as follows:

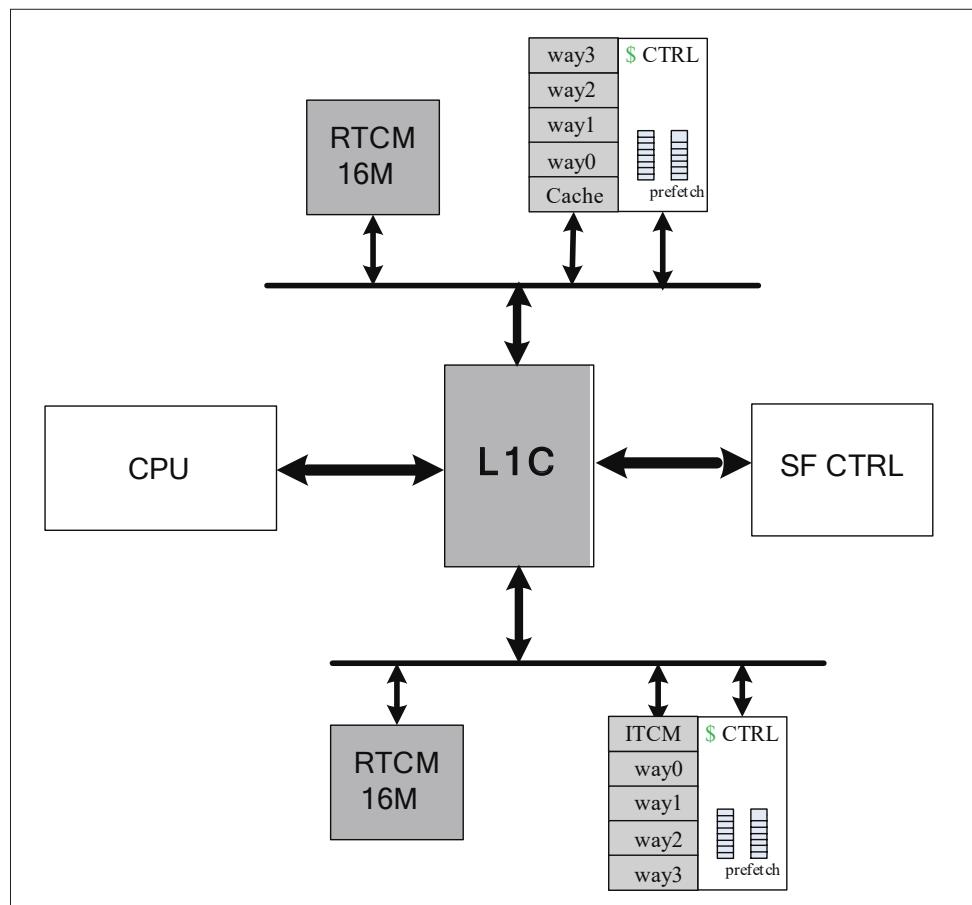


Fig. 7.1: L1c architecture

L1C is a high-speed unit integrated between the processor and Flash. Because the speed of the processor is very fast,

when the processor needs to wait for a long time to access the Flash, the less time wasted, the higher the efficiency. The L1C cache can be used as a lubricating role between the processor and the Flash to improve the efficiency of the processor.

7.2 L1C main features

- 4-way Set-Associative mapping
- Variable cache size
- Connect to TCM address space, can easily configure L1C space as TCM space
- Support cache performance statistics

7.3 L1C function description

7.3.1 Mutual conversion between TCM and Cache RAM resources

In order to increase memory usage efficiency, it is supported to adjust all or part of the Cache's 16K RAM to TCM space, so that users can adjust the memory usage and efficiency according to the actual situation. The maximum Cache can be set to 16K, divided into 4 ways, each way is 4K, and the unit of adjustment is 1 way, which is 4K. The default size of ITCM is 16K. Set by WayDisable. The actual space size of Cache and ITCM can be flexibly adjusted.

Table 7.1: WayDisable settings

WayDisable	Cache	ITCM
none	16K	0K
one way	12K	4K
two way	8K	8K
three way	4K	12K
four way	0K	16K

7.3.2 Cache

The unit of each line buffer is 32 bytes, and the 4-way associative mapping cache is used. The application architecture is as follows:

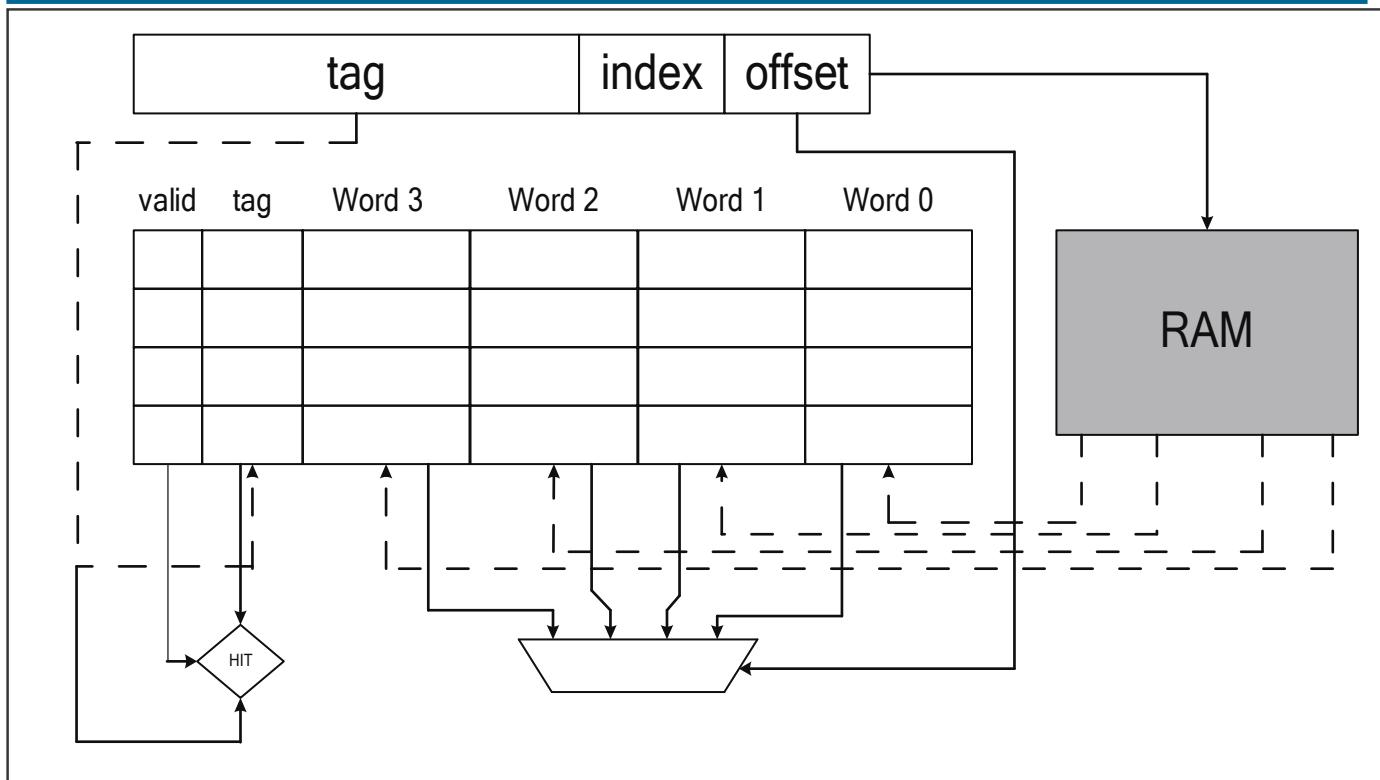


Fig. 7.2: Cache architecture

Each set of associative mapping caches contains two parts, the first is a tag, which contains the valid value and the address mapping relationship. The second part is data storage. When the processor accesses the cache, the cache processor compares the relationship between the address and the tag. When the address comparison is successful, the representative can directly get data from the cache. Conversely, the cache processor will capture related data through the AHB Master and put the data into the cache and respond to the processor's data.

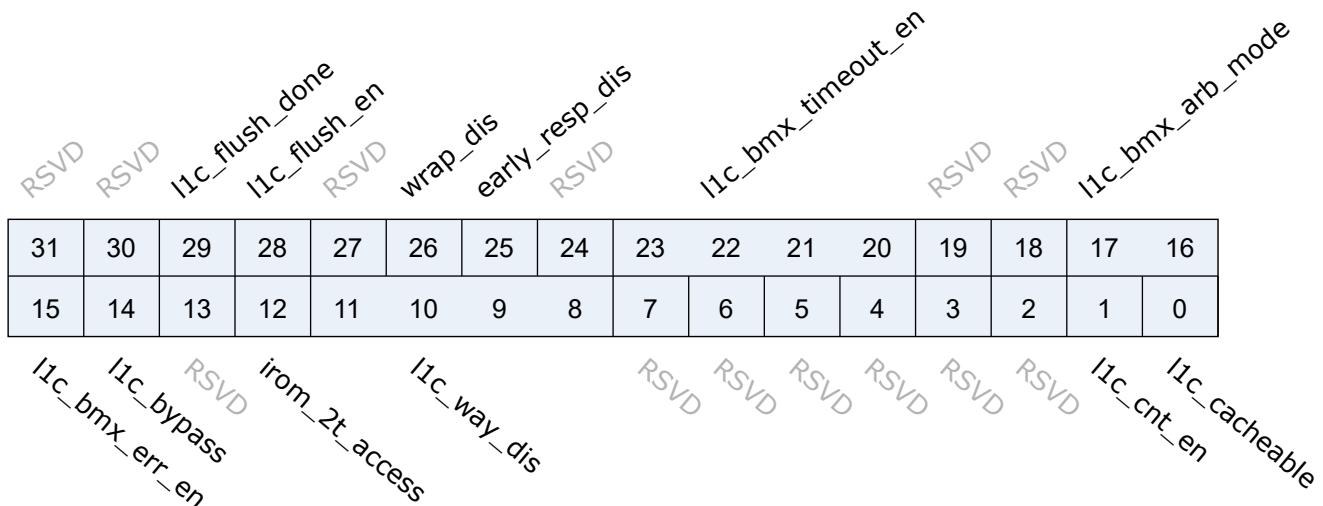
When most of the data can be successfully compared in the tag, the waiting time of the processor can be greatly reduced, and the use efficiency can be increased.

7.4 Register description

Name	Description
l1c_config	L1C configuration
hit_cnt_lsb	Low 32-bit hit counter
hit_cnt_msb	High 32-bit hit counter
miss_cnt	Miss counter

7.4.1 I1c_config

Address: 0x40009000



Bits	Name	Type	Reset	Description
31:30	RSVD			
29	I1c_flush_done	r	0	
28	I1c_flush_en	r/w	0	flush dirty line
27	RSVD			
26	wrap_dis	r/w	1	
25	early_resp_dis	r/w	1	
24	RSVD			
23:20	I1c_bmx_timeout_en	r/w	0	Bus timeout enable: detect slave no reaponse in 1024 cycles
19:18	RSVD			
17:16	I1c_bmx_arb_mode	r/w	0	[1:0] 0:fix, 2:round-robin, 3:random
15	I1c_bmx_err_en	r/w	0	Bus error response enable
14	I1c_bypass	r/w	0	bypass cache ; reset cache
13	RSVD			
12	irom_2t_access	r/w	0	Set 1 for ROM 2T access if CPU freq >72MHz
11:8	I1c_way_dis	r/w	4'b1111	Disable part of cache ways & used as ITCM
7:2	RSVD			
1	I1c_cnt_en	r/w	0	
0	I1c_cacheable	r/w	0	

7.4.2 hit_cnt_lsb

Address: 0x40009004

hit_cnt_lsb

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

hit_cnt_lsb

Bits	Name	Type	Reset	Description
31:0	hit_cnt_lsb	r	0	

7.4.3 hit_cnt_msb

Address: 0x40009008

hit_cnt_msb

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

hit_cnt_msb

Bits	Name	Type	Reset	Description
31:0	hit_cnt_msb	r	0	total hit count = hit_cnt_msb*2 ³² + hit_cnt_lsb

7.4.4 miss_cnt

Address: 0x4000900c

miss_cnt

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

miss_cnt

Bits	Name	Type	Reset	Description
31:0	miss_cnt	r	0	

8.1 IR introduction

Infrared remote (IR for short) is a wireless, non-contact control technology, which has the advantages of strong anti-interference ability, reliable information transmission, low power consumption and low cost. The infrared remote control transmitting circuit uses infrared light emitting diodes to emit modulated infrared light waves. The receiving circuit consists of infrared receiving diodes, triodes or silicon photocells. They convert the infrared light emitted by the infrared transmitter into the corresponding electrical signal and send it to the rear amplifier.

8.2 IR main features

- Receiving data with NEC, RC-5 protocol
- Receiving arbitrary format data in pulse width counting mode
- Powerful infrared waveform editing capabilities, which can emit waveforms conforming to various protocols
- Power settings of up to 15 gears to suit different power requirements
- Supports up to 64-bit data bits
- 64-byte receive FIFO
- Programmable carrier frequency and duty cycle

8.3 IR function description

8.3.1 Fixed receiving protocol

IR receiver supports two fixed protocols, NEC protocol and RC-5 protocol.

- NEC protocol

The logic 1 and logic 0 waveforms of the NEC protocol are shown in the following figure:

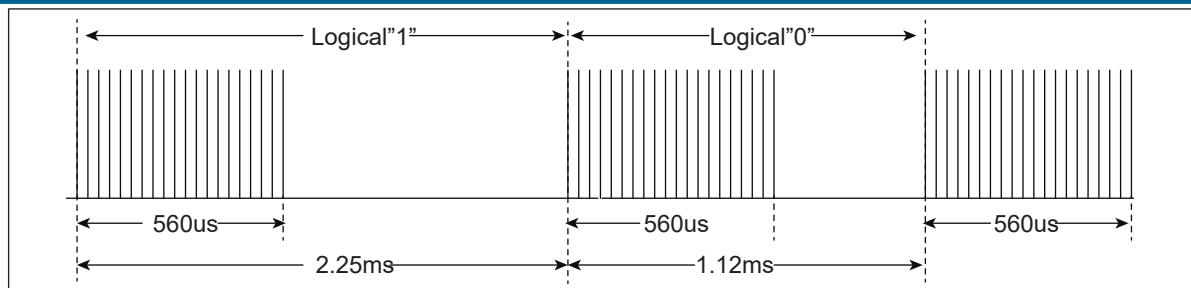


Fig. 8.1: nec logical

Logic 1 is 2.25ms, pulse time is 560us; logic 0 bit is 1.12ms, pulse time is 560us.

The specific format of the NEC protocol is shown in the following figure:

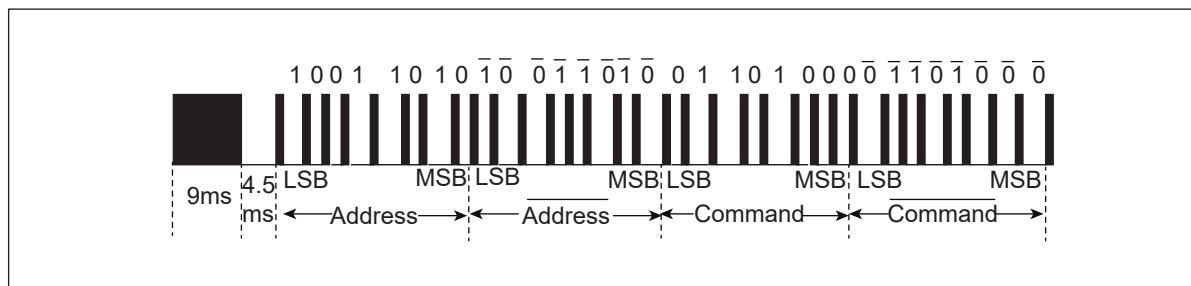


Fig. 8.2: nec

The first pulse is a high-level pulse of 9ms and a low-level of 4.5ms, followed by an 8-bit address code and its inverse code, and then an 8-bit command code and its inverse code. The tail pulse is 560us high and 560us low.

- RC-5 protocol

The logic 1 and logic 0 waveforms of the RC-5 protocol are shown in the following figure:

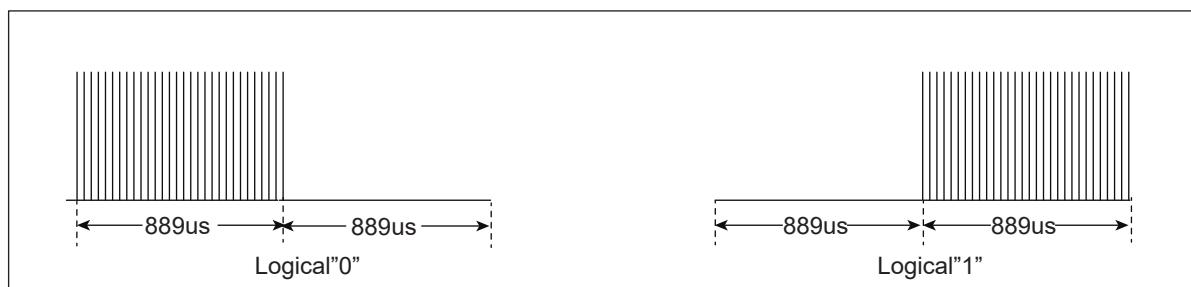


Fig. 8.3: rc5 logical

Logic 1 is 1.778ms, which is 889us low and then 889us high; logic 0 and logic 1 have opposite waveforms.

The specific format of the RC-5 protocol is shown in the following figure:

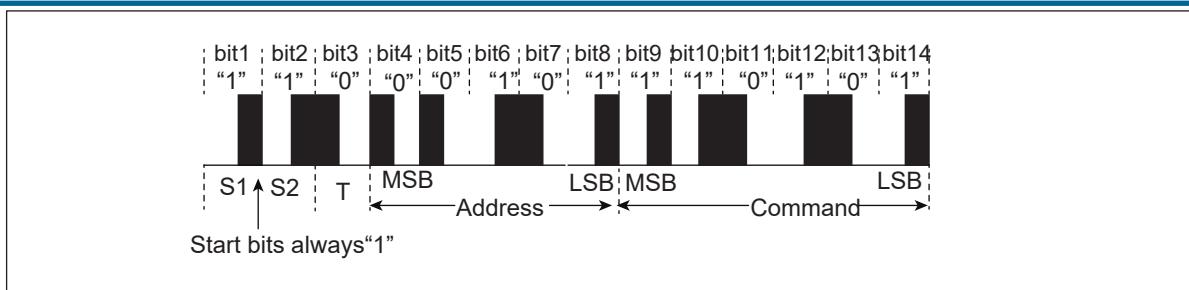


Fig. 8.4: rc5

The first two bits are the start bit, fixed to logic 1, and the third bit is the flip bit. When a key value is issued and then pressed, the bit will be inverted. The next 5 bits are the address code and the 6 bits command code. The first two bits are the start bit, fixed to logic 1, and the third bit is the flip bit. When a key value is issued and then pressed, the bit will be inverted. The next 5 digits are the address code and the 6-digit command code.

It should be noted that in order to improve the receiving sensitivity, the common infrared integrated receiver head outputs a low level after receiving a high level, so when the IR receiving function is used, the receiving flip function must be turned on.

8.3.2 Pulse width reception

For data in any format other than the NEC and RC-5 protocols, the IR will count the duration of each high and low level in turn using its clock, and then store the data in a 64-byte depth receiving FIFO.

8.3.3 Normal sending mode

Users can configure the corresponding configurations of the head pulse, tail pulse, logic 0 and logic 1 pulses according to specific protocols. When setting, it is necessary to calculate the common pulse width unit of various pulses with different widths in the protocol used, that is, the greatest common divisor, fill in the lower 12 bits of the register IRTX_PULSE_WIDTH, and each pulse fills its corresponding multiple in the register IRTX_PW.

IR supports a maximum of 64-bit data bits and is divided into two 32-bit registers IRTX_DATA_WORD0 and IRTX_DATA_WORD1.

8.3.4 Pulse width transmission

For protocols that are not suitable for normal transmission mode, IR provides a pulse width transmission method. First calculate the common pulse width unit of the pulses of different widths in the protocol used, that is, the greatest common divisor, and fill in the lower 12 bits of the register IRTX_PULSE_WIDTH. Then fill the register IRTX_SWM_PW_n($0 \leq n \leq 7$)with multiples corresponding to the respective level widths from the first high level to the last level, each level width multiple occupies 4-bit .

8.3.5 Carrier modulation

Setting the upper 16 bits of the IRTX_PULSE_WIDTH register can generate carriers with different frequencies and duty cycles. The <TXMPH1W> bit in this register sets the width of carrier phase 1, and the <TXMPH0W> bit sets the width of carrier phase 0.

8.3.6 IR interrupt

IR has separate transmit and receive interrupts, and a transmit interrupt is generated when a transmit operation ends. When a piece of data is received, it will wait for the continuous level to reach the set end threshold to generate a receive interrupt.

The user can query the send interrupt status and clear the interrupt by register IRTX_INT_STS, and query the receive interrupt status and clear the interrupt by register IRRX_INT_STS.

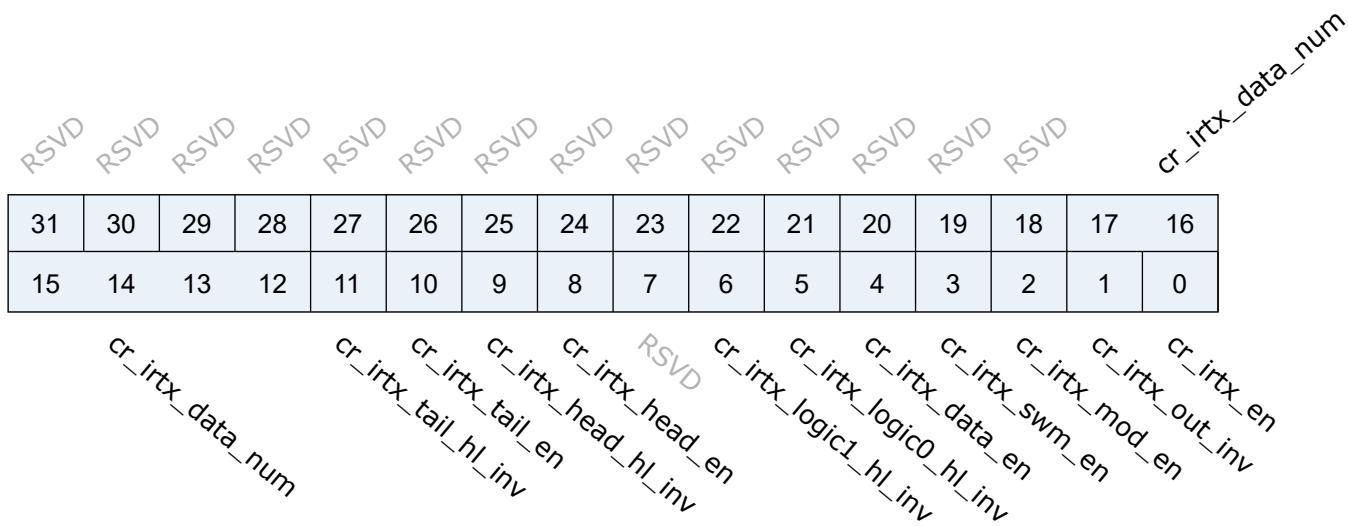
8.4 Register description

Name	Description
irtx_config	IR TX configuration register
irtx_int_sts	IR TX interrupt status
irtx_data_word0	IR TX data word0
irtx_data_word1	IR TX data word1
irtx_pulse_width	IR TX pulse width
irtx_pw	IR TX pulse width of phase
irtx_swm_pw_0	IR TX Software Mode pulse width data0
irtx_swm_pw_1	IR TX Software Mode pulse width data1
irtx_swm_pw_2	IR TX Software Mode pulse width data2
irtx_swm_pw_3	IR TX Software Mode pulse width data3
irtx_swm_pw_4	IR TX Software Mode pulse width data4
irtx_swm_pw_5	IR TX Software Mode pulse width data5
irtx_swm_pw_6	IR TX Software Mode pulse width data6
irtx_swm_pw_7	IR TX Software Mode pulse width data7
irrx_config	IR RX configuration register
irrx_int_sts	IR RX interrupt status
irrx_pw_config	IR RX pulse width configuration

Name	Description
irrx_data_count	IR RX data bit count
irrx_data_word0	IR RX data word0
irrx_data_word1	IR RX data word1
irrx_swm_fifo_config_0	IR RX FIFO configuration
irrx_swm_fifo_rdata	IR RX software mode pulse width data

8.4.1 irtx_config

Address: 0x4000a600

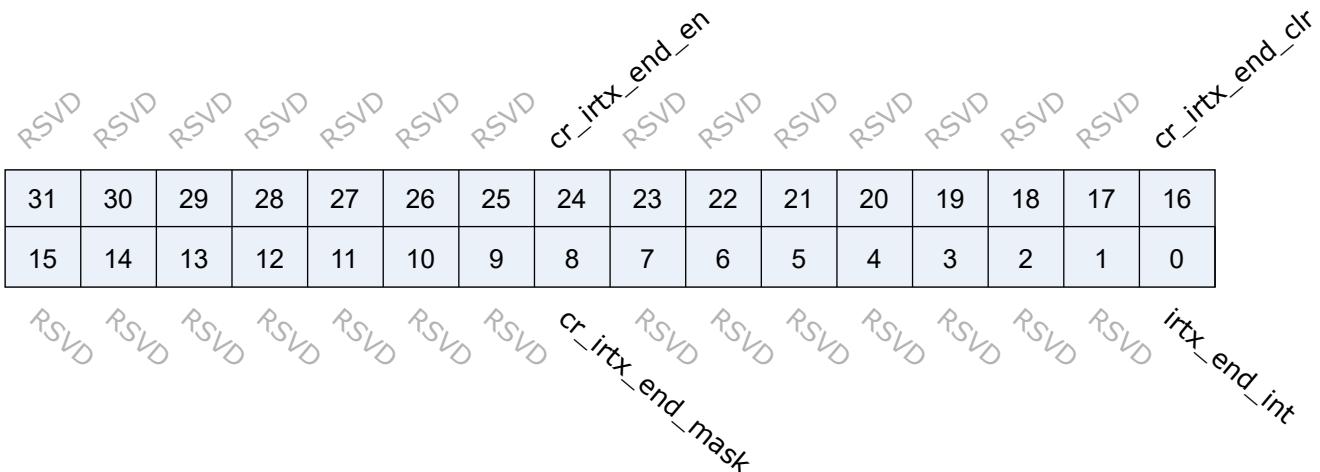


Bits	Name	Type	Reset	Description
31:18	RSVD			
17:12	cr_irtx_data_num	r/w	6'd31	Bit count of Data phase (unit: bit / PW for normal / SWM)
11	cr_irtx_tail_hl_inv	r/w	1'b0	Tail pulse H/L inverse signal (Don't care if SWM is enabled) 0: Phase 0 is High (Active), phase 1 is Low (Idle) (H -> L) 1: Phase 0 is Low (Idle), phase 1 is High (Active) (L -> H)
10	cr_irtx_tail_en	r/w	1'b1	Enable signal of tail pulse (Don't care if SWM is enabled)
9	cr_irtx_head_hl_inv	r/w	1'b0	Tail pulse H/L inverse signal (Don't care if SWM is enabled) 0: Phase 0 is High (Active), phase 1 is Low (Idle) (H -> L) 1: Phase 0 is Low (Idle), phase 1 is High (Active) (L -> H)
8	cr_irtx_head_en	r/w	1'b1	Enable signal of head pulse (Don't care if SWM is enabled)

Bits	Name	Type	Reset	Description
7	RSVD			
6	cr_irtx_logic1_hl_inv	r/w	1'b0	Logic 1 H/L inverse signal (Don't care if SWM is enabled) 0: Phase 0 is High (Active), phase 1 is Low (Idle) (H -> L) 1: Phase 0 is Low (Idle), phase 1 is High (Active) (L -> H)
5	cr_irtx_logic0_hl_inv	r/w	1'b0	Logic 0 H/L inverse signal (Don't care if SWM is enabled) 0: Phase 0 is High (Active), phase 1 is Low (Idle) (H -> L) 1: Phase 0 is Low (Idle), phase 1 is High (Active) (L -> H)
4	cr_irtx_data_en	r/w	1'b1	Enable signal of data phase (Don't care if SWM is enabled)
3	cr_irtx_swm_en	r/w	1'b0	Enable signal of IRTX Software Mode (SWM)
2	cr_irtx_mod_en	r/w	1'b0	Enable signal of output modulation
1	cr_irtx_out_inv	r/w	1'b0	Output inverse signal 1'b0: Output stays at Low during idle state 1'b1: Output stays at High during idle state
0	cr_irtx_en	r/w	1'b0	Enable signal of IRTX function Asserting this bit will trigger the transaction, and should be de-asserted after finish

8.4.2 irtx_int_sts

Address: 0x4000a604



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

Bits	Name	Type	Reset	Description
31:25	RSVD			
24	cr_irtx_end_en	r/w	1'b1	Interrupt enable of irtx_end_int
23:17	RSVD			

Bits	Name	Type	Reset	Description
16	cr_irtx_end_clr	w1c	1'b0	Interrupt clear of irtx_end_int
15:9	RSVD			
8	cr_irtx_end_mask	r/w	1'b1	Interrupt mask of irtx_end_int
7:1	RSVD			
0	irtx_end_int	r	1'b0	IRTX transfer end interrupt

8.4.3 irtx_data_word0

Address: 0x4000a608

cr_irtx_data_word0

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

cr_irtx_data_word0

Bits	Name	Type	Reset	Description
31:0	cr_irtx_data_word0	r/w	32'h0	TX data word 0 (Don't care if SWM is enabled)

8.4.4 irtx_data_word1

Address: 0x4000a60c

cr_irtx_data_word1

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

cr_irtx_data_word1

Bits	Name	Type	Reset	Description
31:0	cr_irtx_data_word1	r/w	32'h0	TX data word 1 (Don't care if SWM is enabled)

8.4.5 irtx_pulse_width

Address: 0x4000a610

cr_irtx_mod_ph1_w										cr_irtx_mod_ph0_w								
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			
RSVD RSVD RSVD RSVD cr_irtx_pw_unit																		

Bits	Name	Type	Reset	Description
31:24	cr_irtx_mod_ph1_w	r/w	8'd34	Modulation phase 1 width
23:16	cr_irtx_mod_ph0_w	r/w	8'd17	Modulation phase 0 width
15:12	RSVD			
11:0	cr_irtx_pw_unit	r/w	12'd1124	Pulse width unit

8.4.6 irtx_pw

Address: 0x4000a614

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
cr_irtx_tail_ph1_w cr_irtx_tail_ph0_w cr_irtx_head_ph1_w cr_irtx_head_ph0_w															
cr_irtx_logic1_ph1_w cr_irtx_logic1_ph0_w cr_irtx_logic0_ph1_w cr_irtx_logic0_ph0_w															

Bits	Name	Type	Reset	Description
31:28	cr_irtx_tail_ph1_w	r/w	4'd0	Pulse width of tail pulse phase 1 (Don't care if SWM is enabled)
27:24	cr_irtx_tail_ph0_w	r/w	4'd0	Pulse width of tail pulse phase 0 (Don't care if SWM is enabled)

Bits	Name	Type	Reset	Description
23:20	cr_irtx_head_ph1_w	r/w	4'd7	Pulse width of head pulse phase 1 (Don't care if SWM is enabled)
19:16	cr_irtx_head_ph0_w	r/w	4'd15	Pulse width of head pulse phase 0 (Don't care if SWM is enabled)
15:12	cr_irtx_logic1_ph1_w	r/w	4'd2	Pulse width of logic1 phase 1 (Don't care if SWM is enabled)
11:8	cr_irtx_logic1_ph0_w	r/w	4'd0	Pulse width of logic1 phase 0 (Don't care if SWM is enabled)
7:4	cr_irtx_logic0_ph1_w	r/w	4'd0	Pulse width of logic0 phase 1 (Don't care if SWM is enabled)
3:0	cr_irtx_logic0_ph0_w	r/w	4'd0	Pulse width of logic0 phase 0 (Don't care if SWM is enabled)

8.4.7 irtx_swm_pw_0

Address: 0x4000a640

cr_irtx_swm_pw_0

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

cr_irtx_swm_pw_0

Bits	Name	Type	Reset	Description
31:0	cr_irtx_swm_pw_0	r/w	32'h0	IRTX Software Mode pulse width data #0 #7, each pulse is represented by 4-bit ([3:0] is the 1st pulse, [7:4] is the 2nd pulse, [11:8] is the 3rd pulse, etc)

8.4.8 irtx_swm_pw_1

Address: 0x4000a644

cr_irtx_swm_pw_1

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

cr_irtx_swm_pw_1

Bits	Name	Type	Reset	Description
31:0	cr_irtx_swm_pw_1	r/w	32'h0	IRTX Software Mode pulse width data #8 #15, each pulse is represented by 4-bit ([3:0] is the 1st pulse, [7:4] is the 2nd pulse, [11:8] is the 3rd pulse, etc)

8.4.9 irtx_swm_pw_2

Address: 0x4000a648

cr_irtx_swm_pw_2

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

cr_irtx_swm_pw_2

Bits	Name	Type	Reset	Description
31:0	cr_irtx_swm_pw_2	r/w	32'h0	IRTX Software Mode pulse width data #16 #23, each pulse is represented by 4-bit ([3:0] is the 1st pulse, [7:4] is the 2nd pulse, [11:8] is the 3rd pulse, etc)

8.4.10 irtx_swm_pw_3

Address: 0x4000a64c

cr_irtx_swm_pw_3

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

cr_irtx_swm_pw_3

Bits	Name	Type	Reset	Description
31:0	cr_irtx_swm_pw_3	r/w	32'h0	IRTX Software Mode pulse width data #24 #31, each pulse is represented by 4-bit ([3:0] is the 1st pulse, [7:4] is the 2nd pulse, [11:8] is the 3rd pulse, etc)

8.4.11 irtx_swm_pw_4

Address: 0x4000a650

cr_irtx_swm_pw_4

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

cr_irtx_swm_pw_4

Bits	Name	Type	Reset	Description
31:0	cr_irtx_swm_pw_4	r/w	32'h0	IRTX Software Mode pulse width data #32 #39, each pulse is represented by 4-bit ([3:0] is the 1st pulse, [7:4] is the 2nd pulse, [11:8] is the 3rd pulse, etc)

8.4.12 irtx_swm_pw_5

Address: 0x4000a654

cr_irtx_swm_pw_5

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

cr_irtx_swm_pw_5

Bits	Name	Type	Reset	Description
31:0	cr_irtx_swm_pw_5	r/w	32'h0	IRTX Software Mode pulse width data #40 #47, each pulse is represented by 4-bit ([3:0] is the 1st pulse, [7:4] is the 2nd pulse, [11:8] is the 3rd pulse, etc)

8.4.13 irtx_swm_pw_6

Address: 0x4000a658

cr_irtx_swm_pw_6

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

cr_irtx_swm_pw_6

Bits	Name	Type	Reset	Description
31:0	cr_irtx_swm_pw_6	r/w	32'h0	IRTX Software Mode pulse width data #48 #55, each pulse is represented by 4-bit ([3:0] is the 1st pulse, [7:4] is the 2nd pulse, [11:8] is the 3rd pulse, etc)

8.4.14 irtx_swm_pw_7

Address: 0x4000a65c

cr_irtx_swm_pw_7

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

cr_irtx_swm_pw_7

Bits	Name	Type	Reset	Description
31:0	cr_irtx_swm_pw_7	r/w	32'h0	IRTX Software Mode pulse width data #56 #63, each pulse is represented by 4-bit ([3:0] is the 1st pulse, [7:4] is the 2nd pulse, [11:8] is the 3rd pulse, etc)

8.4.15 irrx_config

Address: 0x4000a680

RSVD															
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

RSVD RSVD RSVD

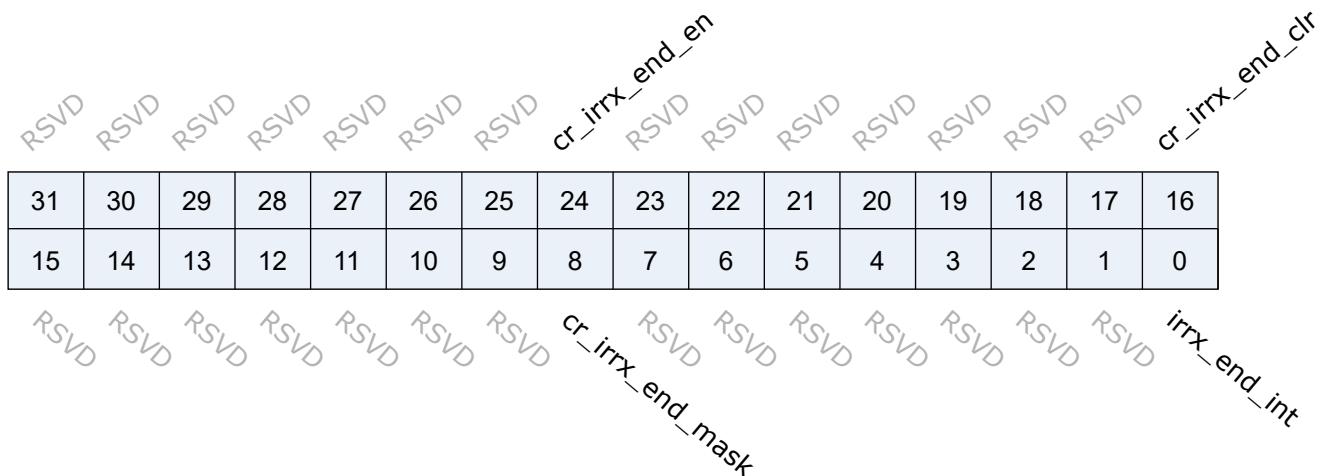
cr_irrx_deg_cnt cr_irrx_deg_en cr_irrx_mode cr_irrx_in_inv

Bits	Name	Type	Reset	Description
31:12	RSVD			
11:8	cr_irrx_deg_cnt	r/w	4'd0	De-glitch function cycle count

Bits	Name	Type	Reset	Description
7:5	RSVD			
4	cr_irrx_deg_en	r/w	1'b0	Enable signal of IRRX input de-glitch function
3:2	cr_irrx_mode	r/w	2'd0	IRRX mode 0: NEC 1: RC5 2: SW pulse-width detection mode (SWM) 3: Reserved
1	cr_irrx_in_inv	r/w	1'b1	Input inverse signal
0	cr_irrx_en	r/w	1'b0	Enable signal of IRRX function Asserting this bit will trigger the transaction, and should be de-asserted after finish

8.4.16 irrx_int_sts

Address: 0x4000a684



Bits	Name	Type	Reset	Description
31:25	RSVD			
24	cr_irrx_end_en	r/w	1'b1	Interrupt enable of irrx_end_int
23:17	RSVD			
16	cr_irrx_end_clr	w1c	1'b0	Interrupt clear of irrx_end_int
15:9	RSVD			
8	cr_irrx_end_mask	r/w	1'b1	Interrupt mask of irrx_end_int
7:1	RSVD			

Bits	Name	Type	Reset	Description
0	irrx_end_int	r	1'b0	IRRX transfer end interrupt

8.4.17 irrx_pw_config

Address: 0x4000a688

cr_irrx_end_th

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

cr_irrx_data_th

Bits	Name	Type	Reset	Description
31:16	cr_irrx_end_th	r/w	16'd8999	Pulse width threshold to trigger END condition
15:0	cr_irrx_data_th	r/w	16'd3399	Pulse width threshold for Logic0/1 detection (Don't care if SWM is enabled)

8.4.18 irrx_data_count

Address: 0x4000a690

RSVD																
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	

sts_irrx_data_cnt

Bits	Name	Type	Reset	Description
31:7	RSVD			
6:0	sts_irrx_data_cnt	r	7'd0	RX data bit count (pulse-width count for SWM)

8.4.19 irrx_data_word0

Address: 0x4000a694

sts_irrx_data_word0

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

sts_irrx_data_word0

Bits	Name	Type	Reset	Description
31:0	sts_irrx_data_word0	r	32'h0	RX data word 0

8.4.20 irrx_data_word1

Address: 0x4000a698

sts_irrx_data_word1

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

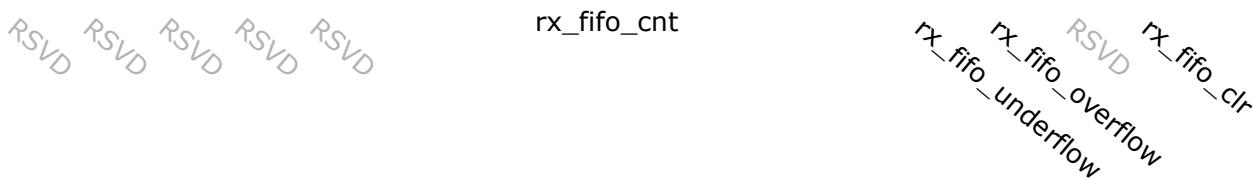
sts_irrx_data_word1

Bits	Name	Type	Reset	Description
31:0	sts_irrx_data_word1	r	32'h0	RX data word 1

8.4.21 irrx_swm_fifo_config_0

Address: 0x4000a6c0

RSVD															
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



Bits	Name	Type	Reset	Description
31:11	RSVD			
10:4	rx_fifo_cnt	r	7'd0	RX FIFO available count
3	rx_fifo_underflow	r	1'b0	Underflow flag of RX FIFO, can be cleared by rx_fifo_clr
2	rx_fifo_overflow	r	1'b0	Overflow flag of RX FIFO, can be cleared by rx_fifo_clr
1	RSVD			
0	rx_fifo_clr	w1c	1'b0	Clear signal of RX FIFO

8.4.22 irrx_swm_fifo_rdata

Address: 0x4000a6c4

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | | | |

rx_fifo_rdata

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	rx_fifo_rdata	r	16'h0	IRRX Software Mode pulse width data

9.1 SPI introduction

Serial Peripheral Interface Bus(SPI) is a synchronous serial communication interface specification for short-range communication. Devices use full-duplex mode for communication. There is a master and one or more slaves. Requires at least 4 wires, in fact 3 wires are also available (the one-way transmission), including SDI (data input), SDO (data output), SCLK (clock), CS (chip select).

9.2 SPI main features

- Can be used as SPI master or SPI slave
- The transmit and receive channels each have a FIFO with a depth of 4 frames
- Both master and slave devices support 4 clock formats(CPOL,CPHA)
- Both master and slave devices support 1/2/3/4 byte transmission mode
- Flexible clock configuration, support up to 40M clock
- Configurable MSB/LSB priority transmission
- Acceptance filtering function
- Timeout mechanism under the slave
- Support DMA transfer mode

9.3 SPI function description

9.3.1 Clock control

According to different clock phases and polarity settings, the SPI clock has four modes, which can be set by bit4 (CPOL) and bit5 (CPHA) of the SPI_CONFIG register. CPOL is used to determine the level of the SCK clock signal when idle, CPOL = 0 means the idle level is low, and CPOL = 1 means the idle level is high. CPHA is used to determine the sampling time. CPHA = 0 samples on the first clock edge of each cycle, and CPHA = 1 samples on the second clock edge of each cycle.

By setting registers SPI_PRD_0 and SPI_PRD_1, you can also adjust the start and end level duration of the clock, the time of phase 0/1, and the interval between each frame of data. The specific settings in the four modes are shown below:

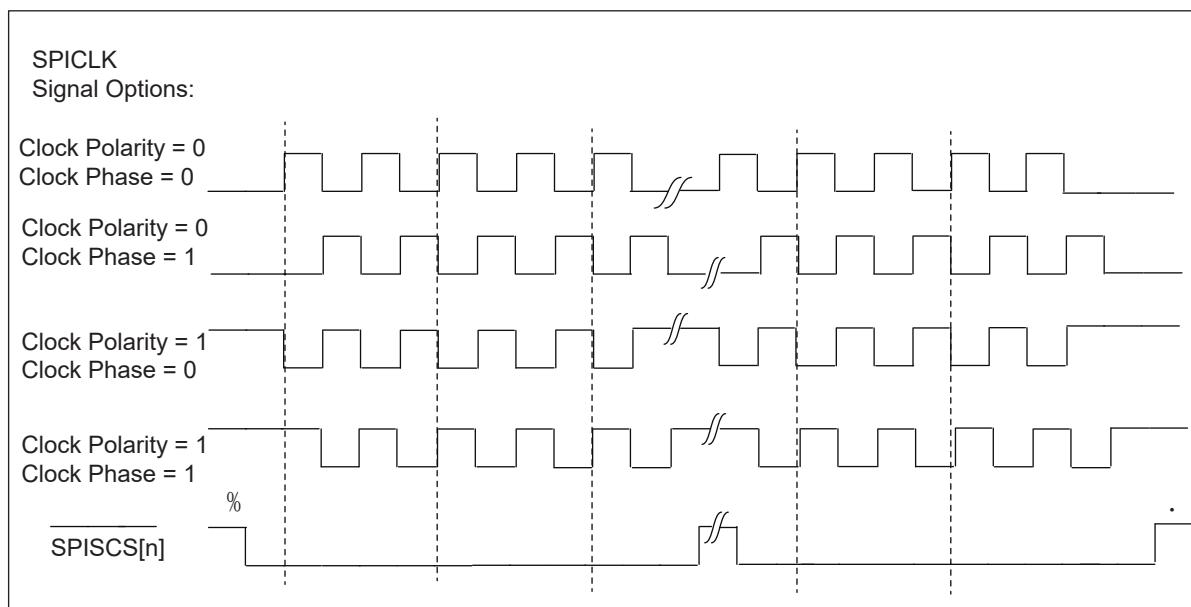


Fig. 9.1: SPI clock

The meaning of each number is as follows: 1 is the length of the start condition, 2 is the length of the stop condition, 3 is the length of phase 0, 4 is the length of phase 1, and 5 is the interval between each frame of data.

9.3.2 Master continuous transmission mode

When this mode is enabled, the CS signal will not be released when the current data is transmitted and there is still data available in the FIFO.

9.3.3 Acceptance filtering function

By setting the start and end bits that need to be filtered out, the SPI discards the corresponding data segment in the received data. As shown below:

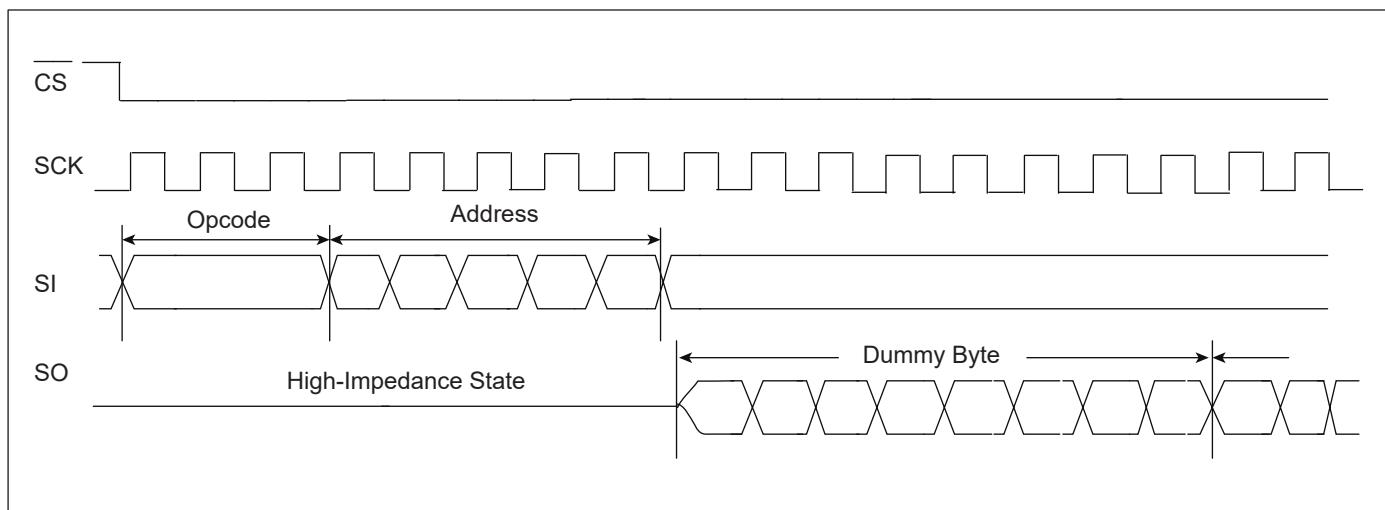


Fig. 9.2: SPI Ignore waveform

In the figure above, the start bit of the filter is set to 0, the end bit is set to 7, the dummy byte is received, and the end bit is set to 15, the dummy byte is discarded.

9.3.4 Receive error correction

By enabling this function and setting the threshold, the SPI will discard data that does not reach the threshold width.

9.3.5 Slave mode timeout mechanism

By setting a timeout threshold, an interrupt will be triggered when the SPI does not receive a clock signal after exceeding this time value in slave mode.

9.3.6 I/O transfer mode

The chip communications processor can perform FIFO fill and empty operations in response to interrupts from the FIFO. Each FIFO has a programmable FIFO trigger threshold to trigger interrupts. When the value in the RX FIFO exceeds the RX FIFO trigger threshold in the SPI controller 1, an interrupt will be generated and a signal will be sent to the chip communication processor to clear the RX FIFO. When the value in the TX FIFO is less than or equal to the TX FIFO trigger threshold in the SPI control register 1 plus 1, an interrupt will be generated and a signal will be sent to the chip communication processor to refill the TX FIFO.

Query the SPI status register to determine the sampled value in the FIFO and the status of the FIFO. Software is responsible for ensuring the correct RX FIFO trigger threshold and TX FIFO trigger threshold to prevent receive FIFO overrun and transmit FIFO underrun.

9.3.7 DMA transfer mode

SPI supports DMA transfer mode. The use of this mode requires the TX and RX FIFO thresholds to be set separately. When this mode is enabled, the UART will check the TX / RX FIFO. Once the TX / RX FIFO available count value is greater than its set threshold, a DMA request will be initiated , DMA will move data to TX FIFO or out of RX FIFO according to the setting.

9.3.8 SPI interrupt

SPI has a variety of interrupt control, including the following interrupt modes:

- SPI transfer end interrupt
- TX FIFO request interrupt
- RX FIFO request interrupt
- Slave mode transfer timeout interrupt
- Slave mode TX overload interrupt
- TX / RX FIFO overflow interrupt

In master mode, the SPI transfer end interrupt is triggered at the end of each frame of data transfer; in slave mode, the SPI transfer end interrupt is triggered when the CS signal is released. The TX / RX FIFO request interrupt will be triggered when its available FIFO count is greater than its set threshold. When the condition is not met, the interrupt flag will be automatically cleared. Slave mode transmission timeout interrupt is triggered when the threshold is exceeded in slave mode and no clock signal is received. If the TX / RX FIFO overflows or underflows, the TX / RX FIFO overflow interrupt will be triggered. When the FIFO clear bit TFC / RFC is set to 1, the corresponding FIFO will be cleared and the overflow interrupt flag will be automatically cleared.

Query the interrupt status through register SPI_INT_STS and write 1 to the corresponding bit to clear the interrupt.

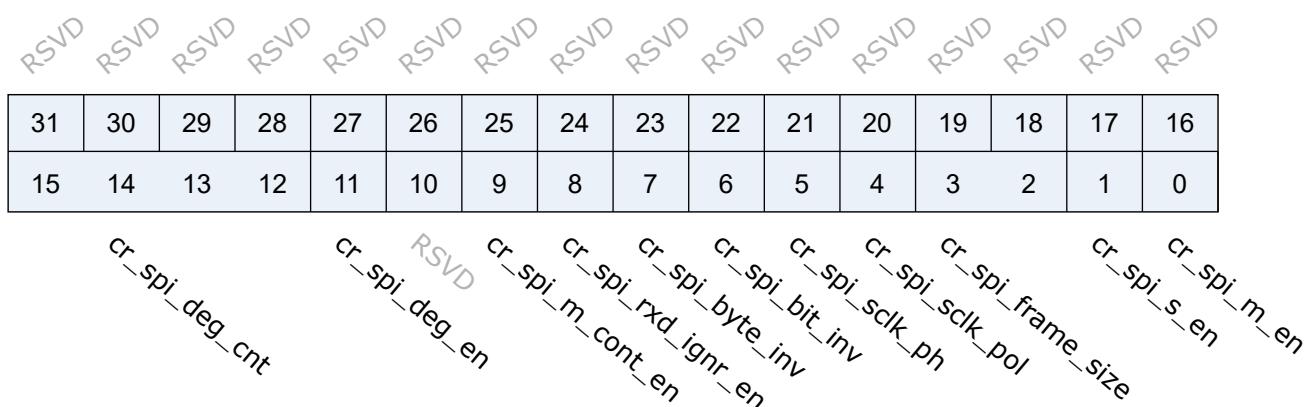
9.4 Register description

Name	Description
spi_config	SPI configuration register
spi_int_sts	SPI interrupt status
spi_bus_busy	SPI bus busy
spi_prd_0	SPI length control register
spi_prd_1	SPI length of interval
spi_rxd_ignr	SPI ignore function
spi_sto_value	SPI time-out value

Name	Description
spi_fifo_config_0	SPI FIFO configuration register0
spi_fifo_config_1	SPI FIFO configuration register1
spi_fifo_wdata	SPI FIFO write data
spi_fifo_rdata	SPI FIFO read data

9.4.1 spi_config

Address: 0x4000a200



Bits	Name	Type	Reset	Description
31:16	RSVD			
15:12	cr_spi_deg_cnt	r/w	4'd0	De-glitch function cycle count
11	cr_spi_deg_en	r/w	1'b0	Enable signal of all input de-glitch function
10	RSVD			
9	cr_spi_m_cont_en	r/w	1'b0	Enable signal of master continuous transfer mode 1'b0: Disabled, SS_n will de-assert between each data frame 1'b1: Enabled, SS_n will stay asserted between each consecutive data frame if the next data is valid in the FIFO
8	cr_spi_rxd_ignr_en	r/w	1'b0	Enable signal of RX data ignore function
7	cr_spi_byte_inv	r/w	1'b0	Byte-inverse signal for each FIFO entry data 0: Byte[0] is sent out first 1: Byte[3] is sent out first

Bits	Name	Type	Reset	Description
6	cr_spi_bit_inv	r/w	1'b0	Bit-inverse signal for each data byte 0: Each byte is sent out MSB-first 1: Each byte is sent out LSB-first
5	cr_spi_sclk_ph	r/w	1'b0	SCLK clock phase inverse signal
4	cr_spi_sclk_pol	r/w	1'b0	SCLK polarity 0: SCLK output LOW at IDLE state 1: SCLK output HIGH at IDLE state
3:2	cr_spi_frame_size	r/w	2'd0	SPI frame size (also the valid width for each FIFO entry) 2'd0: 8-bit 2'd1: 16-bit 2'd2: 24-bit 2'd3: 32-bit
1	cr_spi_s_en	r/w	1'b0	Enable signal of SPI Slave function, Master and Slave should not be both enabled at the same time (This bit becomes don't-care if cr_spi_m_en is enabled)
0	cr_spi_m_en	r/w	1'b0	Enable signal of SPI Master function Asserting this bit will trigger the transaction, and should be de-asserted after finish

9.4.2 spi_int_sts

Address: 0x4000a204

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	
RSVD	RSVD	cr_spi_fer_en	cr_spi_txu_en	cr_spi_sto_en	cr_spi_rxf_en	cr_spi_txf_en	cr_spi_end_en	RSVD	RSVD	RSVD	cr_spi_txu_clr	cr_spi_sto_clr	RSVD	RSVD	cr_spi_end_clr	
RSVD	RSVD	cr_spi_fer_mask	cr_spi_txu_mask	cr_spi_sto_mask	cr_spi_rxf_mask	cr_spi_txf_mask	cr_spi_end_mask	RSVD	RSVD	RSVD	spi_fer_int	spi_txu_int	spi_sto_int	spi_rxf_int	spi_txf_int	spi_end_int

Bits	Name	Type	Reset	Description
31:30	RSVD			
29	cr_spi_fer_en	r/w	1'b1	Interrupt enable of spi_fer_int

Bits	Name	Type	Reset	Description
28	cr_spi_txu_en	r/w	1'b1	Interrupt enable of spi_txu_int
27	cr_spi_sto_en	r/w	1'b1	Interrupt enable of spi_sto_int
26	cr_spi_rxf_en	r/w	1'b1	Interrupt enable of spi_rxv_int
25	cr_spi_txf_en	r/w	1'b1	Interrupt enable of spi_txe_int
24	cr_spi_end_en	r/w	1'b1	Interrupt enable of spi_end_int
23:21	RSVD			
20	cr_spi_txu_clr	w1c	1'b0	Interrupt clear of spi_txu_int
19	cr_spi_sto_clr	w1c	1'b0	Interrupt clear of spi_sto_int
18:17	RSVD			
16	cr_spi_end_clr	w1c	1'b0	Interrupt clear of spi_end_int
15:14	RSVD			
13	cr_spi_fer_mask	r/w	1'b1	Interrupt mask of spi_fer_int
12	cr_spi_txu_mask	r/w	1'b1	Interrupt mask of spi_txu_int
11	cr_spi_sto_mask	r/w	1'b1	Interrupt mask of spi_sto_int
10	cr_spi_rxf_mask	r/w	1'b1	Interrupt mask of spi_rxv_int
9	cr_spi_txf_mask	r/w	1'b1	Interrupt mask of spi_txe_int
8	cr_spi_end_mask	r/w	1'b1	Interrupt mask of spi_end_int
7:6	RSVD			
5	spi_fer_int	r	1'b0	SPI TX/RX FIFO error interrupt, auto-cleared when FIFO overflow/underflow error flag is cleared
4	spi_txu_int	r	1'b0	SPI slave mode TX underrun error flag, triggered when TXD is not ready during transfer in slave mode
3	spi_sto_int	r	1'b0	SPI slave mode transfer time-out interrupt, triggered when SPI bus is idle for a given value
2	spi_rxf_int	r	1'b0	SPI RX FIFO ready (rx_fifo_cnt > rx_fifo_th) interrupt, auto-cleared when data is popped
1	spi_txf_int	r	1'b0	SPI TX FIFO ready (tx_fifo_cnt > tx_fifo_th) interrupt, auto-cleared when data is pushed
0	spi_end_int	r	1'b0	SPI transfer end interrupt, shared by both master and slave mode Master mode: Triggered when the final frame is transferred Slave mode: Triggered when CS_n is de-asserted

9.4.3 spi_bus_busy

Address: 0x4000a208

Bits	Name	Type	Reset	Description
31:1	RSVD			
0	sts_spi_bus_busy	r	1'b0	Indicator of SPI bus busy

9.4.4 spi_prd_0

Address: 0x4000a210

Bits	Name	Type	Reset	Description
31:24	cr_spi_prd_d_ph_1	r/w	8'd15	Length of DATA phase 1 (please refer to "Timing" tab)
23:16	cr_spi_prd_d_ph_0	r/w	8'd15	Length of DATA phase 0 (please refer to "Timing" tab)
15:8	cr_spi_prd_p	r/w	8'd15	Length of STOP condition (please refer to "Timing" tab)
7:0	cr_spi_prd_s	r/w	8'd15	Length of START condition (please refer to "Timing" tab)

9.4.5 spi_prd_1

Address: 0x4000a214

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | |

cr_spi_prd_i

Bits	Name	Type	Reset	Description
31:8	RSVD			
7:0	cr_spi_prd_i	r/w	8'd15	Length of INTERVAL between frame (please refer to "Timing" tab)

9.4.6 spi_rxd_ignr

Address: 0x4000a218

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | |

cr_spi_rxd_ignr_s

cr_spi_rxd_ignr_p

Bits	Name	Type	Reset	Description
31:21	RSVD			
20:16	cr_spi_rxd_ignr_s	r/w	5'd0	Starting point of RX data ignore function
15:5	RSVD			
4:0	cr_spi_rxd_ignr_p	r/w	5'd0	Stopping point of RX data ignore function

9.4.7 spi_sto_value

Address: 0x4000a21c

RSVD															
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

cr_spi_sto_value

Bits	Name	Type	Reset	Description
31:12	RSVD			
11:0	cr_spi_sto_value	r/w	12'hFFF	Time-out value for spi_sto_int triggering

9.4.8 spi_fifo_config_0

Address: 0x4000a280

RSVD															
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

rx_fifo_underflow *rx_fifo_overflow* *tx_fifo_underflow* *tx_fifo_overflow* *tx_fifo_clr* *spi_dma_tx_en* *spi_dma_rx_en*

Bits	Name	Type	Reset	Description
31:8	RSVD			
7	rx_fifo_underflow	r	1'b0	Underflow flag of RX FIFO, can be cleared by rx_fifo_clr
6	rx_fifo_overflow	r	1'b0	Overflow flag of RX FIFO, can be cleared by rx_fifo_clr
5	tx_fifo_underflow	r	1'b0	Underflow flag of TX FIFO, can be cleared by tx_fifo_clr
4	tx_fifo_overflow	r	1'b0	Overflow flag of TX FIFO, can be cleared by tx_fifo_clr
3	rx_fifo_clr	w1c	1'b0	Clear signal of RX FIFO
2	tx_fifo_clr	w1c	1'b0	Clear signal of TX FIFO
1	spi_dma_rx_en	r/w	1'b0	Enable signal of dma_rx_req/ack interface

Bits	Name	Type	Reset	Description
0	spi_dma_tx_en	r/w	1'b0	Enable signal of dma_tx_req/ack interface

9.4.9 spi_fifo_config_1

Address: 0x4000a284

RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	rx_fifo_th	RSVD	RSVD	RSVD	RSVD	RSVD	tx_fifo_th
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



Bits	Name	Type	Reset	Description
31:26	RSVD			
25:24	rx_fifo_th	r/w	2'd0	RX FIFO threshold, dma_rx_req will not be asserted if tx_fifo_cnt is less than this value
23:18	RSVD			
17:16	tx_fifo_th	r/w	2'd0	TX FIFO threshold, dma_tx_req will not be asserted if tx_fifo_cnt is less than this value
15:11	RSVD			
10:8	rx_fifo_cnt	r	3'd0	RX FIFO available count
7:3	RSVD			
2:0	tx_fifo_cnt	r	3'd4	TX FIFO available count

9.4.10 spi_fifo_wdata

Address: 0x4000a288

spi_fifo_wdata

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

spi_fifo_wdata

Bits	Name	Type	Reset	Description
31:0	spi_fifo_wdata	w	x	

9.4.11 spi_fifo_rdata

Address: 0x4000a28c

spi_fifo_rdata

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

spi_fifo_rdata

Bits	Name	Type	Reset	Description
31:0	spi_fifo_rdata	r	32'h0	

10

UART

10.1 UART introduction

Universal Asynchronous Receiver / Transmitter (commonly known as UART) is an asynchronous transceiver that provides a flexible way to exchange full-duplex data with external devices.

BL70x has two sets of UART ports (UART0 and UART1). By using with DMA, you can achieve efficient data communication.

10.2 UART main features

- Full-duplex asynchronous communication
- Data bit length can be selected from 5/6/7/8 bits
- Stop bit length can be selected from 0.5/1/1.5/2 bits
- Supports odd/even/no parity bits
- Detects wrong start bit
- Support LIN protocol (receive and send REAK/SYNC)
- Multiple interrupt control
- Support hardware flow control (RTS / CTS)
- Convenient baud rate programming
- Configurable MSB / LSB priority transmission
- Normal / fixed character automatic baud rate detection
- 32-byte transmit / receive FIFO
- Support DMA transfer mode

10.3 UART function description

10.3.1 Data format description

Normal UART communication data is composed of a start bit, a data bit, a parity bit, and a stop bit. The BL702's UART supports configurable data bits, parity bits, and stop bits, all of which are set in the utx_config and urx_config registers. The waveform of one frame of data is shown below:

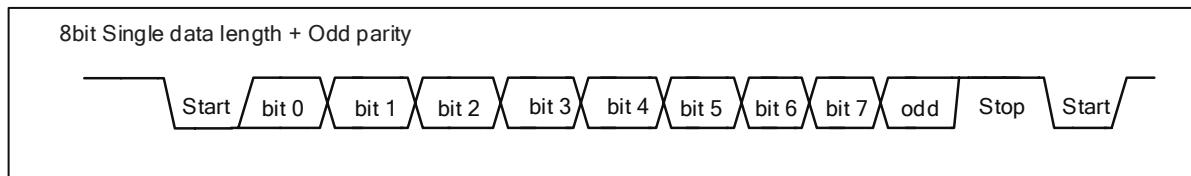


Fig. 10.1: UART data

The start bit of the data frame occupies 1 bit, and the stop bit can be 0.5/1/1.5/2-bit wide by configuring the cr_utx_bit_cnt_p bit in the register utx_config. The start bit is at a low level and the stop bit is at a high level. The data bit width can be set to 5/6/7/8-bit by the cr_utx_bit_cnt_d bit in the register utx_config.

When the cr_utx_prt_en bit in the register utx_config and the cr_urx_prt_en bit in the register urx_config are set, the data frame adds a parity check bit after the data. The cr_utx_prt_sel bit in the register utx_config and the cr_urx_prt_sel bit in the register urx_config are used to select odd or even parity check. When the receiver detects the check bit error of the input data, it will generate the check error interrupt. However, the received data will still be stored into the FIFO.

Calculation method of odd parity check: If there is an odd number of “1” in the current data bit, the odd parity check bit is set to 0. Otherwise, it is set to 1.

Calculation method of even parity check: If there is an odd number of “1” in the current data bit, the even parity check bit is set to 1. Otherwise, it is set to 0.

10.3.2 Clock source

The UART has two clock sources: 96MHz PLL_CLK and FCLK. The frequency divider in the clock is used to divide the clock source and then generate a clock signal to drive the UART module. As shown below:

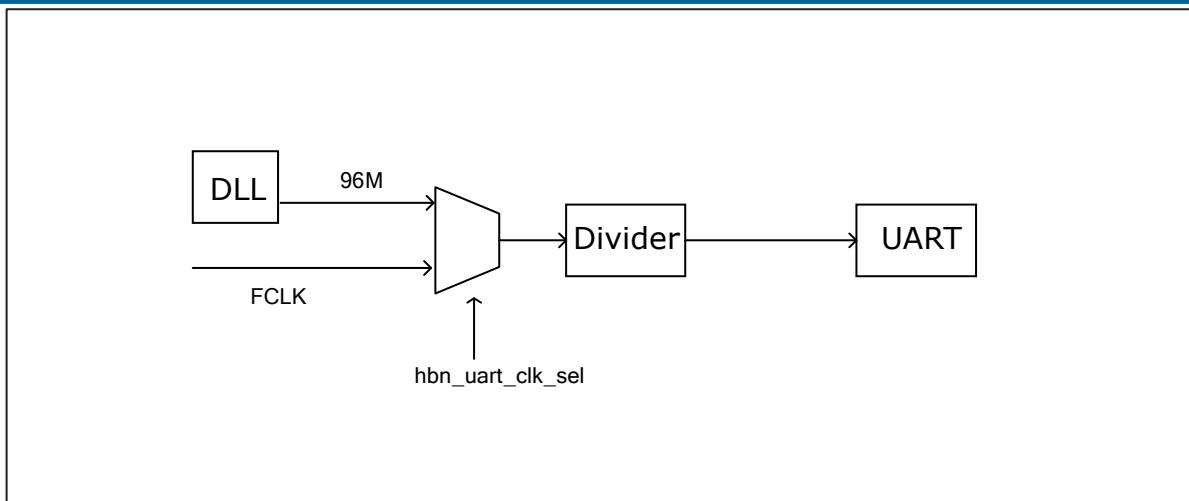


Fig. 10.2: UART clock

10.3.3 Baud rate setting

$$\text{Baudrate} = \frac{\text{UART_clk}}{\text{uart_prd} + 1}$$

The user can set the baud rate of RX and TX separately. Take TX as an example: the value of uart_prd is the value of the lower 16 bits cr_utx_bit_prd of the register UART_BIT_PRD. Since the maximum value of the 16-bit bit width coefficient is 65535, the minimum baud rate supported by UART is : $\text{UART_clk}/65536$.

Before sampling the data, UART will filter the data to remove the burrs in the waveform. Then, the data will be sampled at the intermediate value of the 16-bit width factor, so that the sampling time can be adjusted based on baud rates, to ensure that the intermediate value is always sampled, providing much higher flexibility and accuracy. The sampling process is shown as follows:

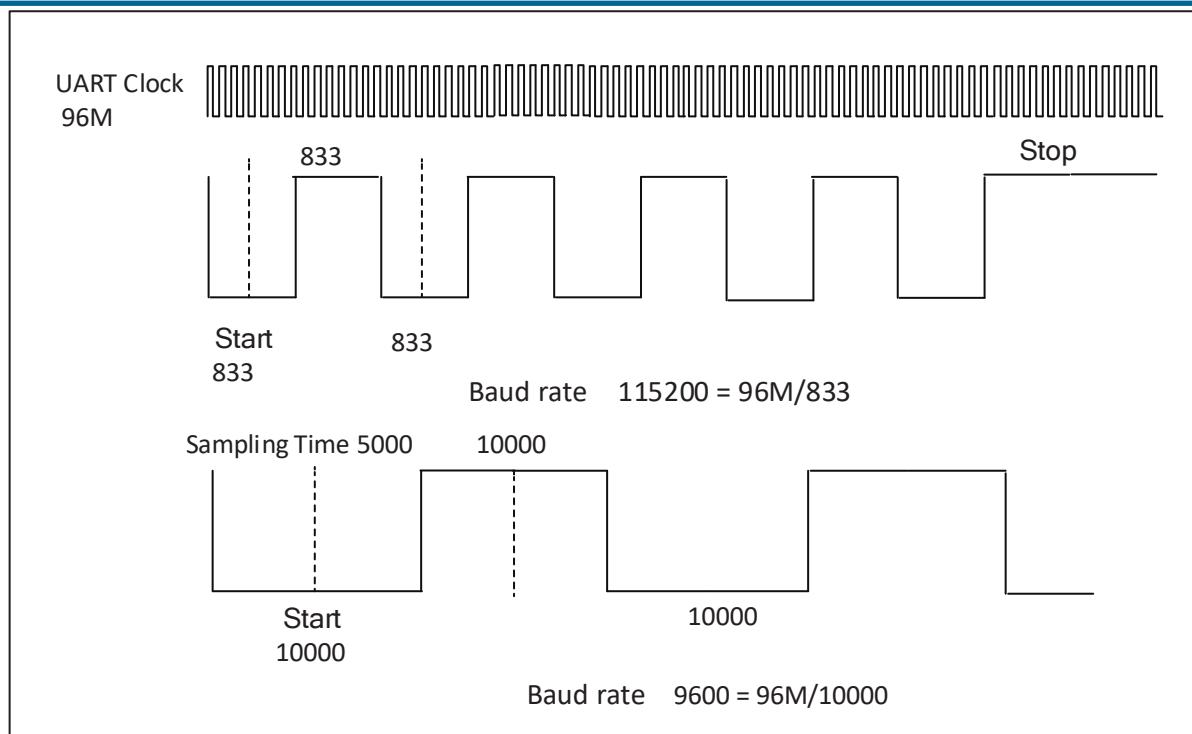


Fig. 10.3: UART sampling waveform

10.3.4 Transmitter

The transmitter contains a 128-byte transmit FIFO to store the data to be transmitted. Software can write TX FIFO through APB bus, and can also move data into TX FIFO through DMA. When the transmit enable bit is set, the data stored in the FIFO will be output from the TX pin. Software can choose to transfer data to TX FIFO through DMA or APB bus. Software can check the status of transmitter by querying the remaining free space count value of TX FIFO through tx_fifo_cnt in the register uart_fifo_config_1.

FreeRun mode of transmitter:

- If the FreeRun mode is disabled, transmission will be terminated and an interrupt will be generated when the sent bytes reach the specified length. Before next transmission, you need to re-disable and enable the TxE bit.
- If the FreeRun mode is enabled, the transmitter will send when there is data in the TX FIFO, and will not stop working because the sent bytes reach the specified length.

10.3.5 Receiver

The receiver contains a 128-byte receive FIFO to store the received data. Software can check the status of receiver by querying the available data count value of RX FIFO through rx_fifo_cnt in the register uart_fifo_config_1. The low 8 bits of the register URX_RTO_TIMER are used to set a receiving timeout threshold, which will trigger an interrupt when the receiver fails to receive data beyond the threshold. The cr_urx_deg_en and cr_urx_deg_cnt in the register urx_config are used to enable the deburring function and set the threshold, which control the filtering part before sampling by UART. UART will filter out the burrs whose width is lower than the threshold in the waveform and then

send it to sampling.

10.3.6 Automatic baud rate detection

The UART module supports automatic baud rate detection, which is divided into two modes, a generic mode and a fixed character (square wave) mode. The cr_urx_abr_en in the urx_config register enables auto baud rate detection, and when it is turned on, both detection modes are enabled.

General mode

For any character data received, the UART module will count the number of clocks in the bit width. This number will then be written into the lower 16 bits of the register STS_URX_ABR_PRD and used to calculate the baud rate. Therefore, when the first received data bit is 1, the correct baud rate can be obtained. Such as ‘0x01’ under LSB-FIRST.

Fixed character mode

In this mode, after counting the number of clocks in the starting bit width, the UART module will continue to count the number of clocks of subsequent data bits and compare with the start bit. If it fluctuates within the allowable error range, it will pass the test, otherwise the count value will be discarded. Therefore, only when the fixed characters ‘0x55’ / ‘0xD5’ under LSB-FIRST or ‘0xAA’ / ‘0xAB’ under MSB-FIRST are received. The UART module will write the count value of the number of clocks in the starting bit width into the upper 16 bits of the register STS_URX_ABR_PRD. As shown below:

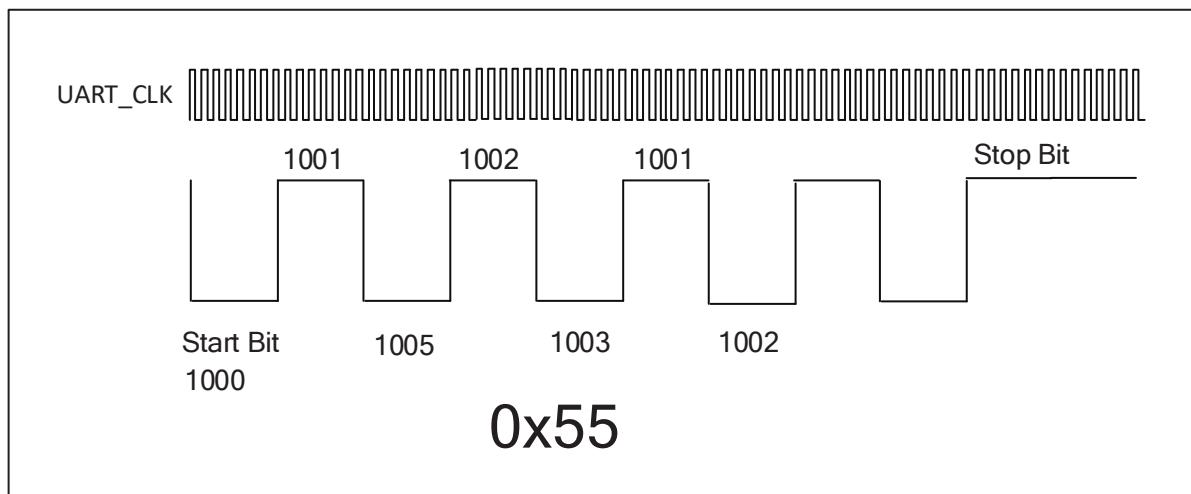


Fig. 10.4: UART fixed character mode waveform

For an unknown baud rate, the UART uses **UART_CLK** to count the start bit width of 1000, and the second bit width of 1001. If there is no more than 4 **UART_CLK** floating up and down from the previous bit width, the UART will continue to count the third bit, which is 1005. If the difference with the start bit exceeds 4, the detection fails and the data is discarded. The UART will sequentially compare the first 6 bits of the data bit with the start bit.

The formula for calculating the detected baud rate is as follows:

$$\text{Baudrate} = \frac{\text{UART_clk}}{\text{Count} + 1}$$

10.3.7 Hardware flow control

The UART supports hardware flow control in CTS / RTS mode to prevent data in the FIFO from being lost because it is too late to process. The hardware flow control connection is shown in the following figure:

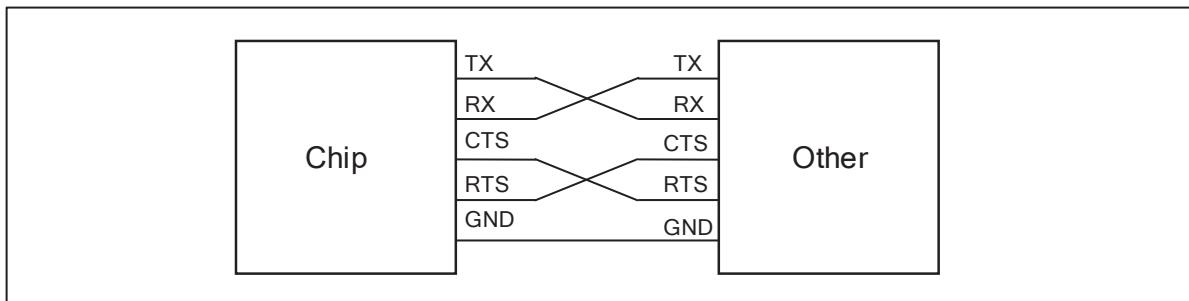


Fig. 10.5: UART flow control

When using the hardware flow control function, the output signal RTS is low to indicate that the other party is requested to send data, and RTS is high to indicate that the other party is notified to suspend data transmission until RTS is restored to low level. Two ways for hardware flow control of the transmitter:

- Hardware control (the `cr_urx_rts_sw_mode` in the register `uart_sw_mode` is 0): RTS goes high when `cr_urx_en` in the register `urx_config` is not turned on or the RX FIFO is almost full (one byte left).
- Software control (the `cr_urx_rts_sw_mode` in the register `uart_sw_mode` is 1): The level of RTS can be changed by configuring `cr_urx_rts_sw_val` in the register `uart_sw_mode`.

The TX CTS can be enabled by configuring the bit `<cr_utx_cts_en>` of the `utx_config` register. When the device detects that the input signal CTS is pulled high, TX will stop sending data until it detects that CTS is pulled low before continuing to send.

10.3.8 LIN transmission mode

When the transmitter needs to use the LIN transmission mode, it can send the BREAK field and the SYNC field by configuring `<cr_utx_lin_en>`. The width of the interval field is determined by `<cr_utx_bit_cnt_b>`.

When the receiver needs to use the LIN transmission mode, you can configure `<cr_urx_lin_en>` to detect the interval field and the synchronization field, and trigger the corresponding interrupt `<urx_lse_int>` when the format of the synchronization field is wrong.

10.3.9 DMA transfer mode

UART supports DMA transfer. Using DMA transfer, the TX and RX FIFO thresholds need to be set respectively by tx_fifo_th and rx_fifo_th in register uart_fifo_config_1. When this mode is enabled, if tx_fifo_cnt in uart_fifo_config_1 is greater than tx_fifo_th, a DMA TX request will be triggered. After the DMA is configured, when the DMA receives the request, it will move the data from the memory to the TX FIFO according to the settings. If the rx_fifo_cnt in uart_fifo_config_1 is greater than rx_fifo_th, the DMA RX request will be triggered. After the DMA is configured, when the DMA receives the request, it will transfer the data of the RX FIFO to the memory according to the settings.

10.3.10 UART interrupt

UART has a wealth of interrupt control, including the following interrupt modes:

- TX transmission end interrupt
- RX transmission end interrupt
- TX FIFO request interrupt
- RX FIFO request interrupt
- RX timeout interrupt
- RX parity error interrupt
- TX FIFO overflow interrupt
- RX FIFO overflow interrupt
- RX LIN mode synchronization field (SYNC Field) error interrupt

TX and RX can respectively set a transmission length value through the upper 16 bits of the UTX_CONFIG and URX_CONFIG registers. When the number of bytes transmitted reaches this value, the corresponding TX/RX transmission end interrupt will be triggered.

The TX/RX FIFO request interrupt will be triggered when the available FIFO count value is greater than the threshold set in the register UART_FIFO_CONFIG_1, and the interrupt flag will be automatically cleared when the condition is not met.

The RX timeout interrupt will be triggered when the receiver exceeds the timeout threshold without receiving data, and the RX parity error interrupt will occur when a parity error occurs. If the TX/RX FIFO overflows or underflows, the corresponding overflow interrupt will be triggered.

When the FIFO clear bit TFICLR/RFICLR is set to 1, the corresponding FIFO will be cleared and the overflow interrupt flag will be automatically cleared. When the LIN mode is enabled, the synchronization field (SYNC Field) should be 0x55 according to the protocol. Therefore, when the received data is not 0x55, the synchronization field error interrupt is triggered.

The interrupt status can be inquired through the register uart_int_sts, and the interrupt can be cleared by writing 1 to

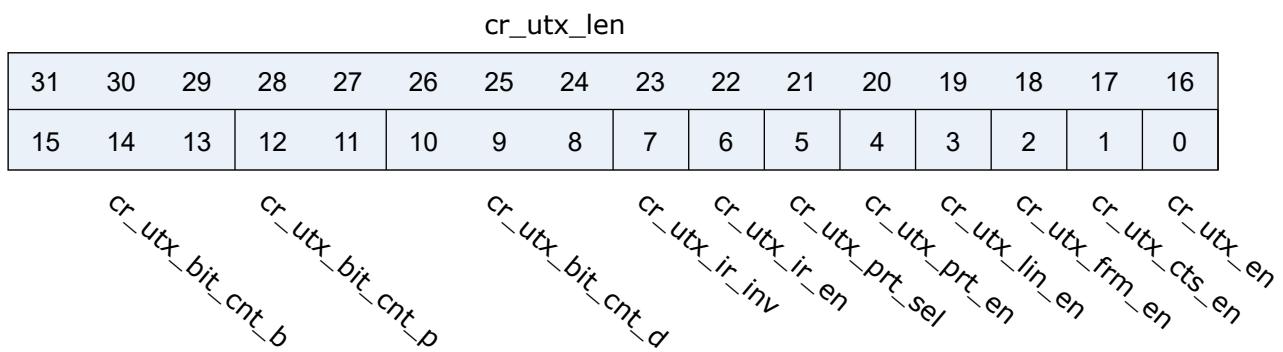
the corresponding bit of the register `uart_int_clear`.

10.4 Register description

Name	Description
<code>utx_config</code>	UART TX configuration register
<code>urx_config</code>	UART RX configuration register
<code>uart_bit_prd</code>	UART period control register
<code>data_config</code>	UART data configuration register
<code>utx_ir_position</code>	UART TX ir position control register
<code>urx_ir_position</code>	UART RX ir position control register
<code>urx_rto_timer</code>	RTO interrupt control register
<code>uart_sw_mode</code>	UART SW mode configuration register
<code>uart_int_sts</code>	UART interrupt status
<code>uart_int_mask</code>	UART interrupt mask
<code>uart_int_clear</code>	UART interrupt clear
<code>uart_int_en</code>	UART interrupt enable
<code>uart_status</code>	UART status control register
<code>sts_urx_abr_prd</code>	Auto baud detection control register
<code>uart_fifo_config_0</code>	UART FIFO configuration register0
<code>uart_fifo_config_1</code>	UART FIFO configuration register1
<code>uart_fifo_wdata</code>	UART FIFO write data
<code>uart_fifo_rdata</code>	UART FIFO read data

10.4.1 utx_config

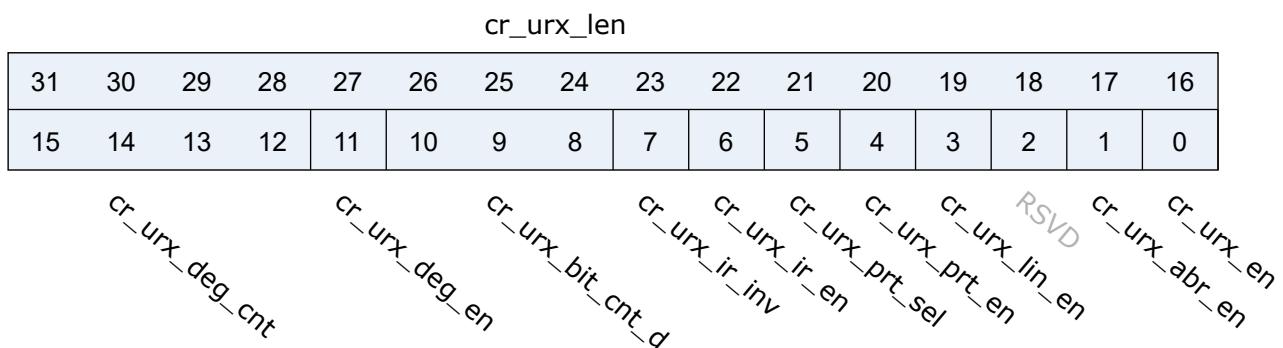
Address: 0x4000a000



Bits	Name	Type	Reset	Description
31:16	cr_utx_len	r/w	16'd0	Length of UART TX data transfer (Unit: character/byte) (Don't-care if cr_utx_frm_en is enabled)
15:13	cr_utx_bit_cnt_b	r/w	3'd4	UART TX BREAK bit count (for LIN protocol) Note: Additional 8 bit times will be added since LIN Break field requires at least 13 bit times
12:11	cr_utx_bit_cnt_p	r/w	2'd1	UART TX STOP bit count (unit: 0.5 bit)
10:8	cr_utx_bit_cnt_d	r/w	3'd7	UART TX DATA bit count for each character
7	cr_utx_ir_inv	r/w	1'b0	Inverse signal of UART TX output in IR mode
6	cr_utx_ir_en	r/w	1'b0	Enable signal of UART TX IR mode
5	cr_utx_prt_sel	r/w	1'b0	Select signal of UART TX parity bit 1: Odd parity 0: Even parity
4	cr_utx_prt_en	r/w	1'b0	Enable signal of UART TX parity bit
3	cr_utx_lin_en	r/w	1'b0	Enable signal of UART TX LIN mode (LIN header will be sent before sending data)
2	cr_utx_frm_en	r/w	1'b0	Enable signal of UART TX freerun mode (utx_end_int will be disabled)
1	cr_utx_cts_en	r/w	1'b0	Enable signal of UART TX CTS flow control function
0	cr_utx_en	r/w	1'b0	Enable signal of UART TX function Asserting this bit will trigger the transaction, and should be de-asserted after finish

10.4.2 urx_config

Address: 0x4000a004



Bits	Name	Type	Reset	Description
31:16	cr_urx_len	r/w	16'd0	Length of UART RX data transfer (Unit: character/byte) urx_end_int will assert when this length is reached
15:12	cr_urx_deg_cnt	r/w	4'd0	De-glitch function cycle count
11	cr_urx_deg_en	r/w	1'b0	Enable signal of RXD input de-glitch function
10:8	cr_urx_bit_cnt_d	r/w	3'd7	UART RX DATA bit count for each character
7	cr_urx_ir_inv	r/w	1'b0	Inverse signal of UART RX input in IR mode
6	cr_urx_ir_en	r/w	1'b0	Enable signal of UART RX IR mode
5	cr_urx_prt_sel	r/w	1'b0	Select signal of UART RX parity bit 1: Odd parity 0: Even parity
4	cr_urx_prt_en	r/w	1'b0	Enable signal of UART RX parity bit
3	cr_urx_lin_en	r/w	1'b0	Enable signal of UART RX LIN mode (LIN header will be required and checked before receiving data)
2	RSVD			
1	cr_urx_abr_en	r/w	1'b0	Enable signal of UART RX Auto Baud Rate detection function
0	cr_urx_en	r/w	1'b0	Enable signal of UART RX function

10.4.3 uart_bit_prd

Address: 0x4000a008

cr_urx_bit_prd

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

cr_utx_bit_prd

Bits	Name	Type	Reset	Description
31:16	cr_urx_bit_prd	r/w	16'd255	Period of each UART RX bit, related to baud rate
15:0	cr_utx_bit_prd	r/w	16'd255	Period of each UART TX bit, related to baud rate

10.4.4 data_config

Address: 0x4000a00c

RSVD															
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

cr_uart_bit_inv

Bits	Name	Type	Reset	Description
31:1	RSVD			
0	cr_uart_bit_inv	r/w	1'b0	Bit-inverse signal for each data byte 0: Each byte is sent out LSB-first 1: Each byte is sent out MSB-first

10.4.5 utx_ir_position

Address: 0x4000a010

cr_utx_ir_pos_p

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

cr_utx_ir_pos_s

Bits	Name	Type	Reset	Description
31:16	cr_utx_ir_pos_p	r/w	16'd159	STOP position of UART TX IR pulse
15:0	cr_utx_ir_pos_s	r/w	16'd112	START position of UART TX IR pulse

10.4.6 urx_ir_position

Address: 0x4000a014

RSVD																
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	

cr_urx_ir_pos_s

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	cr_urx_ir_pos_s	r/w	16'd111	START position of UART RXD pulse recovered from IR signal

10.4.7 urx_rto_timer

Address: 0x4000a018

RSVD																
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	

cr_urx_rto_value

Bits	Name	Type	Reset	Description
31:8	RSVD			
7:0	cr_urx_rto_value	r/w	8'd15	Time-out value for triggering RTO interrupt (unit: bit time)

10.4.8 uart_sw_mode

Address: 0x4000a01c

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

RSVD RSVD

cr_urx_rts_sw_val cr_urx_tx_sw_mode cr_utx_txd_sw_mode cr_utx_txd_sw_val

Bits	Name	Type	Reset	Description
31:4	RSVD			
3	cr_urx_rts_sw_val	r/w	1'b0	UART RX RTS output SW control value
2	cr_urx_rts_sw_mode	r/w	1'b0	UART RX RTS output SW control mode
1	cr_utx_txd_sw_val	r/w	1'b0	UART TX TXD output SW control value
0	cr_utx_txd_sw_mode	r/w	1'b0	UART TX TXD output SW control mode

10.4.9 uart_int_sts

Address: 0x4000a020

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

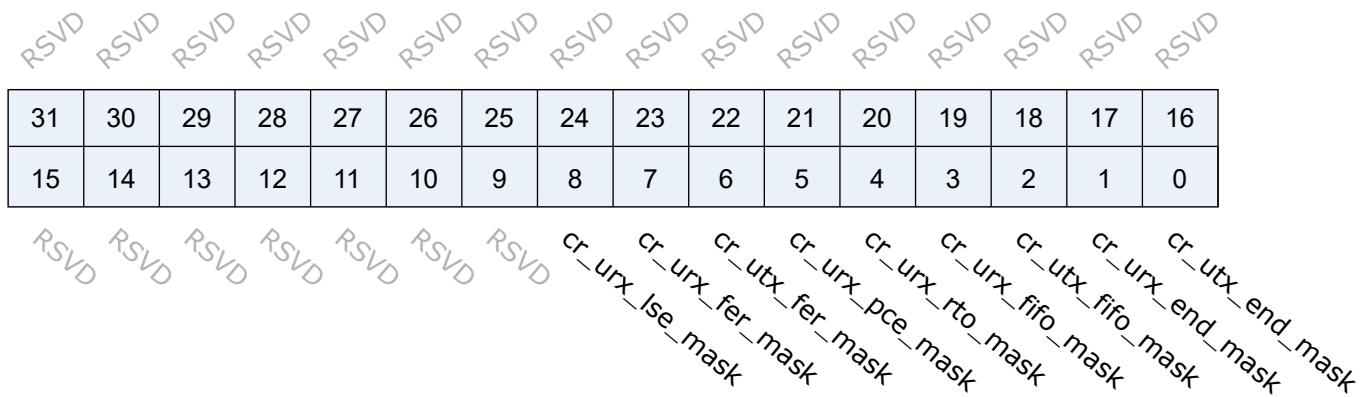
RSVD RSVD

urx_lse_int urx_fer_int utx_fer_int urx_pce_int urx_rto_int urx_fifo_int urx_end_int utx_end_int

Bits	Name	Type	Reset	Description
31:9	RSVD			
8	urx_lse_int	r	1'b0	UART RX LIN mode sync field error interrupt
7	urx_fer_int	r	1'b0	UART RX FIFO error interrupt, auto-cleared when FIFO overflow/underflow error flag is cleared
6	utx_fer_int	r	1'b0	UART TX FIFO error interrupt, auto-cleared when FIFO overflow/underflow error flag is cleared
5	urx_pce_int	r	1'b0	UART RX parity check error interrupt
4	urx_rto_int	r	1'b0	UART RX Time-out interrupt
3	urx_fifo_int	r	1'b0	UART RX FIFO ready ($rx_fifo_cnt > rx_fifo_th$) interrupt, auto-cleared when data is popped
2	utx_fifo_int	r	1'b0	UART TX FIFO ready ($tx_fifo_cnt > tx_fifo_th$) interrupt, auto-cleared when data is pushed
1	urx_end_int	r	1'b0	UART RX transfer end interrupt (set according to cr_urx_len)
0	utx_end_int	r	1'b0	UART TX transfer end interrupt (set according to cr_utx_len)

10.4.10 uart_int_mask

Address: 0x4000a024

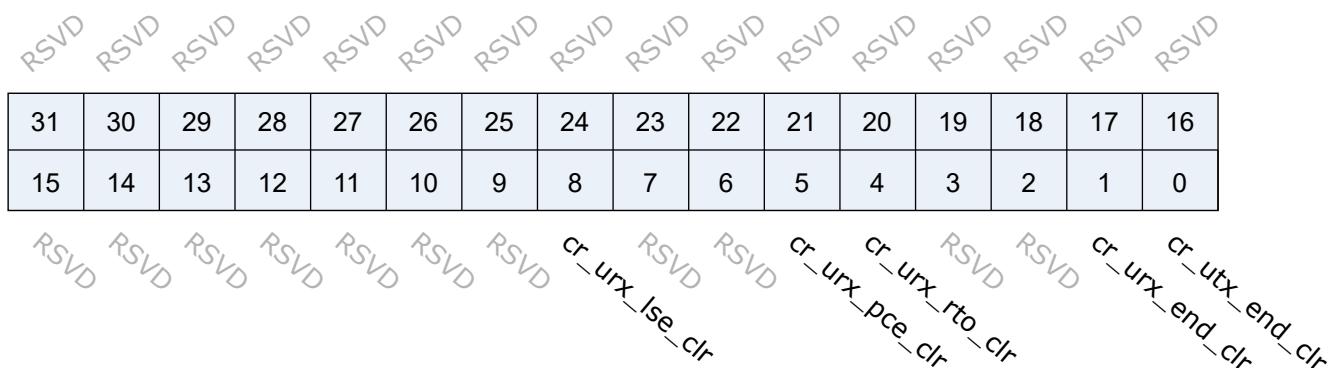


Bits	Name	Type	Reset	Description
31:9	RSVD			
8	cr_urx_lse_mask	r/w	1'b1	Interrupt mask of urx_lse_int
7	cr_urx_fer_mask	r/w	1'b1	Interrupt mask of urx_fer_int
6	cr_utx_fer_mask	r/w	1'b1	Interrupt mask of utx_fer_int

Bits	Name	Type	Reset	Description
5	cr_urx_pce_mask	r/w	1'b1	Interrupt mask of urx_pce_int
4	cr_urx_rto_mask	r/w	1'b1	Interrupt mask of urx_rto_int
3	cr_urx_fifo_mask	r/w	1'b1	Interrupt mask of urx_fifo_int
2	cr_utx_fifo_mask	r/w	1'b1	Interrupt mask of utx_fifo_int
1	cr_urx_end_mask	r/w	1'b1	Interrupt mask of urx_end_int
0	cr_utx_end_mask	r/w	1'b1	Interrupt mask of utx_end_int

10.4.11 uart_int_clear

Address: 0x4000a028



Bits	Name	Type	Reset	Description
31:9	RSVD			
8	cr_urx_lse_clr	w1c	1'b0	Interrupt clear of urx_lse_int
7:6	RSVD			
5	cr_urx_pce_clr	w1c	1'b0	Interrupt clear of urx_pce_int
4	cr_urx_rto_clr	w1c	1'b0	Interrupt clear of urx_rto_int
3:2	RSVD			
1	cr_urx_end_clr	w1c	1'b0	Interrupt clear of urx_end_int
0	cr_utx_end_clr	w1c	1'b0	Interrupt clear of utx_end_int

10.4.12 uart_int_en

Address: 0x4000a02c

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | |

Labels for columns 13-18: cr_urx_lse_en, cr_urx_fer_en, cr_utx_fer_en, cr_urx_pce_en, cr_urx_rto_en, cr_urx_fifo_en, cr_utx_fifo_en, cr_utx_end_en

Bits	Name	Type	Reset	Description
31:9	RSVD			
8	cr_urx_lse_en	r/w	1'b1	Interrupt enable of urx_lse_int
7	cr_urx_fer_en	r/w	1'b1	Interrupt enable of urx_fer_int
6	cr_utx_fer_en	r/w	1'b1	Interrupt enable of utx_fer_int
5	cr_urx_pce_en	r/w	1'b1	Interrupt enable of urx_pce_int
4	cr_urx_rto_en	r/w	1'b1	Interrupt enable of urx_rto_int
3	cr_urx_fifo_en	r/w	1'b1	Interrupt enable of urx_fifo_int
2	cr_utx_fifo_en	r/w	1'b1	Interrupt enable of utx_fifo_int
1	cr_urx_end_en	r/w	1'b1	Interrupt enable of urx_end_int
0	cr_utx_end_en	r/w	1'b1	Interrupt enable of utx_end_int

10.4.13 uart_status

Address: 0x4000a030

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | |

Labels for columns 13-18: sts_utx_bus_busy, sts_urx_bus_busy

Bits	Name	Type	Reset	Description
31:2	RSVD			
1	sts_urx_bus_busy	r	1'b0	Indicator of UART RX bus busy
0	sts_utx_bus_busy	r	1'b0	Indicator of UART TX bus busy

10.4.14 sts_urx_abr_prd

Address: 0x4000a034

sts_urx_abr_prd_0x55

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

sts_urx_abr_prd_start

Bits	Name	Type	Reset	Description
31:16	sts_urx_abr_prd_0x55	r	16'd0	Bit period of Auto Baud Rate detection using codeword 0x55
15:0	sts_urx_abr_prd_start	r	16'd0	Bit period of Auto Baud Rate detection using START bit

10.4.15 uart_fifo_config_0

Address: 0x4000a080

Bits	Name	Type	Reset	Description
31:8	RSVD			
7	rx_fifo_underflow	r	1'b0	Underflow flag of RX FIFO, can be cleared by rx_fifo_clr

Bits	Name	Type	Reset	Description
6	rx_fifo_overflow	r	1'b0	Overflow flag of RX FIFO, can be cleared by rx_fifo_clr
5	tx_fifo_underflow	r	1'b0	Underflow flag of TX FIFO, can be cleared by tx_fifo_clr
4	tx_fifo_overflow	r	1'b0	Overflow flag of TX FIFO, can be cleared by tx_fifo_clr
3	rx_fifo_clr	w1c	1'b0	Clear signal of RX FIFO
2	tx_fifo_clr	w1c	1'b0	Clear signal of TX FIFO
1	uart_dma_rx_en	r/w	1'b0	Enable signal of dma_rx_req/ack interface
0	uart_dma_tx_en	r/w	1'b0	Enable signal of dma_tx_req/ack interface

10.4.16 uart_fifo_config_1

Address: 0x4000a084

rx_fifo_th								tx_fifo_th							
RSVD								RSVD							
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							
rx_fifo_cnt								tx_fifo_cnt							

Bits	Name	Type	Reset	Description
31	RSVD			
30:24	rx_fifo_th	r/w	7'd0	RX FIFO threshold, dma_rx_req will not be asserted if tx_fifo_cnt is less than this value
23	RSVD			
22:16	tx_fifo_th	r/w	7'd0	TX FIFO threshold, dma_tx_req will not be asserted if tx_fifo_cnt is less than this value
15:8	rx_fifo_cnt	r	8'd0	RX FIFO available count
7:0	tx_fifo_cnt	r	8'd128	TX FIFO available count

10.4.17 uart_fifo_wdata

Address: 0x4000a088

RSVD															
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

uart_fifo_wdata

Bits	Name	Type	Reset	Description
31:8	RSVD			
7:0	uart_fifo_wdata	w	x	

10.4.18 uart_fifo_rdata

Address: 0x4000a08c

RSVD															
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

uart_fifo_rdata

Bits	Name	Type	Reset	Description
31:8	RSVD			
7:0	uart_fifo_rdata	r	8'h0	

11

I2C

11.1 I2C introduction

I2C (Inter-Integrated Circuit) is a serial communication bus that uses a multi-master-slave architecture to connect low-speed peripheral devices.

Each device has a unique address identification and can be used as a transmitter or receiver. Each device connected to the bus can set the address by software with a unique address and the always-receiving master-slave relationship. The host can be used as a host transmitter or a host receiver.

If two or more hosts are initialized at the same time, data transmission can prevent data from being destroyed through collision detection and arbitration.

BL702 includes an I2C controller host, which can be flexibly configured with slaveAddr, subAddr, and data transmission to facilitate communication with slave devices. It provides 2 word depth fifo and provides interrupt functions. It can be used with DMA to improve efficiency and flexibly adjust clock frequency.

11.2 I2C main features

- Support host mode
- Support multi-master mode and arbitration function
- Flexible clock frequency adjustment

11.3 I2C function description

Table 11.1: Pin lists

Name	Type	Description
I2Cx_SCL	input/output	I2C serial clock signal
I2Cx_SDA	input/output	I2C serial data signal

11.3.1 Start and stop conditions

All transfers begin with a START condition and end with a STOP condition.

The start and stop conditions are generally generated by the master. The bus is considered to be in a busy state after the start condition, and is considered to be in an idle state for a period of time after the stop condition.

Start condition: SDA generates a high-to-low level transition when SCL is high;

Stop condition: SDA generates a low-to-high level transition when SCL is high.

The waveform diagram is as follows:

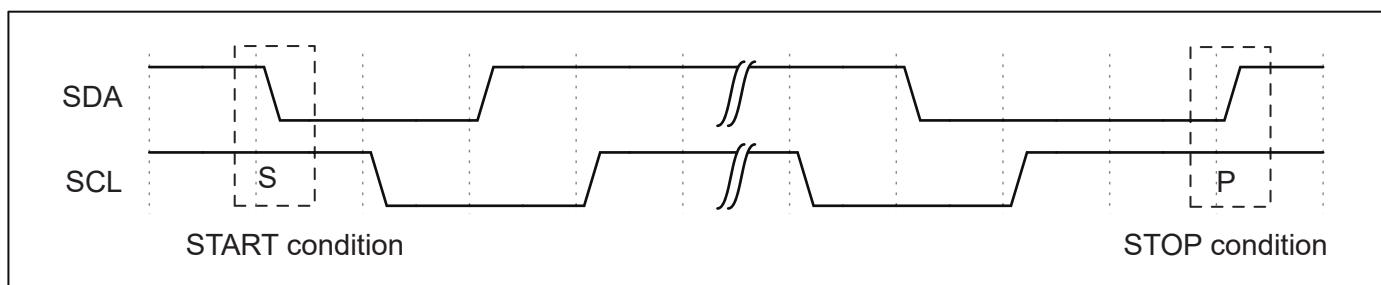


Fig. 11.1: I2C stop/start condition

11.3.2 Data transmission format

The first 8 bits transmitted are the address byte, including the 7-bit slave address and the 1-bit direction bit. Data sent or received by the host is controlled by the eighth bit of the first byte sent by the host.

If it is 0, it means that the data is sent by the master; if it is 1, it means that the data is received by the master, and then the slave sends an acknowledge bit (ACK). After the data transmission is completed, the master sends a stop signal. The waveform is as follows:

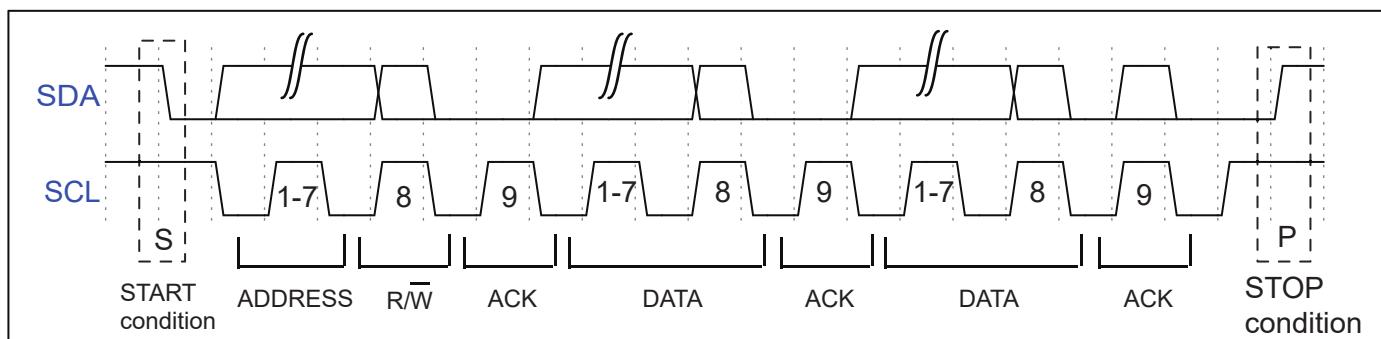


Fig. 11.2: I2C data transmission format

Timing of master transmission and slave reception

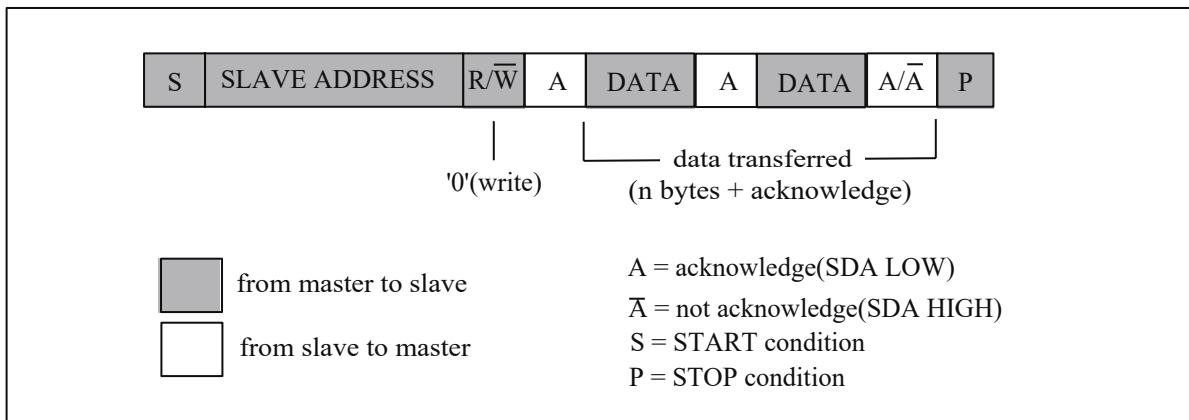


Fig. 11.3: Master tx and slave rx

Timing of master receive and slave send

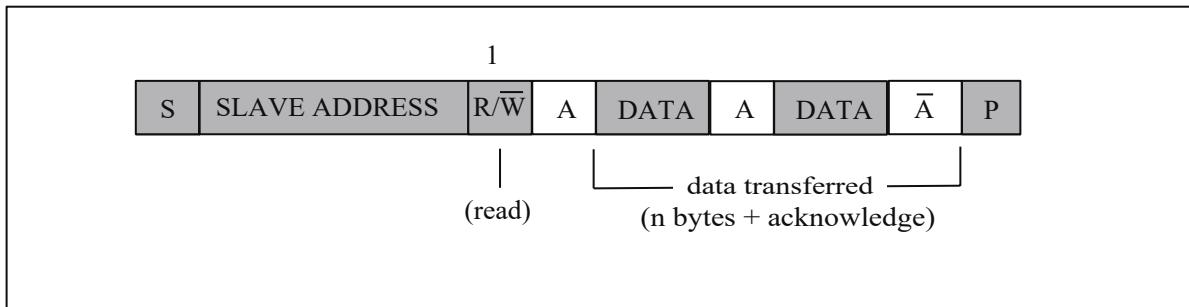


Fig. 11.4: Master rx and slave tx

11.3.3 Arbitration

When there are multiple masters on the I²C bus, multiple masters may start transmitting at the same time. At this time, it is necessary to rely on the arbitration mechanism to determine which master has the right to complete the next data transfer. The remaining masters must give up control of the bus. The transmission cannot be started again until the bus is free.

During the transmission process, all hosts need to check whether SDA is consistent with the data they want to send when SCL is high. When the SDA level is different from expected, it means that other hosts are also transmitting at the same time. Hosts with different SDA levels will lose the arbitration and other hosts will complete the data transmission.

The waveform diagram of two hosts transmitting data and starting the arbitration mechanism at the same time is as follows:

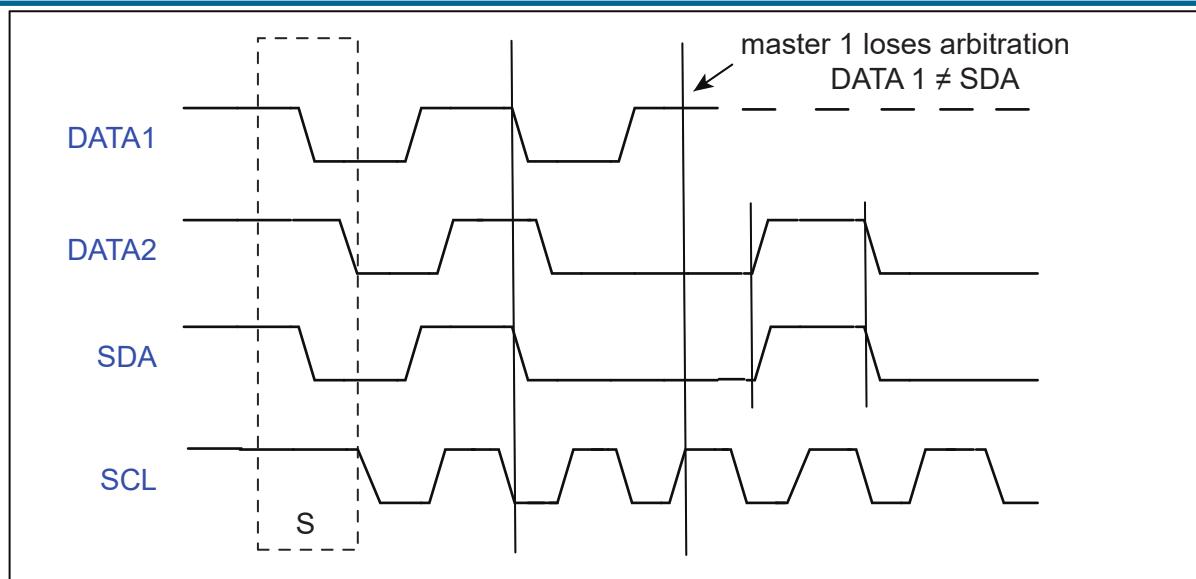


Fig. 11.5: Tx and Rx together

11.4 I2C clock setting

The I2C clock is derived from bclk (bus clock), which can be divided based on the bclk clock.

Register i2c_prd_data can divide the clock of the data segment. The i2c module divides the data transmission into 4 phases. Each phase is controlled by a single byte in the register. The number of samples in each phase can be set. The 4 samples together determine the frequency division coefficient of the i2c clock.

For example, bclk is 32M and the value of register i2c_prd_data is 0x0f0f0f0f by default without configuration. Then the clock frequency of I2C is $32M / ((15 + 1) * 4) = 500K$.

Similarly, the registers i2c_prd_start and i2c_prd_stop also divide the clock of the start bit and stop bit respectively.

11.5 I2C configuration process

11.5.1 Configuration item

- Read and write flags
- Slave address
- Slave device address
- Slave device address length
- Data (when sending, configure the data to be sent; when receiving, store the received data)
- Data length
- Enable signal

11.5.2 Read and write flags

I2C supports two working states: sending and receiving. Register cr_i2c_pkt_dir indicates the sending or receiving status. When it is set to 0, it indicates the sending state, and when it is set to 1, it indicates the receiving state.

11.5.3 Slave address

Each slave device connected to I2C will have a unique address. Usually the address length is 7 bits. The slave device address will be written into the register cr_i2c_slv_addr. I2C will automatically shift left by 1 bit before sending it from the device address. Transmit/receive direction bit on the low-order complement.

11.5.4 Slave register address

Slave device register address indicates the register address that I2C needs to read and write to a certain register of the slave device. The slave device address will be written to the register i2c_sub_addr, and the register cr_i2c_sub_addr_en needs to be set.

If the register cr_i2c_sub_addr_en is set to 0, the I2C master will skip the slave register address segment when transmitting.

11.5.5 Slave device address length

The slave device address length is decremented by one and written to the register cr_i2c_sub_addr_bc.

11.5.6 Data

The data part represents the data that needs to be sent to the slave device, or the data that needs to be received from the slave device.

When I2C sends data, the data needs to be written into the I2C FIFO in word units in turn, and the data is written to the register address i2c_fifo_wdata of the FIFO.

When the I2C receives data, it needs to read the data from the I2C FIFO in units of words in order, and the received data reads the register address i2c_fifo_rdata of the FIFO.

11.5.7 Data length

Decrement the data length by one and write to the register cr_i2c_pkt_len.

11.5.8 Enable signal

After the above configurations are completed, write the enable signal register cr_i2c_m_en to 1 to automatically start the I2C transmission process.

When the read-write flag is set to 0, I2C sends data, and the host sends the process:

1. Start bit
2. (1 bit left from device address + 0) + ACK
3. Slave device address + ACK
4. 1 byte data + ACK
5. 1 byte data + ACK
6. Stop bit

When the read / write flag is set to 1, I2C receives data and the host sends the process:

1. Start bit
2. (1 bit left from device address + 0) + ACK
3. Slave device address + ACK
4. Start bit
5. (1 bit left from device address + 1) + ACK
6. 1 byte data + ACK
7. 1 byte data + ACK
8. Stop bit

11.6 FIFO management

I2C FIFO has a 2-word depth, and I2C includes RX FIFO and TX FIFO. The rx_fifo_cnt in the register i2c_fifo_config_1 represents how much data (in word) in RX FIFO needs to be read. The tx_fifo_cnt in the register i2c_fifo_config_1 represents how much free space (in word) in TX FIFO for writing.

I2C FIFO status:

- RX FIFO underflow: When the data in RX FIFO is completely read out or empty, if I2C continues to read data from RX FIFO, the rx_fifo_underflow in the register i2c_fifo_config_0 will be set to 1;
- RX FIFO overflow: When I2C receives data until the two words of RX FIFO are filled, without reading RX FIFO, if I2C receives data again, the rx_fifo_overflow in the register i2c_fifo_config_0 will be set to 1;
- TX FIFO underflow: When the data size filled into TX FIFO does not meet the configured I2C data length: cr_i2c_-

pkt_len in i2c_config, and no new data is filled into TX FIFO, the tx_fifo_underflow in the register i2c_fifo_config_0 will be set to 1;

- TX FIFO overflow: After the two words of TX FIFO are filled, before the data in TX FIFO is sent out, if data is filled into TX FIFO again, the tx_fifo_overflow in the register i2c_fifo_config_0 will be set to 1.

11.7 Using DMA

I2C can send and receive data through DMA. Setting i2c_dma_tx_en in the register i2c_fifo_config_0 to 1 will enable the DMA TX mode. After the channel for I2C is allocated, DMA will transfer data from memory to the i2c_fifo_wdata register. Setting i2c_dma_rx_en in the register i2c_fifo_config_0 to 1 will enable the DMA RX mode. After the channel for I2C is allocated, DMA will transfer the data in the register i2c_fifo_rdata to memory. When I2C is used with DMA, DMA will automatically transfer data, so it is unnecessary for CPU to write data into I2C TX FIFO or read data from I2C RX FIFO.

11.7.1 DMA transmission process

- Configure the read and write flag i2c_config[cr_i2c_pkt_dir] to 0
- Configure the slave device address i2c_config[cr_i2c_slv_addr]
- Configure the slave device register address i2c_sub_addr, the slave device register address length i2c_config[cr_i2c_sub_addr_bc], and configure the slave device register address enable bit i2c_config[cr_i2c_sub_addr_en] to 1
- Set the length of the sent data i2c_config[cr_i2c_pkt_len]
- Enable DMA mode transmission, set i2c_fifo_config_0[i2c_dma_tx_en] to 1
- Configure the data length of DMA transmission DMA_CxControl[TransferSize] (x=0~7)
- Configure DMA source address DMA_CxSrcAddr, data width DMA_CxControl[SWidth], burst size DMA_CxControl[SBSIZE], set DMA_CxControl[SI] to 1 to enable automatic address accumulation mode
- Configure the DMA destination address DMA_CxDstAddr to i2c_fifo_wdata, data width DMA_CxControl[DWidth] to 32 and burst size DMA_CxControl[DBSSize], clear DMA_CxControl[DI] to disable the address automatic accumulation mode
- Enable DMA
- Configure i2c_config[cr_i2c_m_en] to 1 to enable I2C

11.7.2 DMA receiving process

1. Configure the read-write flag i2c_config[cr_i2c_pkt_dir] to 1
2. Configure the slave device address i2c_config[cr_i2c_slv_addr]
3. Configure slave device register address i2c_sub_addr, slave device register address length i2c_config[cr_i2c_sub_addr_bc], configure slave device register address enable bit i2c_config[cr_i2c_sub_addr_en] to 1
4. Set the length of the received data i2c_config[cr_i2c_pkt_len]
5. Enable DMA mode reception, set i2c_fifo_config_0[i2c_dma_rx_en] to 1
7. Configure the data length of DMA transmission DMA_CxControl[TS] (x=0~7)
8. Configure DMA source address DMA_CxSrcAddr to i2c_fifo_rdata, data width DMA_CxControl[SWidth] to 32 and burst size DMA_CxControl[SBSIZE], clear DMA_CxControl[SI] to disable address automatic accumulation mode
7. Configure DMA destination address DMA_CxDstAddr, data width DMA_CxControl[DWidth], burst size DMA_CxControl[DBSSize], set DMA_CxControl[DI] to 1 to enable automatic address accumulation mode
9. Enable DMA
10. Configure i2c_config[cr_i2c_m_en] to 1 to enable I2C

11.8 I2C interrupt

I2C includes the following interrupts:

- I2C_TRANS_END_INT: I2C transfer end interrupt
- I2C_TX_FIFO_READY_INT: Interrupt is triggered when I2C TX FIFO has free space available for filling
- I2C_RX_FIFO_READY_INT: When I2C RX FIFO receives data, trigger interrupt
- I2C_NACK_RECV_INT: When the I2C module detects a NACK state, an interrupt is triggered
- I2C_ARB_LOST_INT: I2C arbitration lost interrupt
- I2C_FIFO_ERR_INT: I2C FIFO ERROR interrupt

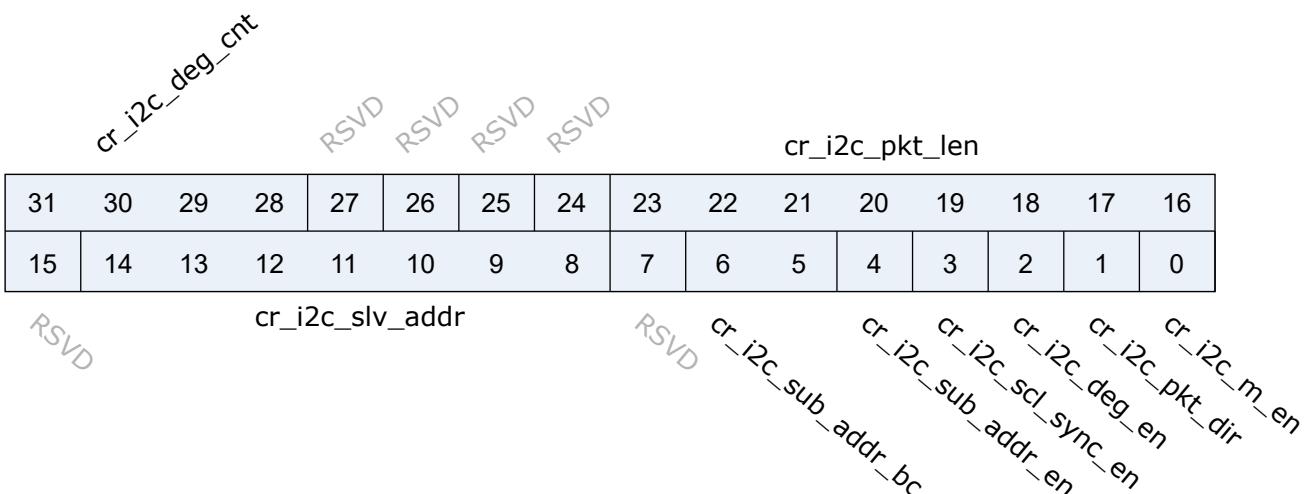
11.9 Register description

Name	Description
i2c_config	I2C configuration register
i2c_int_sts	I2C interrupt status
i2c_sub_addr	I2C sub-address configuration

Name	Description
i2c_bus_busy	I2C bus busy control register
i2c_prd_start	I2C length of start phase
i2c_prd_stop	I2C length of stop phase
i2c_prd_data	I2C length of data phase
i2c_fifo_config_0	I2C FIFO configuration register0
i2c_fifo_config_1	I2C FIFO configuration register1
i2c_fifo_wdata	I2C FIFO write data
i2c_fifo_rdata	I2C FIFO read data

11.9.1 i2c_config

Address: 0x4000a300



Bits	Name	Type	Reset	Description
31:28	cr_i2c_deg_cnt	r/w	4'd0	De-glitch function cycle count
27:24	RSVD			
23:16	cr_i2c_pkt_len	r/w	8'd0	Packet length (unit: byte)
15	RSVD			
14:8	cr_i2c_slv_addr	r/w	7'd0	Slave address for I2C transaction (target address)
7	RSVD			

Bits	Name	Type	Reset	Description
6:5	cr_i2c_sub_addr_bc	r/w	2'd0	Sub-address field byte count 2'd0: 1-byte, 2'd1: 2-byte, 2'd2: 3-byte, 2'd3: 4-byte
4	cr_i2c_sub_addr_en	r/w	1'b0	Enable signal of I2C sub-address field
3	cr_i2c_scl_sync_en	r/w	1'b1	Enable signal of I2C SCL synchronization, should be enabled to support Multi-Master and Clock-Stretching (Normally should not be turned-off)
2	cr_i2c_deg_en	r/w	1'b0	Enable signal of I2C input de-glitch function (for all input pins)
1	cr_i2c_pkt_dir	r/w	1'b1	Transfer direction of the packet 1'b0: Write; 1'b1: Read
0	cr_i2c_m_en	r/w	1'b0	Enable signal of I2C Master function Asserting this bit will trigger the transaction, and should be de-asserted after finish

11.9.2 i2c_int_sts

Address: 0x4000a304

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

Labels for fields:

- RSVD (Registers Reserved)
- cr_i2c_fer_en
- cr_i2c_arb_en
- cr_i2c_nak_en
- cr_i2c_rxf_en
- cr_i2c_txf_en
- cr_i2c_end_en
- RSVD
- rsvd
- cr_i2c_arb_clr
- cr_i2c_nak_clr
- rsvd
- rsvd
- cr_i2c_end_clr
- RSVD
- RSVD
- cr_i2c_fer_mask
- cr_i2c_arb_mask
- cr_i2c_nak_mask
- cr_i2c_rxf_mask
- cr_i2c_txf_mask
- cr_i2c_end_mask
- RSVD
- RSVD
- i2c_fer_int
- i2c_arb_int
- i2c_nak_int
- i2c_rxf_int
- i2c_txf_int
- i2c_end_int

Bits	Name	Type	Reset	Description
31:30	RSVD			
29	cr_i2c_fer_en	r/w	1'b1	Interrupt enable of i2c_fer_int
28	cr_i2c_arb_en	r/w	1'b1	Interrupt enable of i2c_arb_int
27	cr_i2c_nak_en	r/w	1'b1	Interrupt enable of i2c_nak_int
26	cr_i2c_rxf_en	r/w	1'b1	Interrupt enable of i2c_rxf_int
25	cr_i2c_txf_en	r/w	1'b1	Interrupt enable of i2c_txf_int

Bits	Name	Type	Reset	Description
24	cr_i2c_end_en	r/w	1'b1	Interrupt enable of i2c_end_int
23:22	RSVD			
21	rsvd	rsvd	1'b0	
20	cr_i2c_arb_clr	w1c	1'b0	Interrupt clear of i2c_arb_int
19	cr_i2c_nak_clr	w1c	1'b0	Interrupt clear of i2c_nak_int
18	rsvd	rsvd	1'b0	
17	rsvd	rsvd	1'b0	
16	cr_i2c_end_clr	w1c	1'b0	Interrupt clear of i2c_end_int
15:14	RSVD			
13	cr_i2c_fer_mask	r/w	1'b1	Interrupt mask of i2c_fer_int
12	cr_i2c_arb_mask	r/w	1'b1	Interrupt mask of i2c_arb_int
11	cr_i2c_nak_mask	r/w	1'b1	Interrupt mask of i2c_nak_int
10	cr_i2c_rxf_mask	r/w	1'b1	Interrupt mask of i2c_rxf_int
9	cr_i2c_txf_mask	r/w	1'b1	Interrupt mask of i2c_txf_int
8	cr_i2c_end_mask	r/w	1'b1	Interrupt mask of i2c_end_int
7:6	RSVD			
5	i2c_fer_int	r	1'b0	I2C TX/RX FIFO error interrupt, auto-cleared when FIFO overflow/underflow error flag is cleared
4	i2c_arb_int	r	1'b0	I2C arbitration lost interrupt
3	i2c_nak_int	r	1'b0	I2C NACK-received interrupt
2	i2c_rxf_int	r	1'b0	I2C RX FIFO ready (rx_fifo_cnt > rx_fifo_th) interrupt, auto-cleared when data is popped
1	i2c_txf_int	r	1'b0	I2C TX FIFO ready (tx_fifo_cnt > tx_fifo_th) interrupt, auto-cleared when data is pushed
0	i2c_end_int	r	1'b0	I2C transfer end interrupt

11.9.3 i2c_sub_addr

Address: 0x4000a308

cr_i2c_sub_addr_b3

cr_i2c_sub_addr_b2

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

cr_i2c_sub_addr_b1

cr_i2c_sub_addr_b0

Bits	Name	Type	Reset	Description
31:24	cr_i2c_sub_addr_b3	r/w	8'd0	I2C sub-address field - byte[3]
23:16	cr_i2c_sub_addr_b2	r/w	8'd0	I2C sub-address field - byte[2]
15:8	cr_i2c_sub_addr_b1	r/w	8'd0	I2C sub-address field - byte[1]
7:0	cr_i2c_sub_addr_b0	r/w	8'd0	I2C sub-address field - byte[0] (sub-address starts from this byte)

11.9.4 i2c_bus_busy

Address: 0x4000a30c

RSVD	RSVD	RSVD															
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		
RSVD	sts_i2c_bus_busy	cr_i2c_bus_busy_clr															

Bits	Name	Type	Reset	Description
31:2	RSVD			
1	cr_i2c_bus_busy_clr	w1c	1'b0	Clear signal of bus_busy status, not for normal usage (in case I2C bus hangs)
0	sts_i2c_bus_busy	r	1'b0	Indicator of I2C bus busy

11.9.5 i2c_prd_start

Address: 0x4000a310

cr_i2c_prd_s_ph_3								cr_i2c_prd_s_ph_2							
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
cr_i2c_prd_s_ph_1								cr_i2c_prd_s_ph_0							

Bits	Name	Type	Reset	Description
31:24	cr_i2c_prd_s_ph_3	r/w	8'd15	Length of START condition phase 3
23:16	cr_i2c_prd_s_ph_2	r/w	8'd15	Length of START condition phase 2
15:8	cr_i2c_prd_s_ph_1	r/w	8'd15	Length of START condition phase 1
7:0	cr_i2c_prd_s_ph_0	r/w	8'd15	Length of START condition phase 0

11.9.6 i2c_prd_stop

Address: 0x4000a314

Bits	Name	Type	Reset	Description
31:24	cr_i2c_prd_p_ph_3	r/w	8'd15	Length of STOP condition phase 3
23:16	cr_i2c_prd_p_ph_2	r/w	8'd15	Length of STOP condition phase 2
15:8	cr_i2c_prd_p_ph_1	r/w	8'd15	Length of STOP condition phase 1
7:0	cr_i2c_prd_p_ph_0	r/w	8'd15	Length of STOP condition phase 0

11.9.7 i2c_prd_data

Address: 0x4000a318

Bits	Name	Type	Reset	Description
31:24	cr_i2c_prd_d_ph_3	r/w	8'd15	Length of DATA phase 3
23:16	cr_i2c_prd_d_ph_2	r/w	8'd15	Length of DATA phase 2

Bits	Name	Type	Reset	Description
15:8	cr_i2c_prd_d_ph_1	r/w	8'd15	Length of DATA phase 1 Note: This value should not be set to 8'd0, adjust source clock rate instead if higher I2C clock rate is required
7:0	cr_i2c_prd_d_ph_0	r/w	8'd15	Length of DATA phase 0

11.9.8 i2c_fifo_config_0

Address: 0x4000a380

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	

RSVD RSVD

rx_fifo_overflow rx_fifo_underflow tx_fifo_overflow tx_fifo_underflow tx_fifo_clr i2c_dma_rx_en i2c_dma_tx_en

Bits	Name	Type	Reset	Description
31:8	RSVD			
7	rx_fifo_underflow	r	1'b0	Underflow flag of RX FIFO, can be cleared by rx_fifo_clr
6	rx_fifo_overflow	r	1'b0	Overflow flag of RX FIFO, can be cleared by rx_fifo_clr
5	tx_fifo_underflow	r	1'b0	Underflow flag of TX FIFO, can be cleared by tx_fifo_clr
4	tx_fifo_overflow	r	1'b0	Overflow flag of TX FIFO, can be cleared by tx_fifo_clr
3	rx_fifo_clr	w1c	1'b0	Clear signal of RX FIFO
2	tx_fifo_clr	w1c	1'b0	Clear signal of TX FIFO
1	i2c_dma_rx_en	r/w	1'b0	Enable signal of dma_rx_req/ack interface
0	i2c_dma_tx_en	r/w	1'b0	Enable signal of dma_tx_req/ack interface

11.9.9 i2c_fifo_config_1

Address: 0x4000a384

RSVD	rx_fifo_th	RSVD	tx_fifo_th												
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

RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	rx_fifo_cnt	RSVD	tx_fifo_cnt						
------	------	------	------	------	------	-------------	------	------	------	------	------	------	------	-------------

Bits	Name	Type	Reset	Description
31:25	RSVD			
24	rx_fifo_th	r/w	1'd0	RX FIFO threshold, dma_rx_req will not be asserted if tx_fifo_cnt is less than this value
23:17	RSVD			
16	tx_fifo_th	r/w	1'd0	TX FIFO threshold, dma_tx_req will not be asserted if tx_fifo_cnt is less than this value
15:10	RSVD			
9:8	rx_fifo_cnt	r	2'd0	RX FIFO available count
7:2	RSVD			
1:0	tx_fifo_cnt	r	2'd2	TX FIFO available count

11.9.10 i2c_fifo_wdata

Address: 0x4000a388

i2c_fifo_wdata

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

i2c_fifo_wdata

Bits	Name	Type	Reset	Description
31:0	i2c_fifo_wdata	w	x	

11.9.11 i2c_fifo_rdata

Address: 0x4000a38c

i2c_fifo_rdata

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

i2c_fifo_rdata

Bits	Name	Type	Reset	Description
31:0	i2c_fifo_rdata	r	32'h0	

12

PWM

12.1 PWM introduction

Pulse width modulation (PWM) is an analog control method. The bias of the base of the transistor or the gate of the MOS tube is modulated according to the change of the corresponding load to realize the change of the conduction time of the transistor or the MOS tube. So as to realize the change of the output of the switch stable power supply. This method can keep the output voltage of the power supply constant when the working conditions change, and it is a very effective technology to control the analog circuit with the digital signal of the microprocessor. It is widely used in many fields from measurement and communication to power control and conversion.

12.2 PWM main features

- Support 5-channel PWM signal generation
- Three clock sources can be selected (bus clock <bclk>, crystal oscillator clock <xtal>, slow clock <32k>), with 16-bit clock divider
- Double threshold setting, increase pulse flexibility

12.3 PWM function description

12.3.1 Clock and divider

There are three options for each PWM counter clock source, the sources are as follows:

- A. bclk - Chip bus clock
- B. XTAL - External crystal clock
- C. f32k - System RTC clock

Each counter has its own 16-bit frequency divider. The PWM counter will use the divided clock as the counting cycle unit, and perform one action every time a counting cycle passes .

12.3.2 Pulse generation principle

There is a counter in the PWM. When the counter is in the middle of two settable thresholds, the PWM output is 1, otherwise when the counter is outside the two set thresholds, the PWM output is 0. As shown below:

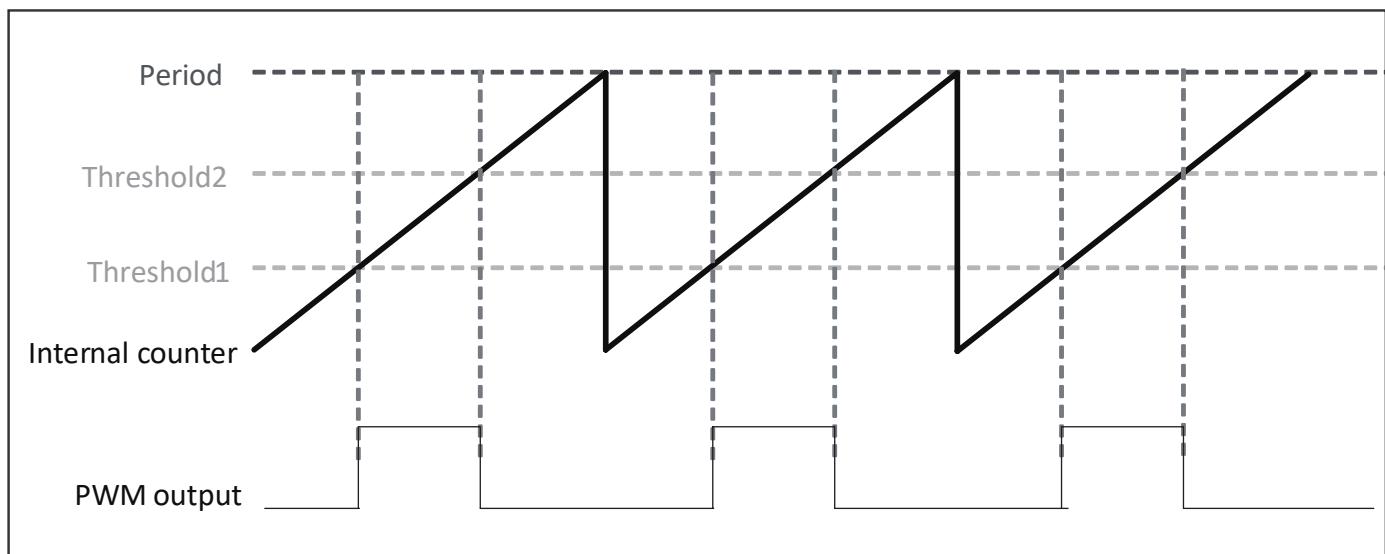


Fig. 12.1: PWM waveform

The PWM cycle is determined by two parts, one is the clock frequency division coefficient, and the other is the clock duration.

The clock division coefficient is set by the register `PWMn_CLK_DIV[15:0]` (n is 0~4), which is used to divide the PWM source clock.

The clock duration is set by the register `PWMn_PERIOD[15:0]` (n is 0~4), which is used to set the number of divided clock cycles for a PWM cycle. That is, the period of PWM=_PWM source clock/`PWMn_CLK_DIV[15:0]`/`PWMn_PERIOD[15:0]`.

The duty cycle of PWM is determined by the clock duration and two thresholds. The first threshold is set by the register `PWMn_THRE1[15:0]` (n is 0~4), the second threshold is set by the register `PWMn_THRE2[15:0]` (n is 0~4), the PWM waveform will be in the first Pull up at one threshold, and pull down at the second threshold. That is, the duty cycle of PWM=(`PWMn_THRE2[15:0]-PWMn_THRE1[15:0]`)/`PWMn_PERIOD[15:0]`.

Example: If the PWM clock source is selected as bclk, which is 72MHz, to generate a 1kHz, 20% duty cycle PWM wave, set as follows:

`PWMn_CLK_DIV[15:0]=2`

`PWMn_PERIOD[15:0]=72000000/2/1000=36000`

`PWMn_THRE1[15:0]=0`

`PWMn_THRE2[15:0]=0+36000*20%=7200`

12.3.3 PWM interrupt

For each PWM channel, you can set the cycle count value. When the cycle number of the PWM output reaches this count value, a PWM interrupt will be generated.

Table 12.1: Duty Cycle Parameters

frequency/MHz	Supported duty cycle (n is an integer, and $2 \leq n \leq 65535^2$)											
36	0%	50%	100%									
24	0%	33.33%	66.67%	100%								
18	0%	25%	50%	75%	100%							
14.4	0%	20%	40%	60%	80%	100%						
12	0%	16.67%	33.33%	50%	66.67%	83.33%	100%					
10.29	0%	14.29%	28.57%	42.86%	57.14%	71.43%	85.71%	100%				
9	0%	12.50%	25%	37.50%	50%	62.50%	75%	87.50%	100%			
8	0%	11.11%	22.22%	33.33%	44.44%	55.56%	66.67%	77.78%	88.89%	100%		
7.2	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%	
.												
.												
.												
72/n	0/n	1/n	2/n	3/n	4/n	5/n	6/n	7/n	8/n	9/n	...	n/n

12.4 Register description

Name	Description
pwm_int_config	PWM interrupt configuration register
pwm0_clkdiv	PWM0 clock division configuration register
pwm0_thre1	PWM0 first counter threshold configuration register
pwm0_thre2	PWM0 second counter threshold configuration register
pwm0_period	PWM0 period setting register
pwm0_config	PWM0 configuration register
pwm0_interrupt	PWM0 interrupt register
pwm1_clkdiv	PWM1 clock division configuration register
pwm1_thre1	PWM1 first counter threshold configuration register

Name	Description
pwm1_thre2	PWM1 sencond counter threshold configuration register
pwm1_period	PWM1 period setting register
pwm1_config	PWM1 configuration register
pwm1_interrupt	PWM1 interrupt register
pwm2_clkdiv	PWM2 clock division configuration register
pwm2_thre1	PWM2 first counter threshold configuration register
pwm2_thre2	PWM2 sencond counter threshold configuration register
pwm2_period	PWM2 period setting register
pwm2_config	PWM2 configuration register
pwm2_interrupt	PWM2 interrupt register
pwm3_clkdiv	PWM3 clock division configuration register
pwm3_thre1	PWM3 first counter threshold configuration register
pwm3_thre2	PWM3 sencond counter threshold configuration register
pwm3_period	PWM3 period setting register
pwm3_config	PWM3 configuration register
pwm3_interrupt	PWM3 interrupt register
pwm4_clkdiv	PWM4 clock division configuration register
pwm4_thre1	PWM4 first counter threshold configuration register
pwm4_thre2	PWM4 sencond counter threshold configuration register
pwm4_period	PWM4 period setting register
pwm4_config	PWM4 configuration register
pwm4_interrupt	PWM4 interrupt register

12.4.1 pwm_int_config

Address: 0x4000a400

RSVD															
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

pwm_int_clear

pwm_interrupt_sts

Bits	Name	Type	Reset	Description
31:14	RSVD			
13:8	pwm_int_clear	w	6'd0	PWM channel interrupt clear
7:6	RSVD			
5:0	pwm_interrupt_sts	r	6'd0	PWM channel interrupt status

12.4.2 pwm0_clkdiv

Address: 0x4000a420

RSVD															
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

pwm_clk_div

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	pwm_clk_div	r/w	16'b0	PWM clock division

12.4.3 pwm0_thre1

Address: 0x4000a424

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | |

pwm_thre1

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	pwm_thre1	r/w	16'b0	PWM first counter threshold, can't be larger than pwm_thre2

12.4.4 pwm0_thre2

Address: 0x4000a428

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | |

pwm_thre2

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	pwm_thre2	r/w	16'd0	PWM sencond counter threshold, can't be smaller than pwm_thre1

12.4.5 pwm0_period

Address: 0x4000a42c

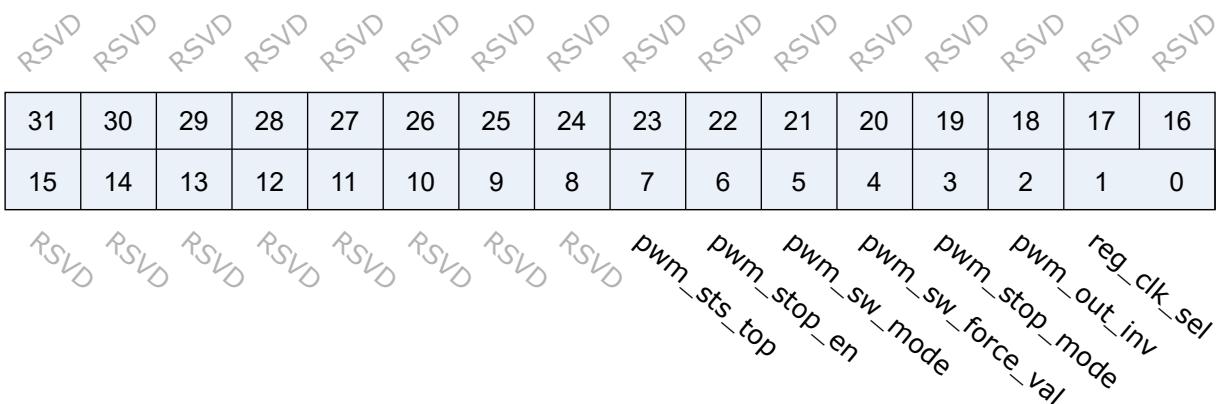
| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |

pwm_period

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	pwm_period	r/w	16'd0	PWM period setting

12.4.6 pwm0_config

Address: 0x4000a430



Bits	Name	Type	Reset	Description
31:8	RSVD			
7	pwm_sts_top	r	1'b0	PWM stop status
6	pwm_stop_en	r/w	1'b0	PWM stop enable
5	pwm_sw_mode	r/w	1'b0	PWM SW Mode setting
4	pwm_sw_force_val	r/w	1'b0	PWM SW Mode force value
3	pwm_stop_mode	r/w	1'b1	PWM stop mode, 1'b1 - graceful ; 1'b0 - abrupt
2	pwm_out_inv	r/w	1'b0	PWM invert output mode
1:0	reg_clk_sel	r/w	2'd0	PWM clock source select, 2'b00-xclk ; 2'b01-bclk ; others-f32k_clk

12.4.7 pwm0_interrupt

Address: 0x4000a434

RSVD	pwm_int_enable															
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	pwm_int_period_cnt

Bits	Name	Type	Reset	Description
31:17	RSVD			
16	pwm_int_enable	r/w	1'b0	PWM interrupt enable
15:0	pwm_int_period_cnt	r/w	16'd0	PWM interrupt period counter threshold

12.4.8 pwm1_clkdiv

Address: 0x4000a440

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|-------------|
| 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 | pwm_clk_div |

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	pwm_clk_div	r/w	16'b0	PWM clock division

12.4.9 pwm1_thre1

Address: 0x4000a444

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | |

pwm_thre1

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	pwm_thre1	r/w	16'b0	PWM first counter threshold, can't be larger than pwm_thre2

12.4.10 pwm1_thre2

Address: 0x4000a448

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | |

pwm_thre2

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	pwm_thre2	r/w	16'd0	PWM sencond counter threshold, can't be smaller than pwm_thre1

12.4.11 pwm1_period

Address: 0x4000a44c

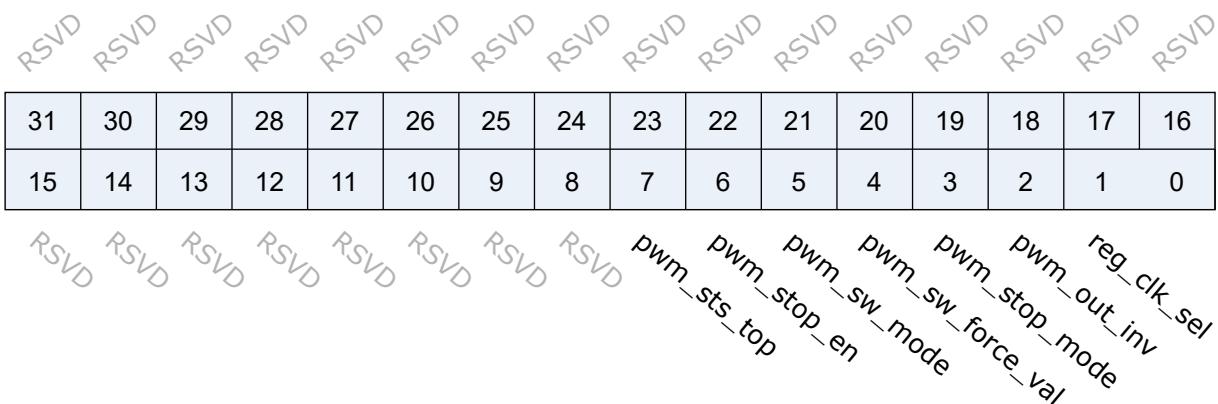
| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |

pwm_period

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	pwm_period	r/w	16'd0	PWM period setting

12.4.12 pwm1_config

Address: 0x4000a450



Bits	Name	Type	Reset	Description
31:8	RSVD			
7	pwm_sts_top	r	1'b0	PWM stop status
6	pwm_stop_en	r/w	1'b0	PWM stop enable
5	pwm_sw_mode	r/w	1'b0	PWM SW Mode setting
4	pwm_sw_force_val	r/w	1'b0	PWM SW Mode force value
3	pwm_stop_mode	r/w	1'b1	PWM stop mode, 1'b1 - graceful ; 1'b0 - abrupt
2	pwm_out_inv	r/w	1'b0	PWM invert output mode
1:0	reg_clk_sel	r/w	2'd0	PWM clock source select, 2'b00-xclk ; 2'b01-bclk ; others-f32k_clk

12.4.13 pwm1_interrupt

Address: 0x4000a454

RSVD	pwm_int_enable																
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		pwm_int_period_cnt

Bits	Name	Type	Reset	Description
31:17	RSVD			
16	pwm_int_enable	r/w	1'b0	PWM interrupt enable
15:0	pwm_int_period_cnt	r/w	16'd0	PWM interrupt period counter threshold

12.4.14 pwm2_clkdiv

Address: 0x4000a460

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|-------------|
| 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 | | pwm_clk_div |

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	pwm_clk_div	r/w	16'b0	PWM clock division

12.4.15 pwm2_thre1

Address: 0x4000a464

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | |

pwm_thre1

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	pwm_thre1	r/w	16'b0	PWM first counter threshold, can't be larger than pwm_thre2

12.4.16 pwm2_thre2

Address: 0x4000a468

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | |

pwm_thre2

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	pwm_thre2	r/w	16'd0	PWM sencond counter threshold, can't be smaller than pwm_thre1

12.4.17 pwm2_period

Address: 0x4000a46c

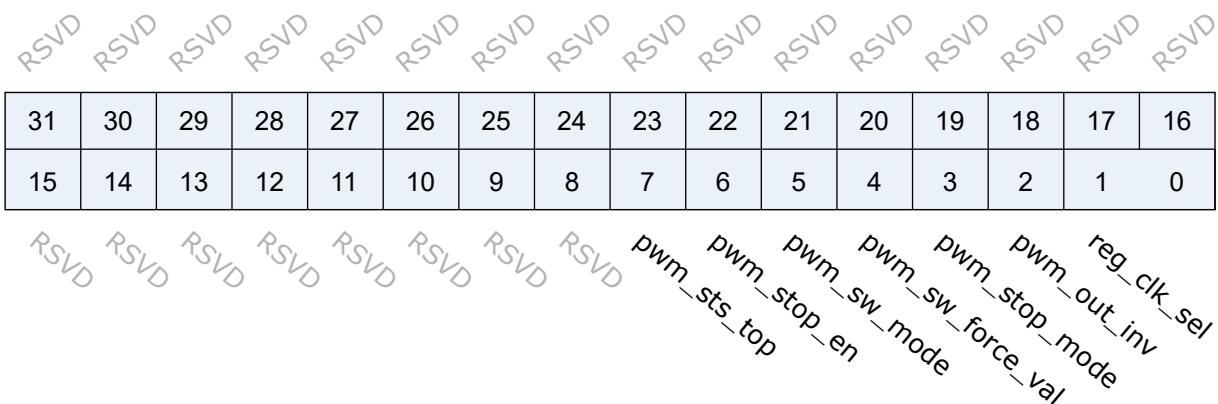
| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |

pwm_period

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	pwm_period	r/w	16'd0	PWM period setting

12.4.18 pwm2_config

Address: 0x4000a470



Bits	Name	Type	Reset	Description
31:8	RSVD			
7	pwm_sts_top	r	1'b0	PWM stop status
6	pwm_stop_en	r/w	1'b0	PWM stop enable
5	pwm_sw_mode	r/w	1'b0	PWM SW Mode setting
4	pwm_sw_force_val	r/w	1'b0	PWM SW Mode force value
3	pwm_stop_mode	r/w	1'b1	PWM stop mode, 1'b1 - graceful ; 1'b0 - abrupt
2	pwm_out_inv	r/w	1'b0	PWM invert output mode
1:0	reg_clk_sel	r/w	2'd0	PWM clock source select, 2'b00-xclk ; 2'b01-bclk ; others-f32k_clk

12.4.19 pwm2_interrupt

Address: 0x4000a474

RSVD	pwm_int_enable																
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		pwm_int_period_cnt

Bits	Name	Type	Reset	Description
31:17	RSVD			
16	pwm_int_enable	r/w	1'b0	PWM interrupt enable
15:0	pwm_int_period_cnt	r/w	16'd0	PWM interrupt period counter threshold

12.4.20 pwm3_clkdiv

Address: 0x4000a480

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|-------------|
| 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 | | pwm_clk_div |

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	pwm_clk_div	r/w	16'b0	PWM clock division

12.4.21 pwm3_thre1

Address: 0x4000a484

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | |

pwm_thre1

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	pwm_thre1	r/w	16'b0	PWM first counter threshold, can't be larger than pwm_thre2

12.4.22 pwm3_thre2

Address: 0x4000a488

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | |

pwm_thre2

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	pwm_thre2	r/w	16'd0	PWM sencond counter threshold, can't be smaller than pwm_thre1

12.4.23 pwm3_period

Address: 0x4000a48c

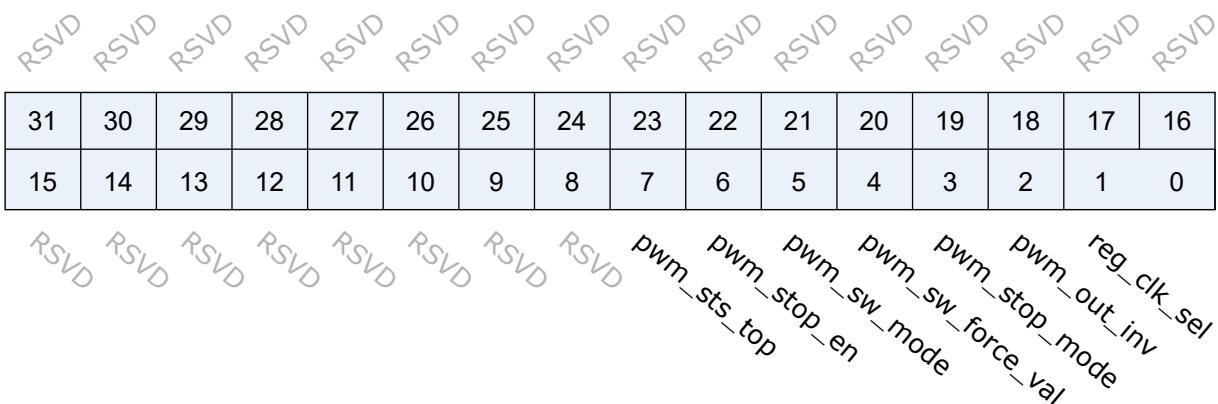
| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |

pwm_period

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	pwm_period	r/w	16'd0	PWM period setting

12.4.24 pwm3_config

Address: 0x4000a490



Bits	Name	Type	Reset	Description
31:8	RSVD			
7	pwm_sts_top	r	1'b0	PWM stop status
6	pwm_stop_en	r/w	1'b0	PWM stop enable
5	pwm_sw_mode	r/w	1'b0	PWM SW Mode setting
4	pwm_sw_force_val	r/w	1'b0	PWM SW Mode force value
3	pwm_stop_mode	r/w	1'b1	PWM stop mode, 1'b1 - graceful ; 1'b0 - abrupt
2	pwm_out_inv	r/w	1'b0	PWM invert output mode
1:0	reg_clk_sel	r/w	2'd0	PWM clock source select, 2'b00-xclk ; 2'b01-bclk ; others-f32k_clk

12.4.25 pwm3_interrupt

Address: 0x4000a494

RSVD	pwm_int_enable																
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		pwm_int_period_cnt

Bits	Name	Type	Reset	Description
31:17	RSVD			
16	pwm_int_enable	r/w	1'b0	PWM interrupt enable
15:0	pwm_int_period_cnt	r/w	16'd0	PWM interrupt period counter threshold

12.4.26 pwm4_clkdiv

Address: 0x4000a4a0

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|-------------|
| 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 | | pwm_clk_div |

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	pwm_clk_div	r/w	16'b0	PWM clock division

12.4.27 pwm4_thre1

Address: 0x4000a4a4

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | |

pwm_thre1

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	pwm_thre1	r/w	16'b0	PWM first counter threshold, can't be larger than pwm_thre2

12.4.28 pwm4_thre2

Address: 0x4000a4a8

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | |

pwm_thre2

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	pwm_thre2	r/w	16'd0	PWM sencond counter threshold, can't be smaller than pwm_thre1

12.4.29 pwm4_period

Address: 0x4000a4ac

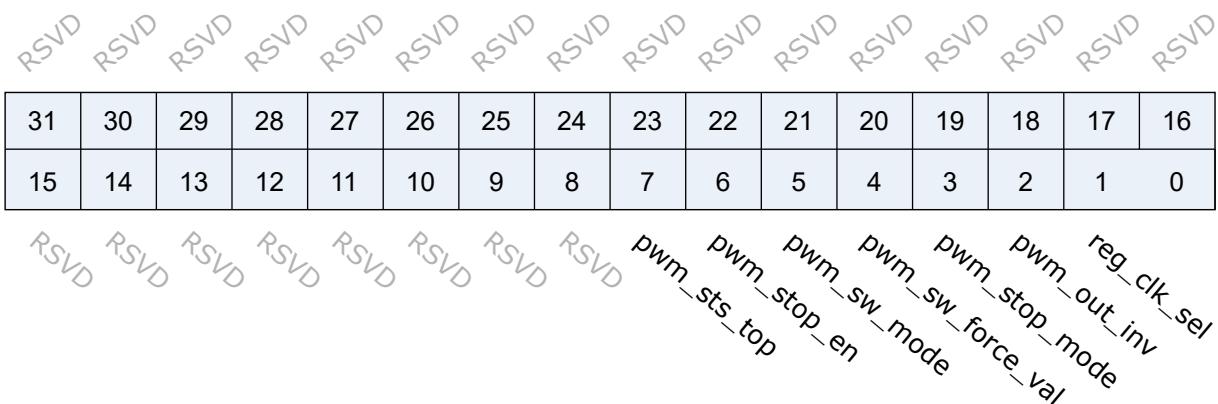
| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |

pwm_period

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	pwm_period	r/w	16'd0	PWM period setting

12.4.30 pwm4_config

Address: 0x4000a4b0



Bits	Name	Type	Reset	Description
31:8	RSVD			
7	pwm_sts_top	r	1'b0	PWM stop status
6	pwm_stop_en	r/w	1'b0	PWM stop enable
5	pwm_sw_mode	r/w	1'b0	PWM SW Mode setting
4	pwm_sw_force_val	r/w	1'b0	PWM SW Mode force value
3	pwm_stop_mode	r/w	1'b1	PWM stop mode, 1'b1 - graceful ; 1'b0 - abrupt
2	pwm_out_inv	r/w	1'b0	PWM invert output mode
1:0	reg_clk_sel	r/w	2'd0	PWM clock source select, 2'b00-xclk ; 2'b01-bclk ; others-f32k_clk

12.4.31 pwm4_interrupt

Address: 0x4000a4b4

RSVD	pwm_int_enable															
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	pwm_int_period_cnt

Bits	Name	Type	Reset	Description
31:17	RSVD			
16	pwm_int_enable	r/w	1'b0	PWM interrupt enable
15:0	pwm_int_period_cnt	r/w	16'd0	PWM interrupt period counter threshold

13.1 TIMER introduction

The chip has two 32-bit counters, each of which can independently control and configure its parameters and clock frequency.

There is a watchdog counter in the chip. Unpredictable software or hardware behavior may cause the application to malfunction. A watchdog timer can help the system recover from it. If the current time exceeds the predetermined time, but the dog is not fed or closed Timer, which can trigger interrupt or system reset according to the setting.

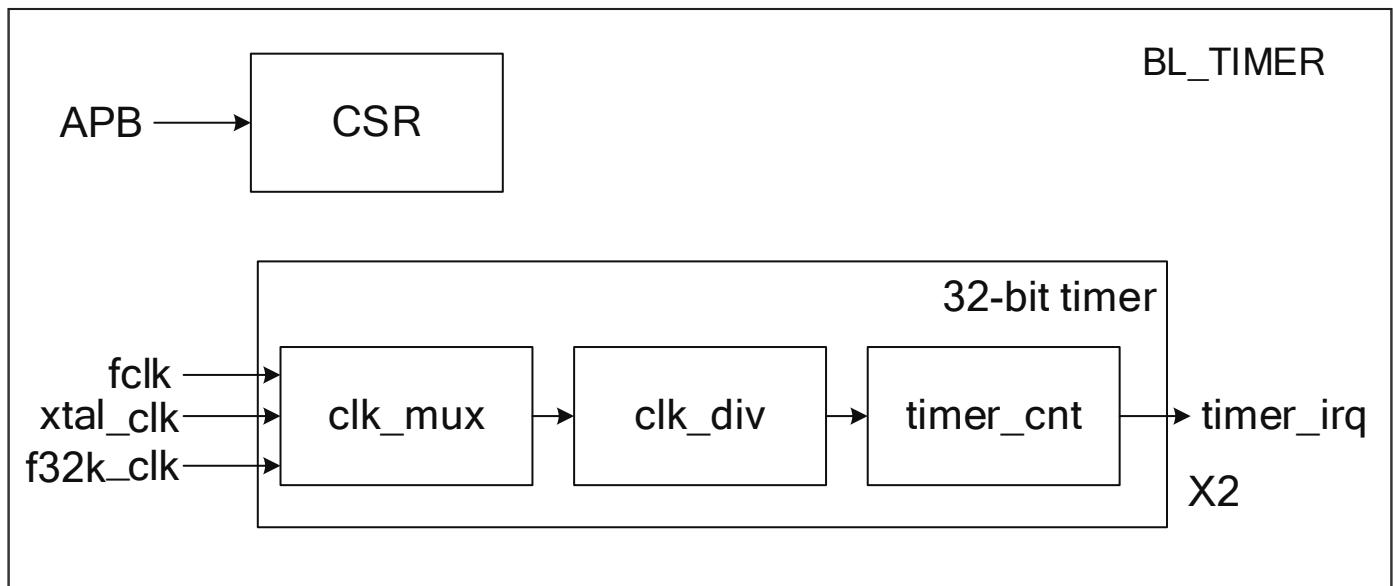


Fig. 13.1: Timer block diagram

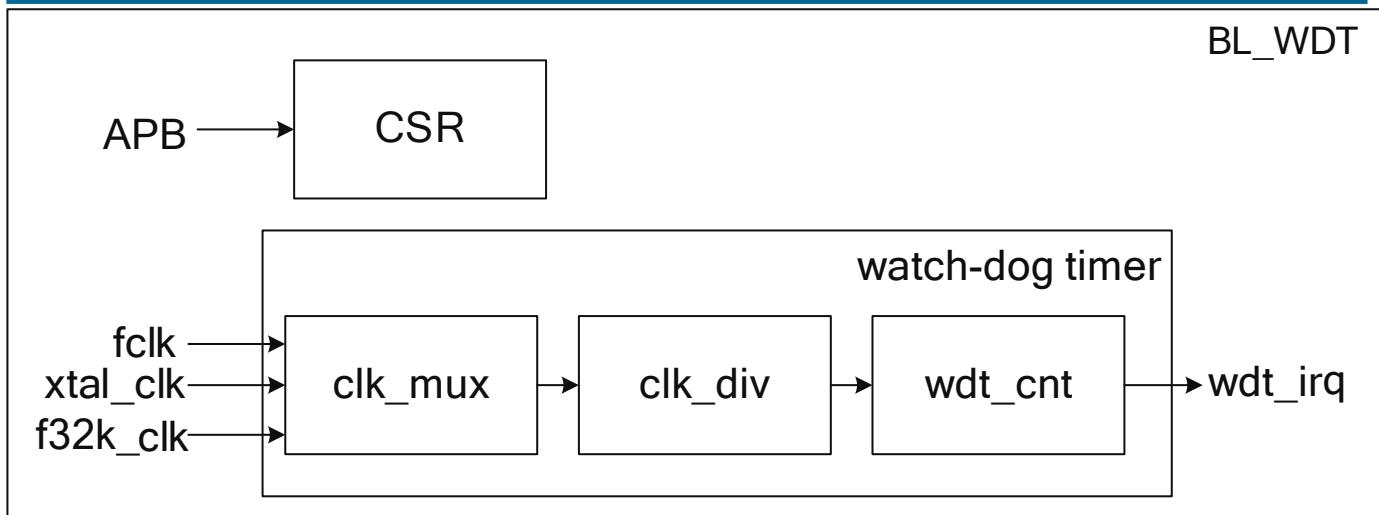


Fig. 13.2: Watchdog timer block diagram

13.2 TIMER main features

- Multiple clock source options
- 8-bit clock divider with a division factor of 1-256.
- Two 32-bit timers
- Each timer contains three alarm value settings, which can be set independently to alarm when each alarm value overflows
- Support FreeRun mode and PreLoad mode
- 16-bit watchdog timer
- Supports write password protection to prevent system abnormalities caused by incorrect settings
- Support two watchdog overflow methods: interrupt or reset

13.3 TIMER function description

13.3.1 8-bit divider

There are three types of Watchdog timer clocks:

- Fclk—System master clock
- 32K-32K clock
- Xtal—External crystal

There are four timer clock sources:

- Fclk—System master clock
- 32K-32K clock
- 1K-1K clock (32K frequency division)
- Xtal—External crystal

Each counter has its own 8-bit frequency divider. The selected clock can be divided by 1-256 through APB. Specifically, when it is set to 0, it means no frequency division, and when it is set to 1, it divides it by 2. The maximum frequency division coefficient is 256, the counter will use the divided clock as the unit of the counting cycle, each time a counting cycle is increased by one.

13.3.2 General timer operating mode

Each general-purpose timer includes three comparators, a counter and a preload register. When the clock source is set and the timer is started, the counter starts to count up. When the counter value is equal to the comparator, the comparison is performed. When the flag is set, a compare interrupt is generated.

The initial value of the counter depends on the timing mode. In FreeRun mode, the initial value of the counter is 0, and then counts up. When it reaches the maximum value, it starts counting from 0 again.

In PreLoad mode, the initial value of the counter is the value of the PreLoad register and then counts up. When the PreLoad condition is met, the value of the counter is set to the value of the PreLoad register, and then the counter starts to count up again. During the counting process, once the value of the counter matches one of the three comparators, the comparator's comparison flag will be set and a corresponding comparison interrupt can be generated.

If the value of the preload register is 10, the value of Comparator 0 is 13, the value of Comparator 1 is 16, and the value of Comparator 2 is 19, the working sequence of the timer in PreLoad mode is as follows:

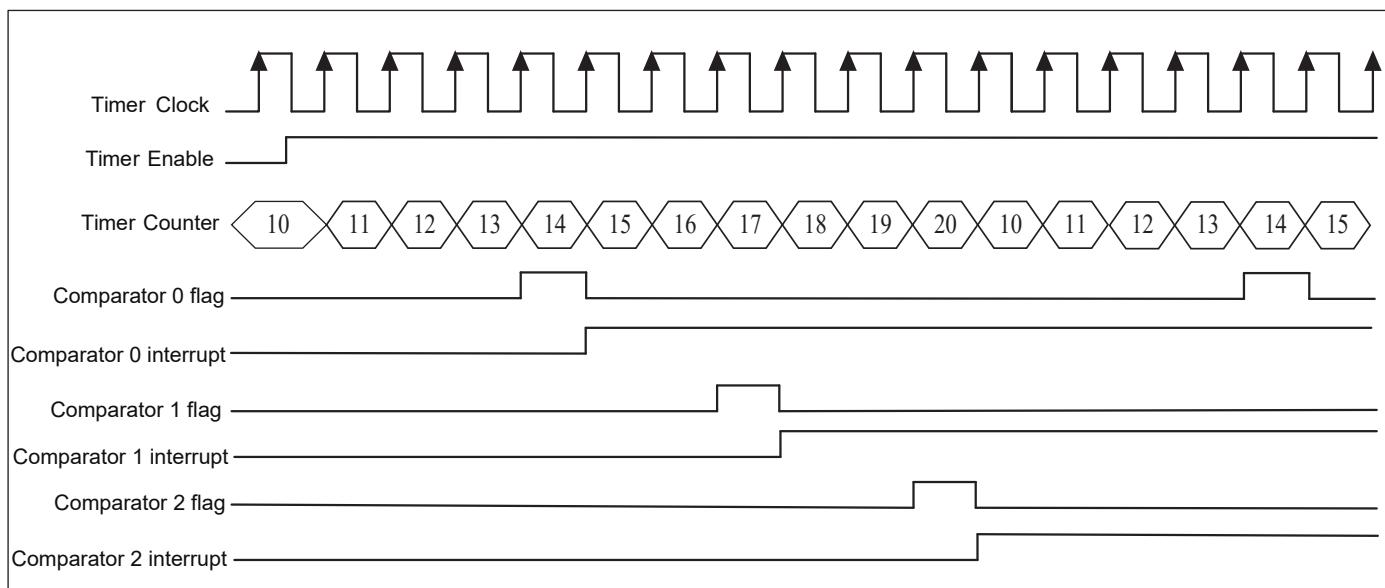


Fig. 13.3: Timer Preload

In FreeRun mode, the timer working sequence is basically the same as PreLoad, the difference is that the counter will start to accumulate from 0 to the maximum value. The mechanism of the generated compare flags and compare interrupts is the same as in PreLoad mode.

13.3.3 Watchdog timer operating mode

The watchdog timer includes a counter and a comparator. The counter counts up from 0. If the counter is reset (feed the dog), it starts counting up from 0 again. When the counter value is equal to the comparator, a comparison interrupt signal or a system reset signal will be generated, and the user can choose to use one of them as required.

The watchdog counter is incremented by one in each counting cycle unit. Software can reset the watchdog counter to zero at any point in time through the APB.

If the value of the comparator is 6, the working sequence of Watchdog is shown in the figure below:

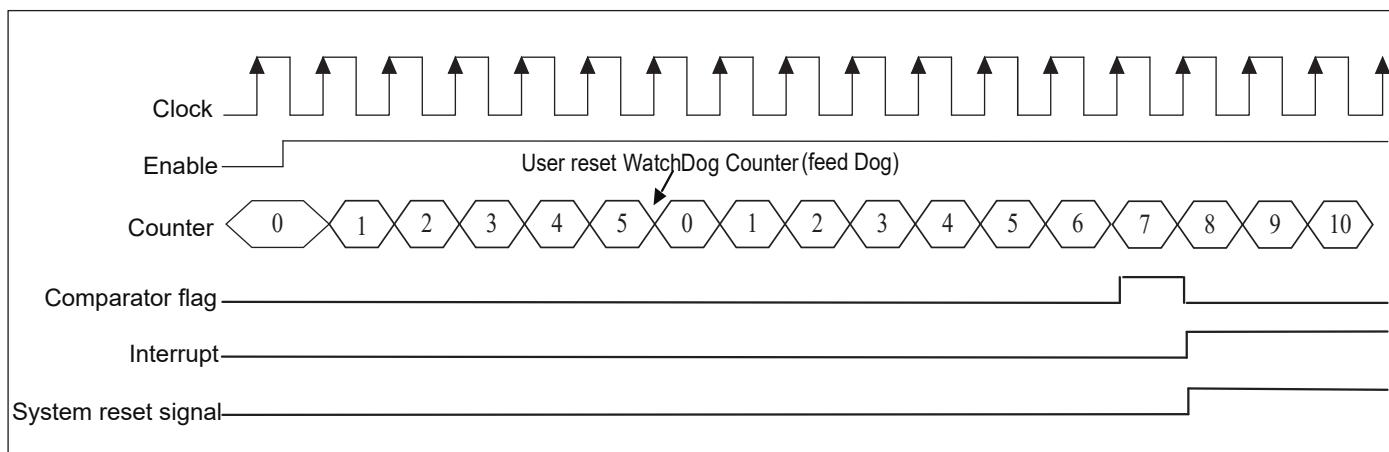


Fig. 13.4: Watchdog timing

13.3.4 Alarm setting

Each counter has three comparison values, and can set whether each comparison value triggers an alarm interrupt. When the counter matches the comparison value and the setting will alarm, the counter will notify the processor through the interrupt.

The software can read through the APB whether an alarm has occurred and which comparison value triggered the alarm interrupt. When the alarm interrupt is cleared, the alarm status is also cleared simultaneously.

13.3.5 Watchdog alarm

A comparison value can be set for each counter. When the software fails to reset the watchdog counter to zero due to a system error, which causes the watchdog counter to exceed the comparison value, a watchdog alarm is triggered. There are two types of alarms. The first is to perform necessary actions through interrupt notification software. The second is to enter the system watchdog reset. When the watchdog reset is triggered, it will notify the system reset controller and prepare for system reset. When everything is ready, enter the system watchdog reset. It is worth noting that software can read the WSR register through APB to know if a watchdog system reset has occurred.

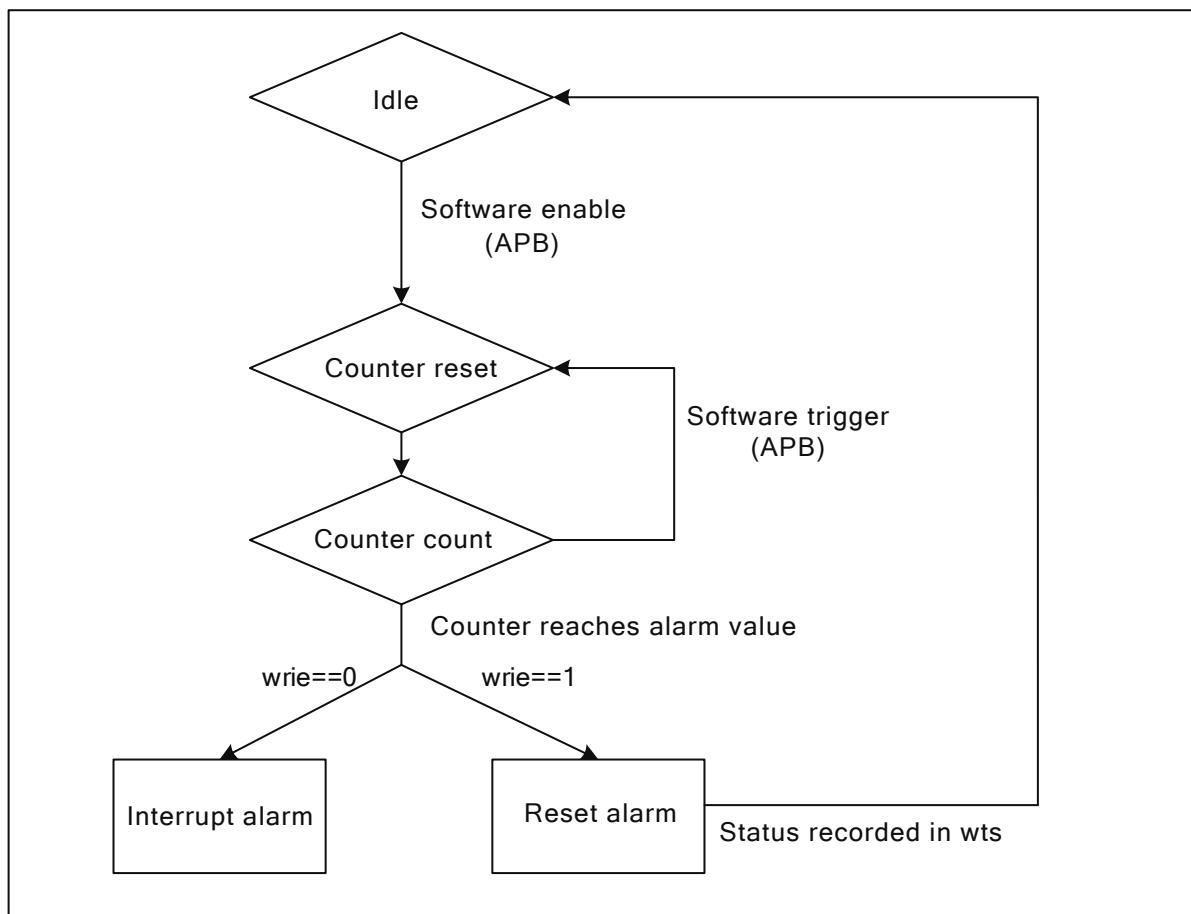


Fig. 13.5: Watchdog alarm mechanism

13.4 Register description

Name	Description
TCCR	Timer clock source configuration register
TMR2_0	Timer2 match register 0
TMR2_1	Timer2 match register 1
TMR2_2	Timer2 match register 2

Name	Description
TMR2_0	Timer3 match register 0
TMR2_1	Timer3 match register 1
TMR2_2	Timer3 match register 2
TCR2	Timer2 counter register
TCR3	Timer3 counter register
TMSR2	Timer2 match register status
TMSR3	Timer3 match register status
TIER2	Timer2 match interrupt enable register
TIER3	Timer3 match interrupt enable register
TPLVR2	Timer2 pre-load value register
TPLVR3	Timer3 pre-load value register
TPLCR2	Timer2 pre-load control register
TPLCR3	Timer3 pre-load control register
WMER	WDT reset/interrupt mode register
WMR	WDT counter match value register
WVR	WDT counter value register
WSR	WDT timer reset indication register
TICR2	Timer2 Interrupt clear control register
TICR3	Timer3 Interrupt clear control register
WICR	WDT Interrupt clear register
TCER	Timer count enable register
TCMR	Timer count mode register
TILR2	Timer2 match interrupt mode register
TILR3	Timer3 match interrupt mode register
WCR	WDT timer count reset register
WFAR	WDT access key1 register
WSAR	WDT access key2 register
TCVWR2	Timer2 capture value of counter register
TCVWR3	Timer3 capture value of counter register

Name	Description
TCVSYN2	Timer2 synchronous value of counter register
TCVSYN3	Timer3 synchronous value of counter register
TCDR	WDT/Timer clock division register

13.4.1 TCCR

Address: 0x4000a500

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | |

RSVD RSVD RSVD RSVD RSVD RSVD cs_wdt RSVD cs_2 RSVD cs_1 RSVD RSVD

Bits	Name	Type	Reset	Description
31:10	RSVD			
9:8	cs_wdt	r/w	2'd0	Clock Source for Timer #1/#2/#3/WDT 2'd0 - fclk 2'd1 - f32k_clk 2'd2 - 1 kHz 2'd3 - PLL 32MHz
7	RSVD			
6:5	cs_2	r/w	2'd0	
4	RSVD			
3:2	cs_1	r/w	2'd0	
1:0	RSVD			

13.4.2 TMR2_0

Address: 0x4000a510

tmr_3_0

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

tmr_3_0

Bits	Name	Type	Reset	Description
31:0	tmr_2_0	r/w	32'hffffffffff	Timer2 match register 0

13.4.3 TMR2_1

Address: 0x4000a514

tmr_3_1

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

tmr_3_1

Bits	Name	Type	Reset	Description
31:0	tmr_2_1	r/w	32'hffffffffff	Timer2 match register 1

13.4.4 TMR2_2

Address: 0x4000a518

tmr_3_2

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

tmr_3_2

Bits	Name	Type	Reset	Description
31:0	tmr_2_2	r/w	32'hffffffffff	Timer2 match register 2

13.4.5 TMR2_0

Address: 0x4000a51c

tmr_3_0

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

tmr_3_0

Bits	Name	Type	Reset	Description
31:0	tmr_3_0	r/w	32'hffffffffff	Timer3 match register 0

13.4.6 TMR2_1

Address: 0x4000a520

tmr_3_1

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

tmr_3_1

Bits	Name	Type	Reset	Description
31:0	tmr_3_1	r/w	32'hffffffffff	Timer3 match register 1

13.4.7 TMR2_2

Address: 0x4000a524

tmr_3_2

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

tmr_3_2

Bits	Name	Type	Reset	Description
31:0	tmr_3_2	r/w	32'hffffffffff	Timer3 match register 2

13.4.8 TCR2

Address: 0x4000a52c

tcr2_counter

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

tcr2_counter

Bits	Name	Type	Reset	Description
31:0	tcr2_counter	r	32'h0	Timer2 counter register

13.4.9 TCR3

Address: 0x4000a530

tcr3_counter

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

tcr3_counter

Bits	Name	Type	Reset	Description
31:0	tcr3_counter	r	32'h0	Timer3 counter register

13.4.10 TMSR2

Address: 0x4000a538

RSVD																
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	

RSVD tmsr2_2 tmsr2_1 tmsr2_0

Bits	Name	Type	Reset	Description
31:3	RSVD			

Bits	Name	Type	Reset	Description
2	tmsr2_2	r	1'b0	Timer2 match register 2 status/Clear interrupt would also clear this bit
1	tmsr2_1	r	1'b0	Timer2 match register 1 status/Clear interrupt would also clear this bit
0	tmsr2_0	r	1'b0	Timer2 match register 0 status/Clear interrupt would also clear this bit

13.4.11 TMSR3

Address: 0x4000a53c

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | |

RSVD tmsr3_2 tmsr3_1 tmsr3_0

Bits	Name	Type	Reset	Description
31:3	RSVD			
2	tmsr3_2	r	1'b0	Timer3 match register 2 status/Clear interrupt would also clear this bit
1	tmsr3_1	r	1'b0	Timer3 match register 1 status/Clear interrupt would also clear this bit
0	tmsr3_0	r	1'b0	Timer3 match register 0 status/Clear interrupt would also clear this bit

13.4.12 TIER2

Address: 0x4000a544

RSVD	tier2_2	tier2_1	tier2_0																											
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															

Bits	Name	Type	Reset	Description
31:3	RSVD			
2	tier2_2	r/w	1'b0	Timer2 match register 2 interrupt enable register
1	tier2_1	r/w	1'b0	Timer2 match register 1 interrupt enable register
0	tier2_0	r/w	1'b0	Timer2 match register 0 interrupt enable register

13.4.13 TIER3

Address: 0x4000a548

RSVD	tier3_2	tier3_1	tier3_0																										
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														

Bits	Name	Type	Reset	Description
31:3	RSVD			
2	tier3_2	r/w	1'b0	Timer3 match register 2 interrupt enable register
1	tier3_1	r/w	1'b0	Timer3 match register 1 interrupt enable register
0	tier3_0	r/w	1'b0	Timer3 match register 0 interrupt enable register

13.4.14 TPLVR2

Address: 0x4000a550

tplvr2

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

tplvr2

Bits	Name	Type	Reset	Description
31:0	tplvr2	r/w	32'h0	Timer2 pre-load value register

13.4.15 TPLVR3

Address: 0x4000a554

tplvr3

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

tplvr3

Bits	Name	Type	Reset	Description
31:0	tplvr3	r/w	32'h0	Timer3 pre-load value register

13.4.16 TPLCR2

Address: 0x4000a55c

RSVD																
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	

RSVD RSVD

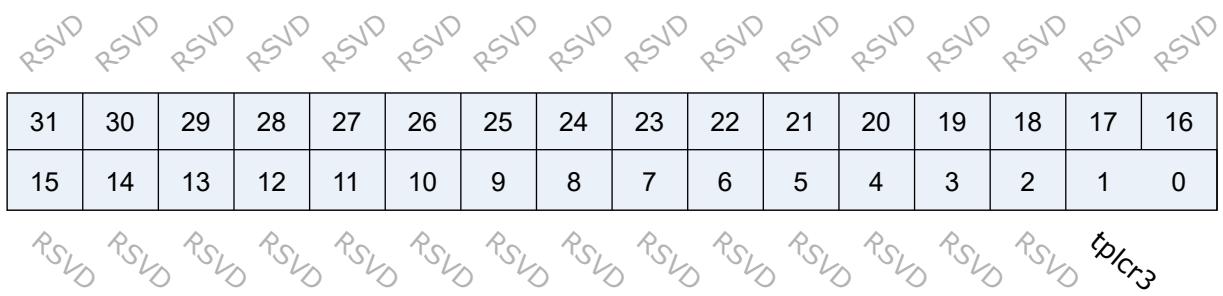
Bits	Name	Type	Reset	Description
31:2	RSVD			



Bits	Name	Type	Reset	Description
1:0	tplcr2	r/w	2'h0	Timer2 pre-load control register 2'd0 - No pre-load 2'd1 - Pre-load with match comparator 0 2'd2 - Pre-load with match comparator 1 2'd3 - Pre-load with match comparator 2

13.4.17 TPLCR3

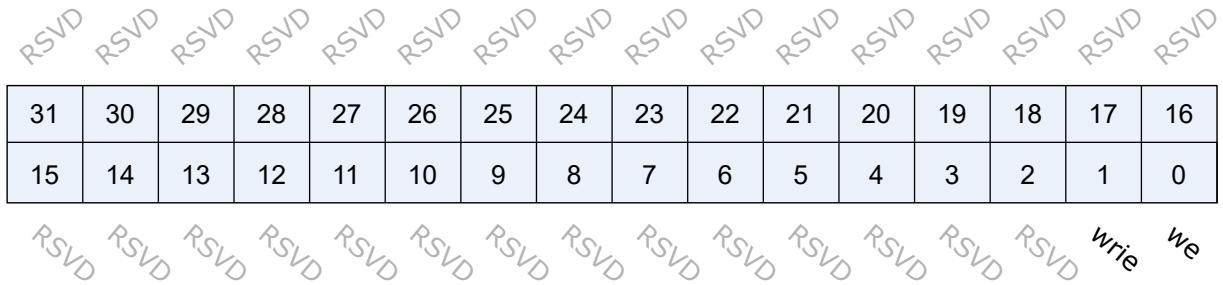
Address: 0x4000a560



Bits	Name	Type	Reset	Description
31:2	RSVD			
1:0	tplcr3	r/w	2'h0	Timer3 pre-load control register 2'd0 - No pre-load 2'd1 - Pre-load with match comparator 0 2'd2 - Pre-load with match comparator 1 2'd3 - Pre-load with match comparator 2

13.4.18 WMER

Address: 0x4000a564



Bits	Name	Type	Reset	Description
31:2	RSVD			
1	wrie	r/w	1'b0	WDT reset/interrupt mode register 1'b0 - WDT expiration to generate interrupt 1'b1 - WDT expiration to generate reset source
0	we	r/w	1'b0	WDT enable register

13.4.19 WMR

Address: 0x4000a568

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | | |

wmr

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	wmr	r/w	16'hfff	WDT counter match value register

13.4.20 WVR

Address: 0x4000a56c

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | | |

wvr

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	wvr	r	16'h0	WDT counter value register

13.4.21 WSR

Address: 0x4000a570

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | |

RSVD wts

Bits	Name	Type	Reset	Description
31:1	RSVD			
0	wts	r/w	1'b0	WDT timer reset indication, Indicates that reset was caused by the WDT. (Write)1'b0 - clear the WDT reset status (Write)1'b1 - no affect (Read)1'b0 - Watchdog timer did not cause reset because this bit was cleared (Read)1'b1 - Watchdog timer caused reset

13.4.22 TICR2

Address: 0x4000a578

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | |

RSVD tclr2_2 tclr2_1 tclr2_0

Bits	Name	Type	Reset	Description
31:3	RSVD			
2	tclr2_2	w	1'b0	Timer2 Interrupt clear for match comparator 2
1	tclr2_1	w	1'b0	Timer2 Interrupt clear for match comparator 1
0	tclr2_0	w	1'b0	Timer2 Interrupt clear for match comparator 0

13.4.23 TICR3

Address: 0x4000a57c

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | |

RSVD tclr3_2 tclr3_1 tclr3_0

Bits	Name	Type	Reset	Description
31:3	RSVD			
2	tclr3_2	w	1'b0	Timer3 Interrupt clear for match comparator 2
1	tclr3_1	w	1'b0	Timer3 Interrupt clear for match comparator 1
0	tclr3_0	w	1'b0	Timer3 Interrupt clear for match comparator 0

13.4.24 WICR

Address: 0x4000a580

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | |

RSVD wiclr

Bits	Name	Type	Reset	Description
31:1	RSVD			
0	wiclr	w	1'b0	WDT Interrupt clear register

13.4.25 TCER

Address: 0x4000a584

RSVD																
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	

RSVD RSVD

Bits	Name	Type	Reset	Description
31:3	RSVD			
2	timer3_en	r/w	1'b0	Timer3 count enable
1	timer2_en	r/w	1'b0	Timer2 count enable
0	RSVD			

13.4.26 TCMR

Address: 0x4000a588

RSVD																
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	

RSVD RSVD

Bits	Name	Type	Reset	Description
31:3	RSVD			
2	timer3_mode	r/w	1'b0	Timer1/2/3 count mode register 1'b0 - pre-load mode 1'b1 - free run mode
1	timer2_mode	r/w	1'b0	
0	RSVD			

13.4.27 TILR2

Address: 0x4000a590

RSVD															
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

RSVD RSVD RSVD
tilr2_2 tilr2_1 tilr2_0

Bits	Name	Type	Reset	Description
31:3	RSVD			
2	tilr2_2	r/w	1'b0	Timer2 match 0/1/2 interrupt mode register 1'b0 - level interrupt 1'b1 - pulse interrupt
1	tilr2_1	r/w	1'b0	
0	tilr2_0	r/w	1'b0	

13.4.28 TILR3

Address: 0x4000a594

RSVD																
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	

RSVD RSVD RSVD
tilr3_2 tilr3_1 tilr3_0

Bits	Name	Type	Reset	Description
31:3	RSVD			
2	tilr3_2	r/w	1'b0	Timer3 match 0/1/2 interrupt mode register 1'b0 - level interrupt 1'b1 - pulse interrupt
1	tilr3_1	r/w	1'b0	
0	tilr3_0	r/w	1'b0	

13.4.29 WCR

Address: 0x4000a598

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | |

wcr

Bits	Name	Type	Reset	Description
31:1	RSVD			
0	wcr	w	1'b0	WDT timer count reset register

13.4.30 WFAR

Address: 0x4000a59c

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | |

wfar

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	wfar	w	16'b0	WDT access key1 - 16'hBABA

13.4.31 WSAR

Address: 0x4000a5a0

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | |

wsar

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	wsar	w	16'b0	WDT access key2 - 16'hEB10

13.4.32 TCVWR2

Address: 0x4000a5a8

tcvwr2

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

tcvwr2

Bits	Name	Type	Reset	Description
31:0	tcvwr2	r	32'h0	Timer2 capture value of counter

13.4.33 TCVWR3

Address: 0x4000a5ac

tcvwr3

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

tcvwr3

Bits	Name	Type	Reset	Description
31:0	tcvwr3	r	32'h0	Timer3 capture value of counter

13.4.34 TCVSYN2

Address: 0x4000a5b4

tcvsyn2

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

tcvsyn2

Bits	Name	Type	Reset	Description
31:0	tcvsyn2	r	32'h0	Timer2 synchronous value of counter

13.4.35 TCVSYN3

Address: 0x4000a5b8

tcvsyn3

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

tcvsyn3

Bits	Name	Type	Reset	Description
31:0	tcvsyn3	r	32'h0	Timer3 synchronous value of counter

13.4.36 TCDR

Address: 0x4000a5bc

wcdr

tcdr3

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

tcdr2

RSVD RSVD RSVD RSVD RSVD RSVD RSVD RSVD

Bits	Name	Type	Reset	Description
31:24	wcdr	r/w	8'h0	WDT clock division value register
23:16	tcdr3	r/w	8'h0	Timer3 clock division value register
15:8	tcdr2	r/w	8'h0	Timer2 clock division value register
7:0	RSVD			

14

QDEC

14.1 QDEC introduction

The quadrature decoder is used to decode the two sets of pulses with a phase difference of 90 degrees generated by the dual-path rotary encoder into the corresponding speed and direction of rotation.

14.2 QDEC main features

- Three sets of QDEC are available
- The clock source of QDEC can be 32K (f32k_clk) or 32M (xclk). It is recommended to select 32M as the clock source during normal operation, and 32K is recommended when entering sleep mode and wish to be awakened by QDEC.
- Supports 5-digit frequency division value, which can be divided from 1 to 32.
- 16-bit pulse count range (-32768~32767 pulse/sample)
- 12 configurable sample periods (32us~131ms per sample at 1MHz)
- 16-bit configurable report period (0~65535 sample/report)
- Built-in a LED function that can flash with sampling (LED on/off 0~511 us/sample)
- Interrupt can be configured (sample interrupt, report interrupt, error interrupt, overflow interrupt)
- Can be configured as a wake-up source for PDS (clock source needs to be configured as 32k)

14.3 QDEC function description

The expected operating frequency of QDEC is 1MHz, and the faster the detection speed, the higher the operating frequency required.

Each sampling will decode the A/B two-phase pulse output by the encoder into high and low levels. Compare the previous sampling results to get the current encoder rotation direction and pulse count change (clockwise rotation +1, counterclockwise rotation -1 , No change, no change, error report and count). After the sampling times set by the report, the rotation direction and pulse count of the encoder during this period can be obtained, and the average value of the rotational speed direction during the report period can be solved accordingly.

The period of each sampling can be configured. When the working frequency is 1MHz, the minimum is 32us for one sampling, and the maximum is 131ms for one sampling.

The interrupt can be configured as a single sampling end trigger (sample interrupt) and multiple sampling end trigger (report interrupt) to flexibly measure the speed.

Configurable LED blinking function, blinking frequency=LED cycle/sampling cycle, each blinking on/off is determined by the LED polarity.

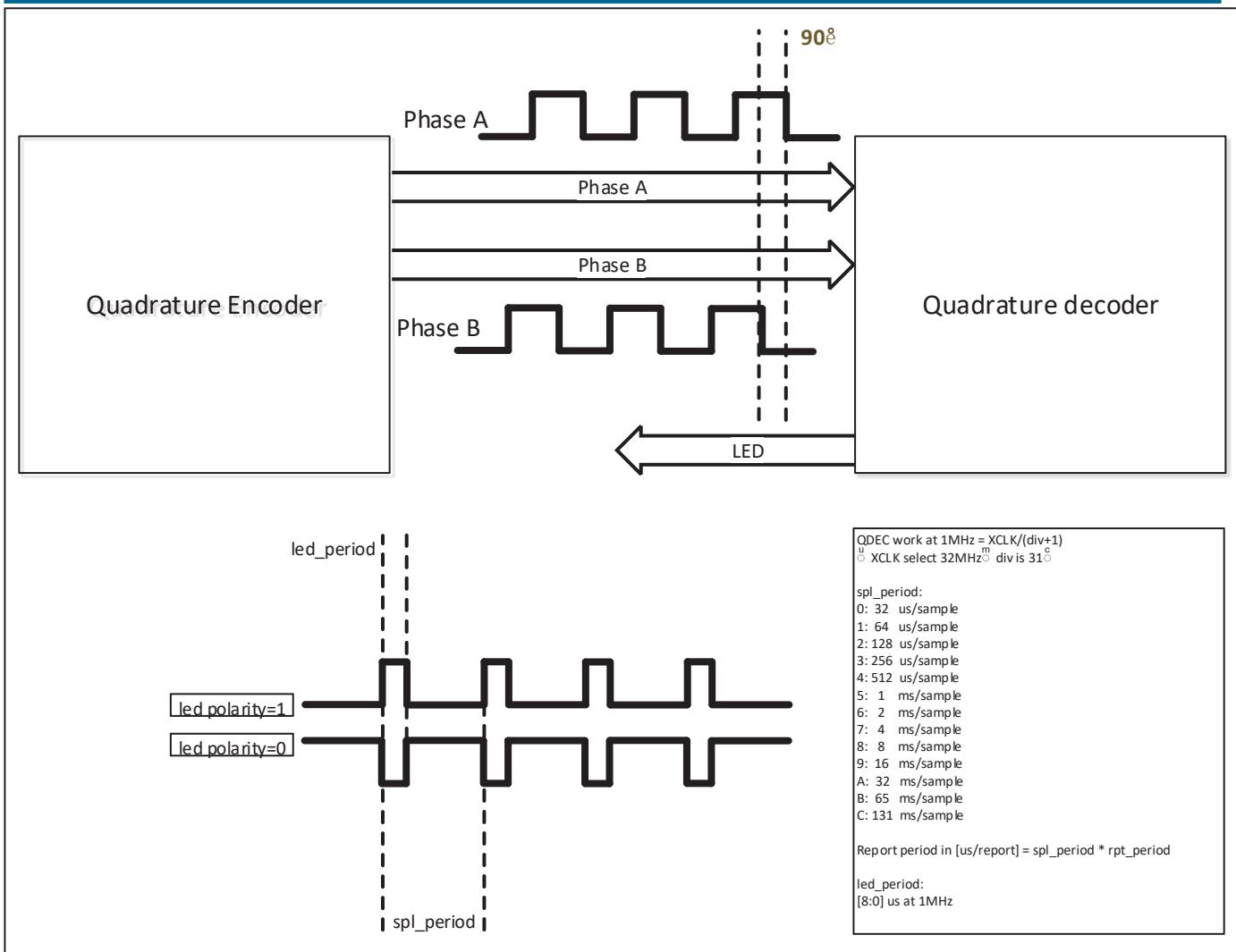


Fig. 14.1: QDEC functional block diagram

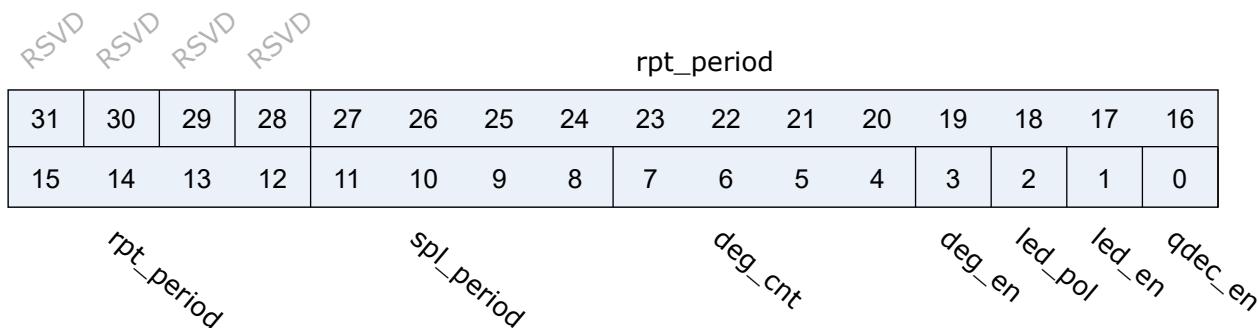
14.4 Register description

Name	Description
qdec0_ctrl0	QDEC0 control0
qdec0_ctrl1	QDEC0 control1
qdec0_value	QDEC0 value
qdec0_int_en	QDEC0 interrupt enable
qdec0_int_sts	QDEC0 interrupt status
qdec0_int_clr	QDEC0 interrupt clear
qdec1_ctrl0	QDEC1 control0

Name	Description
qdec1_ctrl1	QDEC1 control1
qdec1_value	QDEC1 value
qdec1_int_en	QDEC1 interrupt enable
qdec1_int_sts	QDEC1 interrupt status
qdec1_int_clr	QDEC1 interrupt clear
qdec2_ctrl0	QDEC2 control0
qdec2_ctrl1	QDEC2 control1
qdec2_value	QDEC2 value
qdec2_int_en	QDEC2 interrupt enable
qdec2_int_sts	QDEC2 interrupt status
qdec2_int_clr	QDEC2 interrupt clear

14.4.1 qdec0_ctrl0

Address: 0x4000a800

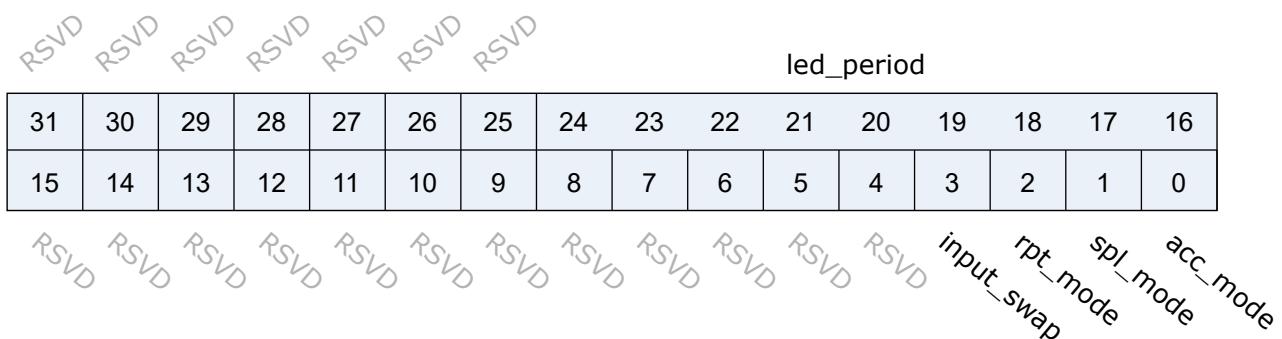


Bits	Name	Type	Reset	Description
31:28	RSVD			
27:12	rpt_period	r/w	16'd10	"RPT" report period in [samples/report]. Specifies the number of samples to be accumulated in the ACC1 register before the RPT_RDY and DBL_RDY events can be generated "RPT_US" report period in [us/report] = SP * RP

Bits	Name	Type	Reset	Description
11:8	spl_period	r/w	4'h2	<p>"SPL" sample period in [us/sample]. The SAMPLE register will be updated for every new sample</p> <p>0: 32 us 1: 64 2: 128 3: 256 4: 512 5: 1 ms 6: 2 7: 4 8: 8 9: 16 A: 32 B: 65 C: 131</p>
7:4	deg_cnt	r/w	0	
3	deg_en	r/w	0	deglitch
2	led_pol	r/w	1	led polarity
1	led_en	r/w	0	
0	qdec_en	r/w	0	

14.4.2 qdec0_ctrl1

Address: 0x4000a804



Bits	Name	Type	Reset	Description
31:25	RSVD			
24:16	led_period	r/w	0	Period in us the LED is switched on prior to sampling
15:4	RSVD			

Bits	Name	Type	Reset	Description
3	input_swap	r/w	0	input a/b swap
2	rpt_mode	r/w	0	rpt option 0: Count time only if sample change 1: Continue time
1	spl_mode	r/w	0	spl option 0: Stop sample if rpt_rdy 1: Continue sample
0	acc_mode	r/w	1	acc option 0: Stop accumulate if overflow 1: Continue accumulate

14.4.3 qdec0_value

Address: 0x4000a808

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	acc2_val
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	acc1_val

Bits	Name	Type	Reset	Description
31:30	RSVD			
29:28	spl_val	r	0	Sample value. Direction of last change 00: no change 01: clockwise 11: counter-clockwise 10: Error
27:20	RSVD			
19:16	acc2_val	r	0	Double error accumulation (0 15)
15:0	acc1_val	r	0	Sample accumulation (-1024 1023)

14.4.4 qdec0_int_en

Address: 0x4000a810

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | |

RSVD RSVD

overflow_en dbl_rdy_en spl_rdy_en rpt_rdy_en

Bits	Name	Type	Reset	Description
31:4	RSVD			
3	overflow_en	r/w	0	
2	dbl_rdy_en	r/w	0	
1	spl_rdy_en	r/w	0	
0	rpt_rdy_en	r/w	1	

14.4.5 qdec0_int_sts

Address: 0x4000a814

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | |

RSVD RSVD

overflow_sts dbl_rdy_sts spl_rdy_sts rpt_rdy_sts

Bits	Name	Type	Reset	Description
31:4	RSVD			
3	overflow_sts	r	0	ACC1 or ACC2 overflow
2	dbl_rdy_sts	r	0	ACC2 double error
1	spl_rdy_sts	r	0	Event being generated for every new sample value written to the SAMPLE register

Bits	Name	Type	Reset	Description
0	rpt_rdy_sts	r	0	Non-null report ready

14.4.6 qdec0_int_clr

Address: 0x4000a818

RSVD																
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	

RSVD RSVD

overflow_clr dbl_rdy_clr spl_rdy_clr rpt_rdy_clr

Bits	Name	Type	Reset	Description
31:4	RSVD			
3	overflow_clr	w1c	0	
2	dbl_rdy_clr	w1c	0	
1	spl_rdy_clr	w1c	0	
0	rpt_rdy_clr	w1c	0	

14.4.7 qdec1_ctrl0

Address: 0x4000a840

RSVD	RSVD	RSVD	RSVD	rpt_period											
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

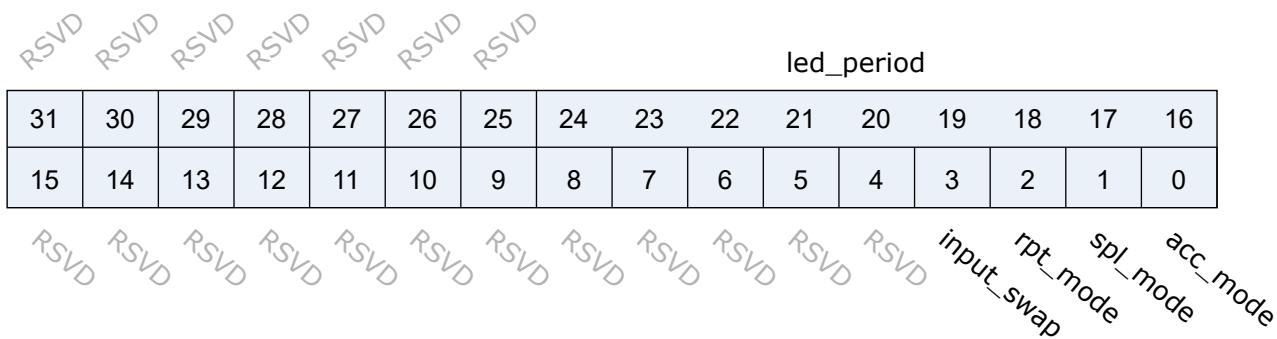
rpt_period rpt_period rpt_period deg_cnt deg_en led_pol led_en qdec_en



Bits	Name	Type	Reset	Description
31:28	RSVD			
27:12	rpt_period	r/w	16'd10	"RPT" report period in [samples/report]. Specifies the number of samples to be accumulated in the ACC1 register before the RPT_RDY and DBL_RDY events can be generated "RPT_US" report period in [us/report] = SP * RP
11:8	spl_period	r/w	4'h2	"SPL" sample period in [us/sample]. The SAMPLE register will be updated for every new sample 0: 32 us 1: 64 2: 128 3: 256 4: 512 5: 1 ms 6: 2 7: 4 8: 8 9: 16 A: 32 B: 65 C: 131
7:4	deg_cnt	r/w	0	
3	deg_en	r/w	0	deglitch
2	led_pol	r/w	1	led polarity
1	led_en	r/w	0	
0	qdec_en	r/w	0	

14.4.8 qdec1_ctrl1

Address: 0x4000a844



Bits	Name	Type	Reset	Description
31:25	RSVD			
24:16	led_period	r/w	0	Period in us the LED is switched on prior to sampling
15:4	RSVD			
3	input_swap	r/w	0	input a/b swap
2	rpt_mode	r/w	0	rpt option 0: Count time only if sample change 1: Continue time
1	spl_mode	r/w	0	spl option 0: Stop sample if rpt_rdy 1: Continue sample
0	acc_mode	r/w	1	acc option 0: Stop accumulate if overflow 1: Continue accumulate

14.4.9 qdec1_value

Address: 0x4000a848

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
acc1_val															

Bits	Name	Type	Reset	Description
31:30	RSVD			
29:28	spl_val	r	0	Sample value. Direction of last change 00: no change 01: clockwise 11: counter-clockwise 10: Error
27:20	RSVD			
19:16	acc2_val	r	0	Double error accumulation (0 15)
15:0	acc1_val	r	0	Sample accumulation (-1024 1023)

14.4.10 qdec1_int_en

Address: 0x4000a850

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | |

RSVD RSVD

overflow_en dbl_rdy_en spl_rdy_en rpt_rdy_en

Bits	Name	Type	Reset	Description
31:4	RSVD			
3	overflow_en	r/w	0	
2	dbl_rdy_en	r/w	0	
1	spl_rdy_en	r/w	0	
0	rpt_rdy_en	r/w	1	

14.4.11 qdec1_int_sts

Address: 0x4000a854

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | |

RSVD RSVD

overflow_sts dbl_rdy_sts spl_rdy_sts rpt_rdy_sts

Bits	Name	Type	Reset	Description
31:4	RSVD			
3	overflow_sts	r	0	ACC1 or ACC2 overflow
2	dbl_rdy_sts	r	0	ACC2 double error
1	spl_rdy_sts	r	0	Event being generated for every new sample value written to the SAMPLE register

Bits	Name	Type	Reset	Description
0	rpt_rdy_sts	r	0	Non-null report ready

14.4.12 qdec1_int_clr

Address: 0x4000a858

RSVD																
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	

RSVD RSVD

overflow_clr dbl_rdy_clr spl_rdy_clr rpt_rdy_clr

Bits	Name	Type	Reset	Description
31:4	RSVD			
3	overflow_clr	w1c	0	
2	dbl_rdy_clr	w1c	0	
1	spl_rdy_clr	w1c	0	
0	rpt_rdy_clr	w1c	0	

14.4.13 qdec2_ctrl0

Address: 0x4000a880

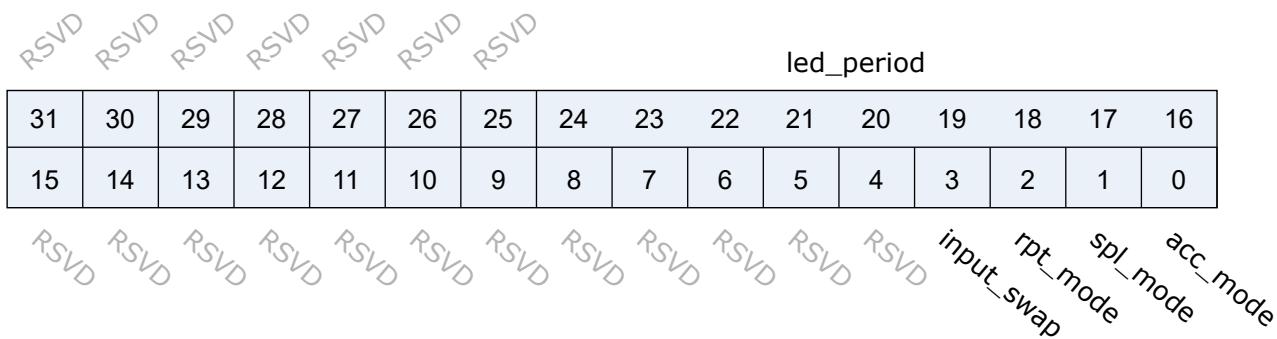
RSVD	RSVD	RSVD	RSVD	rpt_period											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

rpt_period rpt_period rpt_period deg_cnt deg_en led_en led_en qdec_en

Bits	Name	Type	Reset	Description
31:28	RSVD			
27:12	rpt_period	r/w	16'd10	"RPT" report period in [samples/report]. Specifies the number of samples to be accumulated in the ACC1 register before the RPT_RDY and DBL_RDY events can be generated "RPT_US" report period in [us/report] = SP * RP
11:8	spl_period	r/w	4'h2	"SPL" sample period in [us/sample]. The SAMPLE register will be updated for every new sample 0: 32 us 1: 64 2: 128 3: 256 4: 512 5: 1 ms 6: 2 7: 4 8: 8 9: 16 A: 32 B: 65 C: 131
7:4	deg_cnt	r/w	0	
3	deg_en	r/w	0	deglitch
2	led_pol	r/w	1	led polarity
1	led_en	r/w	0	
0	qdec_en	r/w	0	

14.4.14 qdec2_ctrl1

Address: 0x4000a884



Bits	Name	Type	Reset	Description
31:25	RSVD			
24:16	led_period	r/w	0	Period in us the LED is switched on prior to sampling
15:4	RSVD			
3	input_swap	r/w	0	input a/b swap
2	rpt_mode	r/w	0	rpt option 0: Count time only if sample change 1: Continue time
1	spl_mode	r/w	0	spl option 0: Stop sample if rpt_rdy 1: Continue sample
0	acc_mode	r/w	1	acc option 0: Stop accumulate if overflow 1: Continue accumulate

14.4.15 qdec2_value

Address: 0x4000a888

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
acc1_val															

Bits	Name	Type	Reset	Description
31:30	RSVD			
29:28	spl_val	r	0	Sample value. Direction of last change 00: no change 01: clockwise 11: counter-clockwise 10: Error
27:20	RSVD			
19:16	acc2_val	r	0	Double error accumulation (0 15)
15:0	acc1_val	r	0	Sample accumulation (-1024 1023)

14.4.16 qdec2_int_en

Address: 0x4000a890

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | |

RSVD RSVD

overflow_en dbl_rdy_en spl_rdy_en rpt_rdy_en

Bits	Name	Type	Reset	Description
31:4	RSVD			
3	overflow_en	r/w	0	
2	dbl_rdy_en	r/w	0	
1	spl_rdy_en	r/w	0	
0	rpt_rdy_en	r/w	1	

14.4.17 qdec2_int_sts

Address: 0x4000a894

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | |

RSVD RSVD

overflow_sts dbl_rdy_sts spl_rdy_sts rpt_rdy_sts

Bits	Name	Type	Reset	Description
31:4	RSVD			
3	overflow_sts	r	0	ACC1 or ACC2 overflow
2	dbl_rdy_sts	r	0	ACC2 double error
1	spl_rdy_sts	r	0	Event being generated for every new sample value written to the SAMPLE register

Bits	Name	Type	Reset	Description
0	rpt_rdy_sts	r	0	Non-null report ready

14.4.18 qdec2_int_clr

Address: 0x4000a898

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | | | |

RSVD RSVD

overflow_clr *dbl_rdy_clr* *spl_rdy_clr* *rpt_rdy_clr*

Bits	Name	Type	Reset	Description
31:4	RSVD			
3	overflow_clr	w1c	0	
2	dbl_rdy_clr	w1c	0	
1	spl_rdy_clr	w1c	0	
0	rpt_rdy_clr	w1c	0	

15.1 KYS introduction

The KYS (Key Scan) module is used to scan the matrix keyboard to obtain key values. It periodically scans the keyboard connection pins and generates an interrupt as soon as it finds a key press.

15.2 KYS main features

- Configurable number of rows and columns, up to 8 rows*20 columns of matrix keyboard
- Store up to four key values
- Support key interrupt

15.3 KYS function description

15.3.1 Configurable number of rows and columns

The number of rows and columns of the matrix keyboard can be configured through the bits <ROW_NUM> and <COL_NUM> of the register KS_CTRL, and the configuration value is the actual value minus one. The maximum number of rows supports 8 rows, and the maximum number of columns supports 20 columns.

15.3.2 GPIO selection

Since the button scan is fixed from the GPIO pins with the functions of ROW_0 and COL_0, the row pins need to be selected sequentially from the GPIO with the function of ROW_0, that is, from any one of GPIO0/GPIO8/GPIO16/GPIO24. The column pins need to be selected in order from the GPIO with the function of COL_0, that is, from any one of GPIO0/GPIO20.

15.3.3 Key value

The key value will be stored in the register KEYCODE_VALUE, every 8 bits is a key value, and the first key value will be stored in the lowest 8 bits. When the bit corresponding to the serial number in the register KEYCODE_CLR is set to one, the corresponding key value and interrupt flag bit will be cleared. The row number corresponding to the key value = key value% total number of rows, the column number corresponding to the key value = key value / total number of rows.

15.3.4 Interrupt

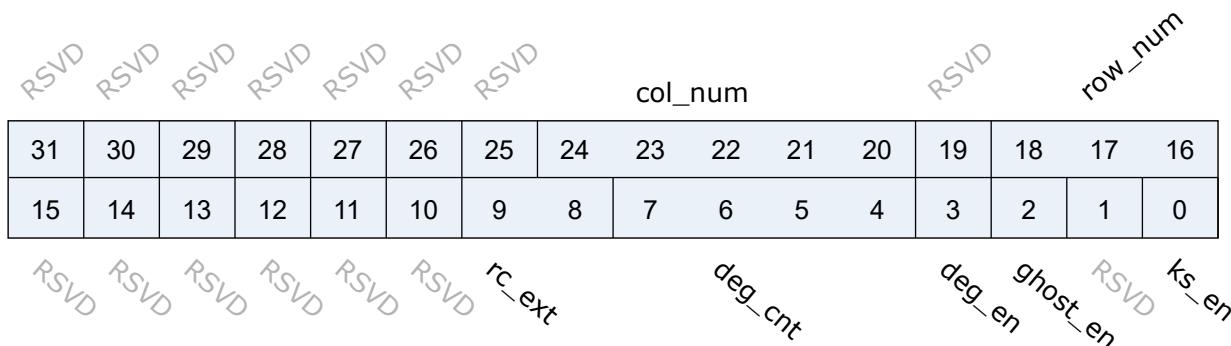
When a key press is detected, the interrupt flag bit will be set and an interrupt will be generated at the same time.

15.4 Register description

Name	Description
ks_ctrl	Keyscan control
ks_int_en	Keyscan interrupt enable
ks_int_sts	Keyscan interrupt status
keycode_clr	Keycode clear
keycode_value	Keycode value

15.4.1 ks_ctrl

Address: 0x4000a900

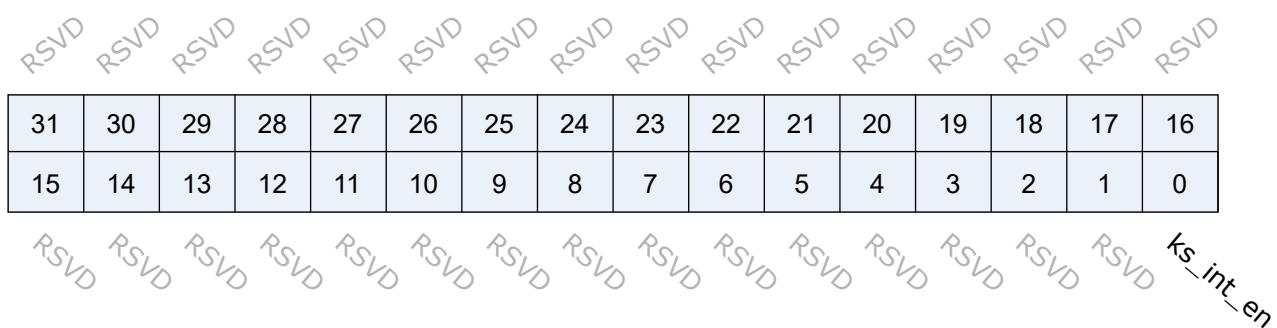


Bits	Name	Type	Reset	Description
31:25	RSVD			
24:20	col_num	r/w	5'd19	col_num + 1

Bits	Name	Type	Reset	Description
19	RSVD			
18:16	row_num	r/w	3'd7	row_num + 1
15:10	RSVD			
9:8	rc_ext	r/w	2'd3	idle duration between column scans
7:4	deg_cnt	r/w	0	
3	deg_en	r/w	0	deglitch
2	ghost_en	r/w	0	ghost key event detection
1	RSVD			
0	ks_en	r/w	0	

15.4.2 ks_int_en

Address: 0x4000a910



Bits	Name	Type	Reset	Description
31:1	RSVD			
0	ks_int_en	r/w	1	

15.4.3 ks_int_sts

Address: 0x4000a914

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | |

RSVD keycode_valid

Bits	Name	Type	Reset	Description
31:4	RSVD			
3:0	keycode_valid	r	0	

15.4.4 keycode_clr

Address: 0x4000a918

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | |

RSVD keycode_clr

Bits	Name	Type	Reset	Description
31:4	RSVD			
3:0	keycode_clr	w1c	0	

15.4.5 keycode_value

Address: 0x4000a91c

keycode3								keycode2							
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							
keycode1								keycode0							

Bits	Name	Type	Reset	Description
31:24	keycode3	r	8'hff	
23:16	keycode2	r	8'hff	
15:8	keycode1	r	8'hff	
7:0	keycode0	r	8'hff	Col = keycode / (row_num+1) Row = keycode

16

I2S

16.1 I2S introduction

I2S is an interface standard for the transmission of digital audio data, and two groups (left and right channel) data are transmitted in a sequential manner.

I2S separates the clock signal and the data signal for transmission, so that the receiving end does not need to restore the clock from the data signal, thereby reducing the design difficulty of the receiving end.

16.2 I2S main features

- Support master mode and slave mode
- Support Left-justified/Right-justified/DSP and other data formats
- Support 8/16/24/32 bit data width
- In addition to mono/dual-channel mode, four-channel mode is also supported
- Support dynamic mute switching function
- The width of the data transmission FIFO is 32 bits and the depth is 16
- The width of the data receiving FIFO is 32 bits and the depth is 16

16.3 I2S function description

Table 16.1: Pin lists

Name	Type	Description
I2Sx_DI	input	Serial data input
I2Sx_DO	output	Serial data output

Table 16.1: Pin lists(continued)

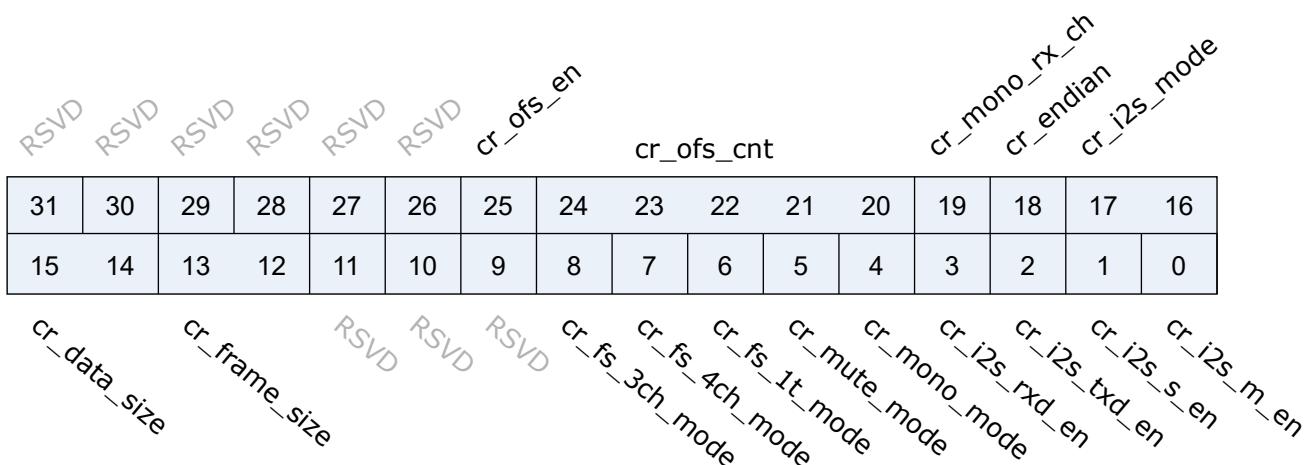
Name	Type	Description
I2Sx_BCLK	input/output	Synchronous transmission clock, output when used as a master, and input when used as a slave
I2Sx_FS	input/output	Data start/end signal, output when used as master, input when used as slave

16.4 Register description

Name	Description
i2s_config	I2S configuration
i2s_int_sts	I2S interrupt status
i2s_bclk_config	I2S clock configuration
i2s_fifo_config_0	I2S FIFO configuration0
i2s_fifo_config_1	I2S DMA FIFO configuration
i2s_fifo_wdata	I2S FIFO write data
i2s_fifo_rdata	I2S FIFO read data
i2s_io_config	I2S IO configuration

16.4.1 i2s_config

Address: 0x4000aa00

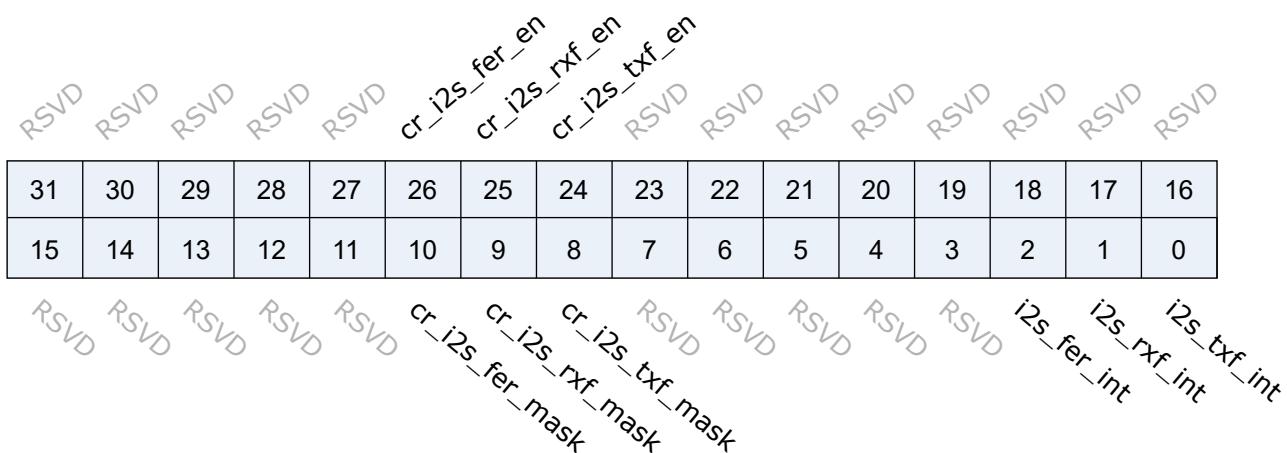


Bits	Name	Type	Reset	Description
31:26	RSVD			
25	cr_ofs_en	r/w	1'b0	Offset enable 1'b0: Disabled, 1'b1: Enabled
24:20	cr_ofs_cnt	r/w	5'd0	Offset cycle count (unit: cycle of I2S BCLK) 5'd0: 1 cycle 5'd1: 2 cycles ...
19	cr_mono_rx_ch	r/w	1'b0	RX mono mode channel select signal 1'b0: L-channel 1'b1: R-channel
18	cr_endian	r/w	1'b0	Data endian (bit reverse) 1'b0: MSB goes out first, 1'b1: LSB goes out first
17:16	cr_i2s_mode	r/w	2'd0	2'd0: Left-Justified, 2'd1: Right-Justified, 2'd2: DSP, 2'd3: Reserved
15:14	cr_data_size	r/w	2'd1	Data bit width of each channel 2'd0: 8, 2'd1: 16, 2'd2: 24, 2'd3: 32 (bits)
13:12	cr_frame_size	r/w	2'd1	Frame size of each channel 2'd0: 8, 2'd1: 16, 2'd2: 24, 2'd3: 32 (cycles)
11:9	RSVD			
8	cr_fs_3ch_mode	r/w	1'b0	1'b0: FS 2-channel mode, 1'b1: FS 3-channel mode (DSP mode only) Note: cr_fs_3ch_mode & cr_fs_4ch_mode should NOT be enabled at the same time Note: cr_mono_mode & cr_fifo_lr_merge will be invalid in 3-channel mode Note: When 3-channel mode is enabled, frame_size must equal data_size
7	cr_fs_4ch_mode	r/w	1'b0	1'b0: FS 2-channel mode, 1'b1: FS 4-channel mode (DSP mode only) Note: cr_fs_3ch_mode & cr_fs_4ch_mode should NOT be enabled at the same time Note: When 4-channel mode is enabled, frame_size must equal data_size
6	cr_fs_1t_mode	r/w	1'b0	1'b0: FS high/low is even, 1'b1: FS only asserts for 1 cycle
5	cr_mute_mode	r/w	1'b0	1'b0: Normal mode, 1'b1: Mute mode
4	cr_mono_mode	r/w	1'b0	1'b0: Stereo mode, 1'b1: Mono mode Note: csr_mono_mode & csr_fifo_lr_merge should NOT be enabled at the same time

Bits	Name	Type	Reset	Description
3	cr_i2s_rxd_en	r/w	1'b0	Enable signal of I2S RXD signal
2	cr_i2s_txd_en	r/w	1'b0	Enable signal of I2S TXD signal
1	cr_i2s_s_en	r/w	1'b0	Enable signal of I2S Slave function, cannot enable both csr_i2s_m_en & csr_i2s_s_en
0	cr_i2s_m_en	r/w	1'b0	Enable signal of I2S Master function, cannot enable both csr_i2s_m_en & csr_i2s_s_en

16.4.2 i2s_int_sts

Address: 0x4000aa04



Bits	Name	Type	Reset	Description
31:27	RSVD			
26	cr_i2s_fer_en	r/w	1'b1	Interrupt enable of i2s_fer_int
25	cr_i2s_rxf_en	r/w	1'b1	Interrupt enable of i2s_rxf_int
24	cr_i2s_txf_en	r/w	1'b1	Interrupt enable of i2s_txf_int
23:11	RSVD			
10	cr_i2s_fer_mask	r/w	1'b1	Interrupt mask of i2s_fer_int
9	cr_i2s_rxf_mask	r/w	1'b1	Interrupt mask of i2s_rxf_int
8	cr_i2s_txf_mask	r/w	1'b1	Interrupt mask of i2s_txf_int
7:3	RSVD			
2	i2s_fer_int	r	1'b0	I2S TX/RX FIFO error interrupt, auto-cleared when FIFO overflow/underflow error flag is cleared

Bits	Name	Type	Reset	Description
1	i2s_rxf_int	r	1'b0	I2S RX FIFO ready (rx_fifo_cnt > rx_fifo_th) interrupt, auto-cleared when data is popped
0	i2s_txf_int	r	1'b0	I2S TX FIFO ready (tx_fifo_cnt > tx_fifo_th) interrupt, auto-cleared when data is pushed

16.4.3 i2s_bclk_config

Address: 0x4000aa10

cr_bclk_div_h															
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
cr_bclk_div_l															

Bits	Name	Type	Reset	Description
31:28	RSVD			
27:16	cr_bclk_div_h	r/w	12'd1	I2S BCLK active high period (unit: cycle of i2s_clk)
15:12	RSVD			
11:0	cr_bclk_div_l	r/w	12'd1	I2S BCLK active low period (unit: cycle of i2s_clk)

16.4.4 i2s_fifo_config_0

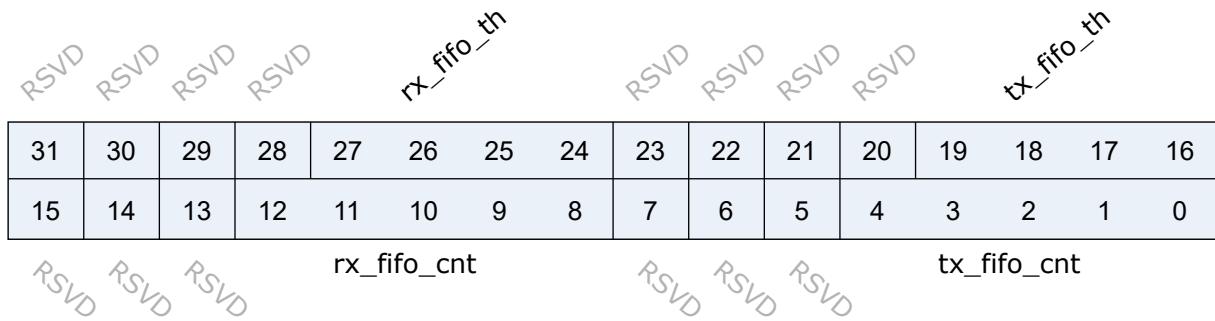
Address: 0x4000aa80

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
RSVD															

Bits	Name	Type	Reset	Description
31:11	RSVD			
10	cr_fifo_24b_lj	r/w	1'b0	FIFO 24-bit data left-justified mode 1'b0: Right-justified, 8'h0, data[23:0] 1'b1: Left-justified, data[23:0], 8'h0 Note: Valid only when cr_data_size = 2'd2 (24-bit)
9	cr_fifo_lr_exchg	r/w	1'b0	The position of L/R channel data within each entry is exchanged if this bit is enabled Can only be enabled if data size is 8 or 16 bits and csr_fifo_lr_merge is enabled
8	cr_fifo_lr_merge	r/w	1'b0	Each FIFO entry contains both L/R channel data if this bit is enabled Can only be enabled if data size is 8 or 16 bits Note: cr_fifo_lr_merge & cr_mono_mode should NOT be enabled at the same time Note: cr_fifo_lr_merge & cr_fifo_l_shift should NOT be enabled at the same time
7	rx_fifo_underflow	r	1'b0	Underflow flag of RX FIFO, can be cleared by rx_fifo_clr
6	rx_fifo_overflow	r	1'b0	Overflow flag of RX FIFO, can be cleared by rx_fifo_clr
5	tx_fifo_underflow	r	1'b0	Underflow flag of TX FIFO, can be cleared by tx_fifo_clr
4	tx_fifo_overflow	r	1'b0	Overflow flag of TX FIFO, can be cleared by tx_fifo_clr
3	rx_fifo_clr	w1c	1'b0	Clear signal of RX FIFO
2	tx_fifo_clr	w1c	1'b0	Clear signal of TX FIFO
1	i2s_dma_rx_en	r/w	1'b0	Enable signal of dma_rx_req/ack interface
0	i2s_dma_tx_en	r/w	1'b0	Enable signal of dma_tx_req/ack interface

16.4.5 i2s_fifo_config_1

Address: 0x4000aa84



Bits	Name	Type	Reset	Description
31:28	RSVD			
27:24	rx_fifo_th	r/w	4'd0	RX FIFO threshold, dma_rx_req will not be asserted if tx_fifo_cnt is less than this value
23:20	RSVD			
19:16	tx_fifo_th	r/w	4'd0	TX FIFO threshold, dma_tx_req will not be asserted if tx_fifo_cnt is less than this value
15:13	RSVD			
12:8	rx_fifo_cnt	r	5'd0	RX FIFO available count
7:5	RSVD			
4:0	tx_fifo_cnt	r	5'd16	TX FIFO available count

16.4.6 i2s_fifo_wdata

Address: 0x4000aa88

i2s_fifo_wdata

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

i2s_fifo_wdata

Bits	Name	Type	Reset	Description
31:0	i2s_fifo_wdata	w	x	

16.4.7 i2s_fifo_rdata

Address: 0x4000aa8c

i2s_fifo_rdata

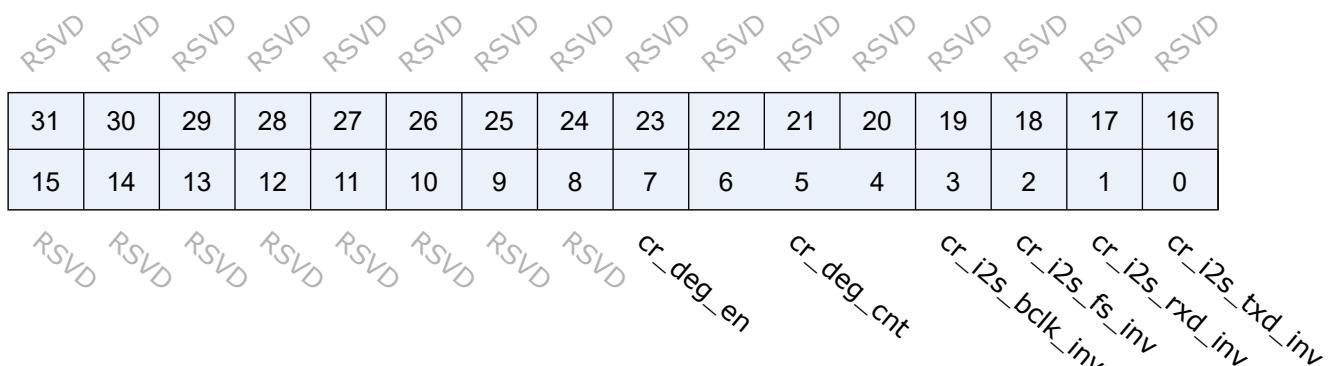
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

i2s_fifo_rdata

Bits	Name	Type	Reset	Description
31:0	i2s_fifo_rdata	r	32'h0	

16.4.8 i2s_io_config

Address: 0x4000aafc



Bits	Name	Type	Reset	Description
31:8	RSVD			
7	cr_deg_en	r/w	1'b0	Deglitch enable (for all th input pins) 1'b0: Disabled, 1'b1: Enabled
6:4	cr_deg_cnt	r/w	3'd0	Deglitch cycle count (unit: cycle of I2S kernel clock) 3'd0: 1 cycle 3'd1: 2 cycles ...
3	cr_i2s_bclk_inv	r/w	1'b0	Inverse BCLK signal 0: No inverse, 1: Inverse
2	cr_i2s_fs_inv	r/w	1'b0	Inverse FS signal 0: No inverse, 1: Inverse
1	cr_i2s_rxd_inv	r/w	1'b0	Inverse RXD signal 0: No inverse, 1: Inverse
0	cr_i2s_txd_inv	r/w	1'b0	Inverse TXD signal 0: No inverse, 1: Inverse

17.1 Emac introduction

The EMAC module is a 10/100Mbps Ethernet MAC (Ethernet Media Access Controller) compatible with IEEE 802.3. It includes status and control register group, transceiver module, transceiver buffer descriptor group, host interface, MDIO, physical layer chip (PHY) interface.

The status and control register group contains the status bits and control bits of the EMAC, which is the interface with the user program, and is responsible for controlling data receiving and sending and reporting the status.

The transceiver module is responsible for obtaining the data frame from the designated memory location according to the control word in the transceiver descriptor, adding the preamble, CRC, and expanding the short frame before sending it out through the PHY; Or receive data from the PHY, and put the data into the designated memory according to the transmit and receive buffer descriptor. Configure related event flags after sending and receiving. If the event interrupt is enabled, the interrupt request will be sent to the host for processing.

The MDIO and MII/RMII interfaces are responsible for communicating with the PHY, including reading and writing PHY registers and sending and receiving data packets.

17.2 Emac main features

- Compatible with the MAC layer functions defined by IEEE 802.3
- Support MII and RMII interface PHY defined by IEEE 802.3
- Interaction with PHY through MDIO
- Support 10Mbps and 100Mbps Ethernet
- Support half duplex and full duplex
- In full duplex mode, support automatic flow control and control frame generation
- Support collision detection and retransmission in half-duplex mode

- Support CRC generation and verification
- Data frame preamble generation and removal
- When sending, automatically expand short data frames
- Detect too long or too short data frame (length limit)
- Can transmit long data frames (> standard Ethernet frame length)
- Automatically discard packets that exceed the number of retransmissions or the frame gap is too small
- Broadcast packet filtering
- Internal RAM for storing up to 128 BD (Buffer Descriptor)
- Various event flags sent/received
- Generate a corresponding interrupt when an event occurs

17.3 Emac function description

The composition of the EMAC module is shown in the figure below.

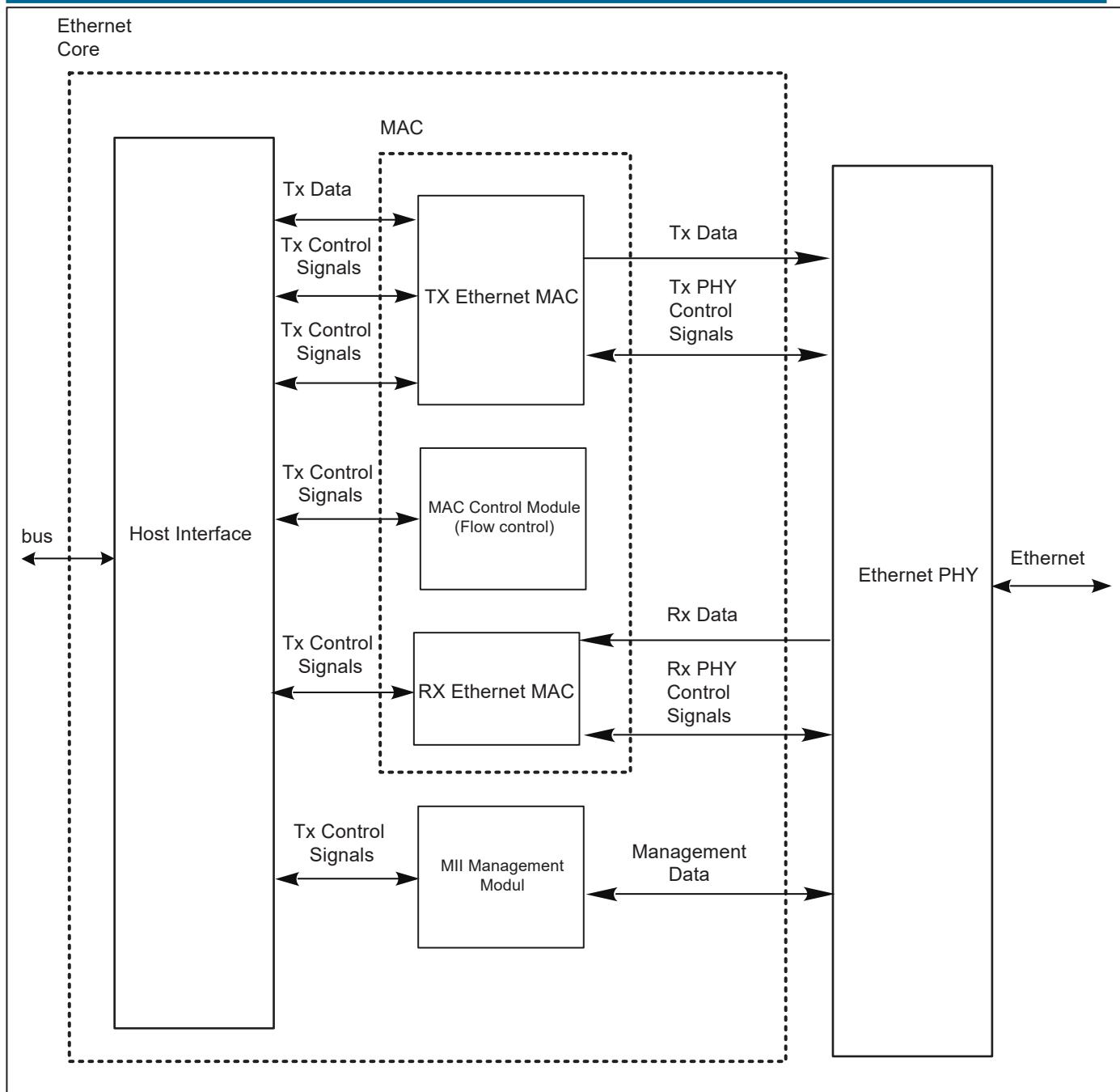


Fig. 17.1: EMAC architecture

The module's control register can read and write the PHY register through MDIO to realize configuration, select mode (half/full duplex), initiate negotiation and other operations.

The receiving module filters and checks the received data frame: whether there is a legal preamble, FCS, length, etc. And according to the descriptor, the data frame is stored in the designated buffer address.

The sending module obtains data from the memory according to the data buffer descriptor, adds preamble, FCS, pad, etc., and then sends the data according to the CSMA/CD protocol.

If CRS is detected, the retry will be delayed.

The send and receive buffer descriptor group is connected to the external RAM, which is used to store the Ethernet data frames sent and received. Each descriptor contains the corresponding control status word and the corresponding buffer memory address. There are 128 groups of descriptors, which can be flexibly allocated for sending or receiving.

17.4 Emac clock

The EMAC module needs a clock for synchronous transmission and reception (at 100Mbps, 25MHz (MII) or 50MHz (RMII); at 10Mbps, 2.5MHz). This clock must be synchronized between EMAC and PHY.

17.5 Send and receive buffer descriptor (BD, Buffer Descriptor)

The transceiver buffer descriptor is used to provide the association between the EAMC and the data frame buffer address information, to control the transceiver data frame, and to provide the transceiver status prompt.

Each descriptor is composed of two consecutive words (32 bits). The low address word provides the length, control and status bits of the data frame contained in the buffer; the high address word is a memory pointer.

Specific BD description can refer to the register description chapter.

Note that: For BD, you need to write in word.

The EMAC module supports 128 BDs, which are shared by the sending/receiving logic and can be freely combined. But the sending BD always occupies the previous continuous area (the number is specified by the TXBDNUM field in the MAC_TX_BD_NUM register).

EMAC processes the sending/receiving BD in the order of BD, until it encounters the BD marked as WR, it wraps around to send/receive the respective first BD.

17.6 PHY interaction

The PHY interaction register group provides the commands and data communication methods needed for PHY interaction. EMAC controls the working mode of PHY through MDIO and ensures that the two match (rate, full/half duplex).

The data packet interacts between EMAC and PHY through the MII/RMII interface, and can be selected by RMII_EN in the EMAC mode register (EMAC_MODE). When this bit is 1, select RMII mode, otherwise it is MII mode.

Both MII and RMII modes support the 10Mbps and 100Mbps transmission rates specified in the IEEE 802.3u standard.

The transmission signal description of MII and RMII is shown in the table below.

Table 17.1: Transmission signal

Name	MII	RMII
EXTCK_EREFCK	ETXCK: send clock signal	EREFCK: reference clock
ECRS	ECRS: carrier detection	-
ECOL	ECOL: collision detection	-
ERXDV	ERXDV: data valid	ECRSDV: Carrier detect/data valid
ERX0-ERX3	ERX0-ERX3: 4-bit receive data	ERX0-ERX1: 2-bit receive data
ERXER	ERXER: Receive error indication	ERXER: Receive error indication
ERXCK	ERXCK: Receive clock signal	-
ETXEN	ETXEN: transmit enable	ETXEN: transmit enable
ETX0-ETX3	ETX0-ETX3: 4-bit transmit data	ETX0-ETX1: 2-bit transmit data
ETXER	ETXER: Send error indication	-
EMDC	MDIO Clock	MDIO Clock
EMDIO	MDIO Data Input Output	MDIO Data Input Output

The RMII interface has fewer pins, and a 2-bit data line is used for receiving and sending. At a rate of 100Mbps, a 50MHz reference clock is required.

17.7 Programming process

17.7.1 PHY initialization

- According to the PHY type, set the RMII_EN bit in the EMAC_MODE register to select the appropriate connection method
- Set the MAC address of EMAC to EMAC_MAC_ADDR0 and EMAC_MAC_ADDR1
- Set the appropriate clock for the MDIO part by programming the field CLKDIV in the EMAC_MIIMODE register
- Set the corresponding PHY address to the FIAD field of the register EMAC_MIIADDRESS
- According to the PHY manual, send commands through the EMAC_MIICOMMAND and EMAC_MIITX_DATA registers
- The data read from the PHY will be stored in the EMAC_MIIRX_DATA register
- The status of interaction with PHY commands can be queried through the EMAC_MIISTATUS register

After the basic interaction is completed, the PHY should enter the auto-negotiation state. After the negotiation is completed, program the mode to the FULLD bit in the EMAC_MODE register according to the negotiation result.

17.7.2 Send data frame

- Configure bit fields such as data frame format and interval in the EMAC_MODE register
- By configuring the TXBDNUM field in the EMAC_TX_BD_NUM register to specify the number of BDs used for transmission, the remaining BDs are RX BDs
- Prepare the data frame to be sent in the memory
- Fill in the address of the data frame into the data pointer field (word 1) corresponding to the sent BD
- Clear the status flag in the control and status field (word 0) corresponding to the sent BD, and set the control field (CRC enable, PAD enable, interrupt enable, etc.)
- Write the length of the data frame and set the RD field to inform EMAC that this BD data needs to be sent; if necessary, set the IRQ bit to enable interrupts
- In particular, if it is the last BD sent, the WR bit needs to be set, and EMAC will “wrap around” to the first sent BD for processing after processing this BD
- If there are multiple BDs to be sent, repeat the steps of setting BD to fill all sending BDs
- If you need to enable the transmit interrupt, you also need to configure the TX related bits in the EMAC_INT_MASK register
- Configure the TXEN bit in the EMAC_MODE register to enable transmission
- If the interrupt is enabled, in the interrupt sent, the current BD can be obtained through the TXBDNUM field in the EMAC_TX_BD_NUM register
- Perform corresponding processing according to the current BD status word
- For the BD whose data has been sent, the RD bit in the control field will be cleared by hardware and will not be sent again; after filling in new data, set RD, and this BD can be used for sending again

17.7.3 Receive data frame

- Configure bit fields such as data frame format and interval in the EMAC_MODE register
- By configuring the TXBDNUM field in the EMAC_TX_BD_NUM register to specify the number of BDs used for transmission, the remaining BDs are RX BDs
- The area in the memory that is ready to receive data
- Fill in the address of the data frame into the data pointer field (word 1) corresponding to the received BD
- Clear the status flag in the control and status field (word 0) corresponding to the sending BD, and set the control field (interrupt enable, etc.)
- Write the receivable data frame length and set the E bit field to inform EMAC that the BD is idle and can be used for data reception; if necessary, set the IRQ bit to enable interrupts

- In particular, if it is the last valid receiving BD, the WR bit needs to be set, and EMAC will “wrap around” to the first receiving BD for processing after processing this BD
- If there are multiple BDs available to receive data, repeat the steps of setting BD to fill all BDs
- If you need to enable the receive interrupt, you also need to configure the RX related bits in the EMAC_INT_MASK register
- Configure the RXEN bit in the EMAC_MODE register to enable reception
- If the interrupt is enabled, in the received interrupt, the current BD can be obtained through the RXBDNUM field in the EMAC_TX_BD_NUM register
- Perform corresponding processing according to the current BD status word
- For the received BD, the E bit in the control field will be cleared by hardware and will not be used for receiving again; the data needs to be taken away, and E is set, this BD can be used for receiving again

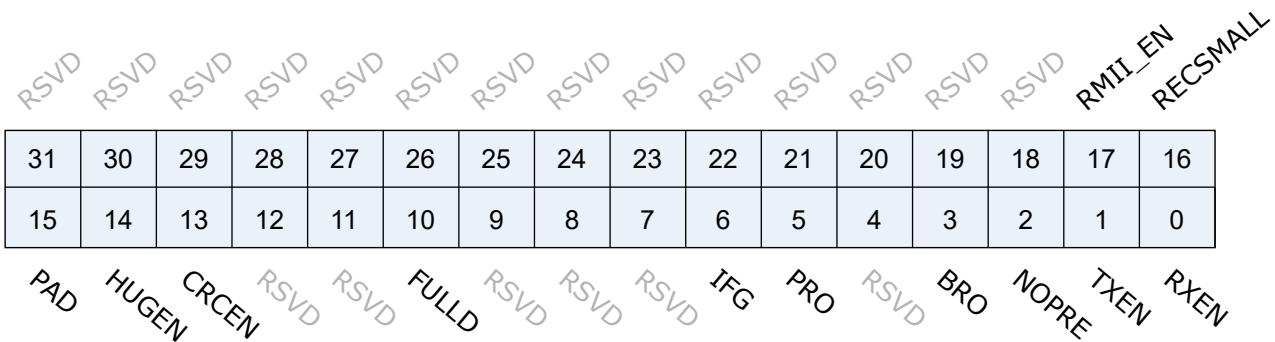
17.8 Register description

Name	Description
MODE	EMAC configuration
INT_SOURCE	EMAC transmit control
INT_MASK	EMAC interrupt mask
IPGT	Inter packet gap
PACKETLEN	Frame length
COLLCONFIG	Collision configuration
TX_BD_NUM	TX buffer descriptors number
MIIMODE	Management Data configuration
MIICOMMAND	Trigger command
MIIADDRESS	Register address
MIITX_DATA	Control data to be written to PHY
MIIRX_DATA	Received data from PHY
MIISTATUS	MIIM I/F status
MAC_ADDR0	Ethernet MAC address0
MAC_ADDR1	Ethernet MAC address1
HASH0_ADDR	Lower 32-bit of HASH register

Name	Description
HASH1_ADDR	Upper 32-bit of HASH register
TXCTRL	TX control

17.8.1 MODE

Address: 0x4000d000



Bits	Name	Type	Reset	Description
31:18	RSVD			
17	RMII_EN	r/w	1'b0	RMII mode enable 0: MII PHY I/F is used 1: RMII PHY I/F is used
16	RECSMALL	r/w	1'b0	Receive small frame enable 0: Frames smaller than MINFL are ignored. 1: Frames smaller than MINFL are accepted.
15	PAD	r/w	1'b1	Padding enable 0: Do not add pads to frames shorter than MINFL. 1: Add pads to short frames, until the length equals MINFL.
14	HUGEN	r/w	1'b0	Huge frames enable 0: The maximum frame length is MAXFL. All additional bytes are dropped. 1: Frame size is not limited by MAXFL and can be up to 64K bytes.
13	CRCEN	r/w	1'b1	CRC Enable 0: TX MAC does not append CRC field. 1: TX MAC will append CRC field to every frame.
12:11	RSVD			

Bits	Name	Type	Reset	Description
10	FULLD	r/w	1'b0	Full duplex 0: Half duplex mode. 1: Full duplex mode.
9:7	RSVD			
6	IFG	r/w	1'b0	Inter frame gap check 0: IFG is verified before each frame be received. 1: All frames are received regardless to IFG requirement.
5	PRO	r/w	1'b0	Promiscuous mode enable 0: The destination address is checked before receiving. 1: All frames received regardless of the address.
4	RSVD			
3	BRO	r/w	1'b1	Broadcast address enable 0: Reject all frames containing the broadcast address unless the PRO bit is asserted. 1: Receive all frames containing broadcast address.
2	NOPRE	r/w	1'b0	No preamble mode 0: 7-byte preamble will be sent. 1: No preamble will be sent.
1	TXEN	r/w	1'b0	Transmit enable 0: Transmitter is disabled. 1: Transmitter is enabled. If TX_BD_NUM equals 0x0 (zero buffer descriptors are used), then the transmitter is disabled regardless of TXEN.
0	RXEN	r/w	1'b0	Receiver enable 0: Receiver is disabled. 1: Receiver is enabled. If TX_BD_NUM equals 0x80 (all buffer descriptors are used for TX), then the receiver is disabled regardless of RXEN.

17.8.2 INT_SOURCE

Address: 0x4000d004

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | | | |

RSVD RSVD RSVD RSVD RSVD RSVD RSVD RSVD RXC TXC BUSY RXE RXB TXF TXB

Bits	Name	Type	Reset	Description
31:7	RSVD			
6	RXC	r/w	1'b0	<p>Receive control frame</p> <p>This bit indicates that the control frame was received. It is cleared by writing 1 to it.</p> <p>Bit RXFLOW in the CTRLMODE register must be set to 1 in order to get the RXC bit set.</p>
5	TXC	r/w	1'b0	<p>Transmit control frame</p> <p>This bit indicates that a control frame was transmitted. It is cleared by writing 1 to it.</p> <p>Bit TXFLOW in the CTRLMODE register must be set to 1 in order to get the TXC bit set.</p>
4	BUSY	r/w	1'b0	<p>Busy</p> <p>This bit indicates that RX packet is being received and there is no empty buffer descriptor to use. It is cleared by writing 1 to it.</p> <p>This bit appears regardless to the IRQ bits in the Receive Buffer Descriptor.</p>
3	RXE	r/w	1'b0	<p>Receive error</p> <p>This bit indicates that an error occurred while receiving data (overrun, receiver error, dribble nibble, too long, >64K, CRC error, bus error or late collision. It is cleared by writing 1 to it.</p> <p>This bit appears only when IRQ bit is set in the Receive Buffer Descriptor.</p>
2	RXB	r/w	1'b0	<p>Receive frame</p> <p>This bit indicates that a frame was received. It is cleared by writing 1 to it.</p> <p>This bit appears only when IRQ bit is set in the Receive Buffer Descriptor.</p>

Bits	Name	Type	Reset	Description
1	TXE	r/w	1'b0	<p>Transmit error</p> <p>This bit indicates that a buffer was not transmitted due to a transmit error (underrun, retransmission limit, late collision, bus error or defer timeout). It is cleared by writing 1 to it.</p> <p>This bit appears only when IRQ bit is set in the Transmit Buffer Descriptor.</p>
0	TXB	r/w	1'b0	<p>Transmit buffer</p> <p>This bit indicates that a buffer has been transmitted. It is cleared by writing 1 to it.</p> <p>This bit appears only when IRQ bit is set in the Transmit Buffer Descriptor.</p>

17.8.3 INT_MASK

Address: 0x4000d008

RSVD																	
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		

RSVD RSVD RSVD RSVD RSVD RSVD RSVD RSVD RXC_M TXC_M BUSY_M RXE_M RXB_M TXE_M TXB_M

Bits	Name	Type	Reset	Description
31:7	RSVD			
6	RXC_M	r/w	1'b1	<p>Receive control frame mask ENABLE</p> <p>0: Interrupt is un-masked</p> <p>1: Interrupt is masked</p>
5	TXC_M	r/w	1'b1	<p>Transmit control frame mask ENABLE</p> <p>0: Interrupt is un-masked</p> <p>1: Interrupt is masked</p>
4	BUSY_M	r/w	1'b1	<p>Busy mask ENABLE</p> <p>0: Interrupt is un-masked</p> <p>1: Interrupt is masked</p>

Bits	Name	Type	Reset	Description
3	RXE_M	r/w	1'b1	Receive error mask ENABLE 0: Interrupt is un-masked 1: Interrupt is masked
2	RXB_M	r/w	1'b1	Receive frame mask ENABLE 0: Interrupt is un-masked 1: Interrupt is masked
1	TXE_M	r/w	1'b1	Transmit error mask ENABLE 0: Interrupt is un-masked 1: Interrupt is masked
0	TXB_M	r/w	1'b1	Transmit buffer mask ENABLE 0: Interrupt is un-masked 1: Interrupt is masked

17.8.4 IPGT

Address: 0x4000d00c

RSVD	IPGT																
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		

Bits	Name	Type	Reset	Description
31:7	RSVD			
6:0	IPGT	r/w	7'h18	Inter packet gap The recommended value is 0x18 (24 clock cycles), which equals 9.6 us for 10 Mbps and 0.96 us for 100 Mbps mode



17.8.5 PACKETLEN

Address: 0x4000d018

MINFL															
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

MAXFI

Bits	Name	Type	Reset	Description
31:16	MINFL	r/w	16'h40	<p>Minimum frame length</p> <p>The minimum Ethernet packet is 64 bytes long (0x40).</p> <p>To receive small packets, assert the RECSMALL bit or change the MINFL value.</p> <p>To transmit small packets, assert the PAD bit or change the MINFL value.</p>
15:0	MAXFL	r/w	16'h600	<p>Maximum frame length</p> <p>The maximum Ethernet packet is 1518 bytes long. To support this and to have some additional space for tags, a default maximum packet length equals to 1536 bytes (0x600).</p> <p>For bigger packets, you can assert the HUGEN bit or increase the value of MAXFL field.</p>

17.8.6 COLLCONFIG

Address: 0x4000d01c

RSVD	MAXRET														
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

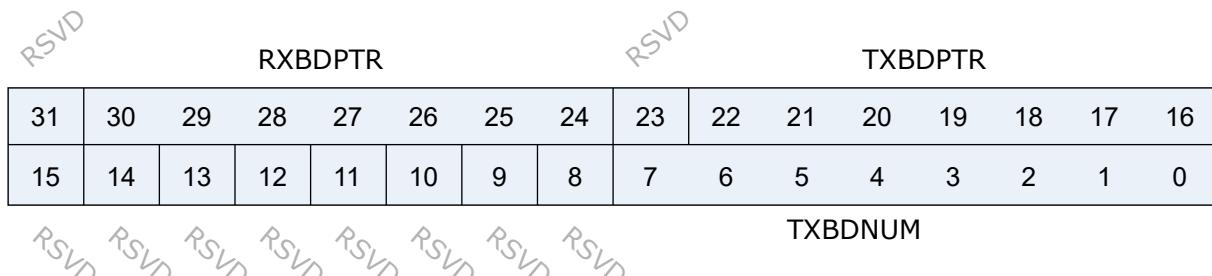
COLLVALID

Bits	Name	Type	Reset	Description
31:20	RSVD			

Bits	Name	Type	Reset	Description
19:16	MAXRET	r/w	4'hF	<p>Maximum retry</p> <p>This field specifies the maximum number of consequential retransmission attempts after the collision is detected.</p> <p>When the maximum number has been reached, the TX MAC reports an error and stops transmitting the current packet.</p> <p>According to the Ethernet standard, the MAXRET default value is set to 0xf (15).</p>
15:6	RSVD			
5:0	COLLVALID	r/w	6'h3F	<p>Collision valid</p> <p>This field specifies a collision time window. A collision that occurs later than the time window is reported as a "Late Collisions" and transmission of the current packet is aborted.</p>

17.8.7 TX_BD_NUM

Address: 0x4000d020



Bits	Name	Type	Reset	Description
31	RSVD			
30:24	RXBDPTR	r	7'h0	RX buffer descriptors (BD) pointer, pointing at the RXBD currently being used
23	RSVD			
22:16	TXBDPTR	r	7'h0	TX buffer descriptors (BD) pointer, pointing at the TXBD currently being used
15:8	RSVD			

Bits	Name	Type	Reset	Description
7:0	TXBDNUM	r/w	8'h40	TX buffer descriptors (BD) number Number of TX BD. TX and RX share 128 (0x80) descriptors, so the number of RX BD equals 0x80 - TXBDNUM. The maximum number of TXBDNUM is 0x80. Values greater than 0x80 cannot be written into this register.

17.8.8 MIIMODE

Address: 0x4000d028

RSVD																	
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		

RSVD RSVD
 MIINOPRE CLKDIV

Bits	Name	Type	Reset	Description
31:9	RSVD			
8	MIINOPRE	r/w	1'b0	No preamble for Management Data (MD) 0: 32-bit preamble will be sent. 1: No preamble will be sent.
7:0	CLKDIV	r/w	8'h64	Clock divider for Management Data Clock (MDC) The source clock is bus clock and can be divided by any even number.

17.8.9 MIICOMMAND

Address: 0x4000d02c

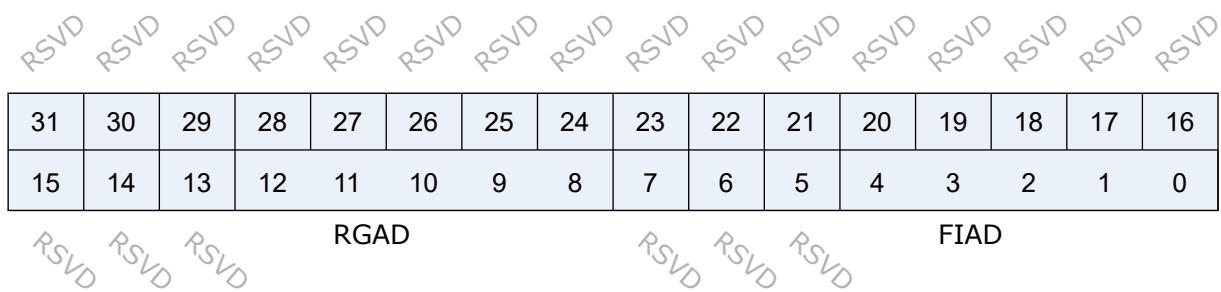
RSVD																	
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		

RSVD WCTRLDATA RSTAT SCANSTAT

Bits	Name	Type	Reset	Description
31:3	RSVD			
2	WCTRLDATA	r/w	1'b0	Write control data, setting this bit to 1 will trigger the command (auto cleared) Note: [2]/[1]/[0] cannot be asserted at the same time, execute one command at a time
1	RSTAT	r/w	1'b0	Read status, setting this bit to 1 will trigger the command (auto cleared) Note: [2]/[1]/[0] cannot be asserted at the same time, execute one command at a time
0	SCANSTAT	r/w	1'b0	Scan status, setting this bit to 1 will trigger the command (auto cleared) Note: [2]/[1]/[0] cannot be asserted at the same time, execute one command at a time

17.8.10 MIIADDRESS

Address: 0x4000d030



Bits	Name	Type	Reset	Description
31:13	RSVD			
12:8	RGAD	r/w	5'h0	Register Address
7:5	RSVD			
4:0	FIAD	r/w	5'h0	PHY Address

17.8.11 MIITX_DATA

Address: 0x4000d034

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | |

CTRLDATA

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	CTRLDATA	r/w	16'h0	Control Data to be written to PHY

17.8.12 MIIRX_DATA

Address: 0x4000d038

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | |

PRSD

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:0	PRSD	r	16'h0	Received Data from PHY

17.8.13 MIISTATUS

Address: 0x4000d03c

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | |



Bits	Name	Type	Reset	Description
31:2	RSVD			
1	MIIM_BUSY	r	1'b0	MIIM I/F busy signal 0: The MIIM I/F is ready. 1: The MIIM I/F is busy.
0	MIIM_LINKFAIL	r	1'b0	MIIM I/F link fail signal

17.8.14 MAC_ADDR0

Address: 0x4000d040

MAC_B2								MAC_B3							
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
MAC_B4								MAC_B5							

Bits	Name	Type	Reset	Description
31:24	MAC_B2	r/w	8'd0	Ethernet MAC address byte 2
23:16	MAC_B3	r/w	8'd0	Ethernet MAC address byte 3
15:8	MAC_B4	r/w	8'd0	Ethernet MAC address byte 4
7:0	MAC_B5	r/w	8'd0	Ethernet MAC address byte 5

17.8.15 MAC_ADDR1

Address: 0x4000d044

RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD
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
MAC_B0								MAC_B1							

Bits	Name	Type	Reset	Description
31:16	RSVD			
15:8	MAC_B0	r/w	8'd0	Ethernet MAC address byte 0

Bits	Name	Type	Reset	Description
7:0	MAC_B1	r/w	8'd0	Ethernet MAC address byte 1

17.8.16 HASH0_ADDR

Address: 0x4000d048

HASH0

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

HASH0

Bits	Name	Type	Reset	Description
31:0	HASH0	r/w	32'h0	Lower 32-bit of HASH register

17.8.17 HASH1_ADDR

Address: 0x4000d04c

HASH1

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

HASH1

Bits	Name	Type	Reset	Description
31:0	HASH1	r/w	32'h0	Upper 32-bit of HASH register

17.8.18 TXCTRL

Address: 0x4000d050

RSVD	TXPAUSERQ														
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

TXPAUSETV

Bits	Name	Type	Reset	Description
31:17	RSVD			
16	TXPAUSERQ	r/w	1'b0	TX Pause Request Writing 1 to this bit starts sending control frame and is automatically cleared to zero.
15:0	TXPAUSETV	r/w	16'h0	TX Pause Timer Value The value that is sent in the pause control frame.

18.1 USB introduction

USB (Universal Serial Bus), fully supports USB1.1 full-speed devices, and partially supports USB2.0.

18.2 USB main features

- Support USB full speed device-mode
- Supports 8 bidirectional endpoints: EP0 can be configured as control/bulk/interrupt/synchronization endpoints, EP1-EP7 can be configured as bulk/interrupt/synchronization endpoints
- Each endpoint contains TX and RX FIFOs with a depth of 64 bytes and supports DMA
- Support internal transceiver
- Support suspend/resume
- Support LPM
- Support multiple USB interrupt configurations

18.3 USB function description

18.3.1 USB steps

1. Configure the internal transceiver, the corresponding addresses are 0x40000228 and 0x4000022C
2. Configure usb_config and epx_config of each endpoint
3. Configure USB interrupt related registers
4. Configure USB DMA related (optional)
5. Configure GPIO as USB function (internal transmitter—function is 10)

-
6. 0x40000228[20]usb_enum is set to 1, so that the host recognizes that the USB device is inserted and triggers the enumeration process

18.3.2 Part of the register configuration and function description

- swrdy:Read only, only when this bit is 0, can write 1 to cr_usb_ep0_sw_rdy
- crsr:Writing 1 is automatically cleared. When the software allows the next packet to reply with ACK, write 1 to this field, and only the next packet will reply with ACK. For OUT/IN transaction data, it will be received or sent from FIFO (FIFO release once)
- e0snko:It needs to be set to 1, which means that the OUT transaction will reply NAK by default, and the data will not enter the FIFO (FIFO is not released)
- e0snki:It needs to be set to 1, which means that the IN transaction will reply NAK by default, and the data will not be sent from the FIFO (FIFO is not released)
- e0ss:Writing 1 is automatically cleared. Write 1 to this field when the software allows the next packet to reply STALL, then only the next packet will reply STALL
- epxdit:" Token package => trigger xxx_cmd_int => data package => trigger xxx_done_int => handshake package "
- epxcit:" Token package => trigger xxx_cmd_int => data package => trigger xxx_done_int => handshake package "
- ep0odit:" Token package => trigger xxx_cmd_int => data package => trigger xxx_done_int => handshake package "
- ep0ocit:" Token package => trigger xxx_cmd_int => data package => trigger xxx_done_int => handshake package "
- ep0idit:" Token package => trigger xxx_cmd_int => data package => trigger xxx_done_int => handshake package "
- ep0icit:" Token package => trigger xxx_cmd_int => data package => trigger xxx_done_int => handshake package "
- ep0sdit:" Token package => trigger xxx_cmd_int => data package => trigger xxx_done_int => handshake package "
- ep0scit:" Token package => trigger xxx_cmd_int => data package => trigger xxx_done_int => handshake package "
- exrs:Read only. Only when this bit is 0, can write 1 to cr_epx_rdy
- exr:Writing 1 is automatically cleared. When the software allows the next packet to reply ACK, write 1 to this field, then only the next packet will reply ACK (FIFO release once)
- exn:It needs to be set to 1, which means that the transaction will reply NAK by default. IN/OUT depends on the transmission direction configuration of the current endpoint (FIFO is not released)
- exs:Set when the software wants to suspend this endpoint, after setting this endpoint will always reply STALL

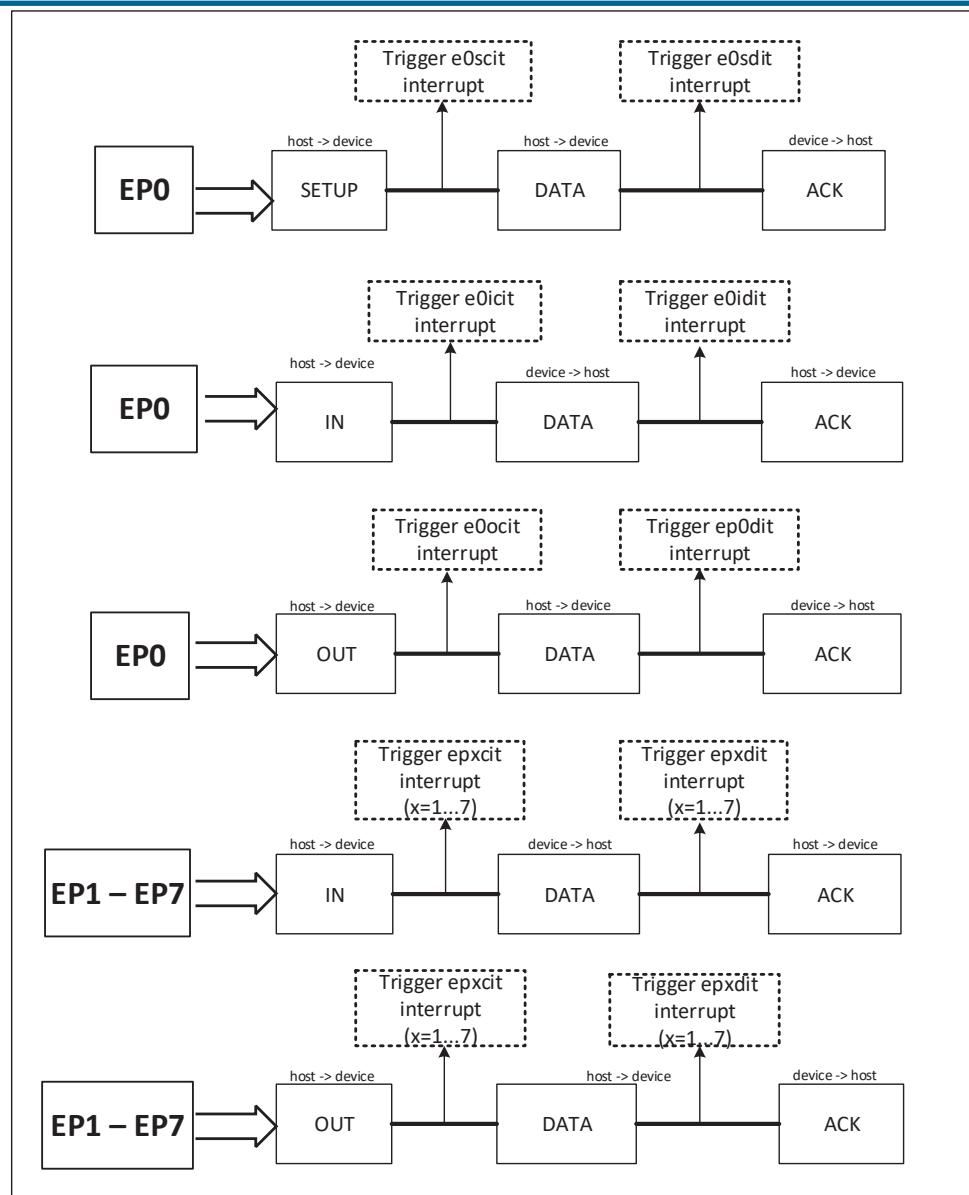


Fig. 18.1: USB interrupt trigger mode

18.3.3 USB enumeration phase interrupt processing flow

1. The first is a reset of more than 10ms, which will trigger the reset interrupt.
2. When the reset ends, the reset end interrupt will be triggered.
3. The SETUP transaction, IN transaction, and OUT transaction of the enumeration process will trigger e0sdit, e0icit, and ep0dit respectively.
4. After the enumeration is over, the IN transaction and OUT transaction between the host and the specific endpoint EPx will trigger epxxit and epdxit respectively.

18.3.4 Register operation flow of each transfer transaction

- Control transmission-SETUP transaction data reception:
 - Enter interrupt
 - Determine the e0sdit interrupt flag bit
 - What is stored in e0rfr is the data from the setup firm, which can be obtained by reading e0rfr according to e0rfc
 - Set a crsr to release subsequent transactions
 - Clear interrupt flag
 - Exit interrupt
- Control transmission-IN transaction data transmission:
 - Enter interrupt
 - Determine the e0idit interrupt flag bit
 - Wait for swrdy to be 0 before writing data to e0tfw
 - Set a crsr to release subsequent transactions
 - Clear interrupt flag
 - Exit interrupt
- Control transmission-OUT transaction data reception:
 - Enter interrupt
 - Determine the ep0dit interrupt flag bit
 - What is stored in e0rfr is the data from the OUT firm, which can be obtained by reading e0rfr according to e0rfc
 - Set a crsr to release subsequent transactions
 - Clear interrupt flag
 - Exit interrupt
- EPx(x=1…7)——IN transaction data transmission:
 - Enter interrupt
 - Determine the epxcit interrupt flag bit
 - Wait for exrs to be 0 before writing data to extfw (exs needs to be changed to 1 only when only 1 byte is sent)

- Set an exr to release subsequent transactions
- Clear interrupt flag
- Exit interrupt
- EPx(x=1…7)——OUT transaction data reception:
 - Enter interrupt
 - Determine the epexit interrupt flag bit
 - The data stored in exrfr is the data from the OUT firm, which can be obtained by reading exrfr according to exrfc
 - Set an exrs to release subsequent transactions
 - Clear interrupt flag
 - Exit interrupt

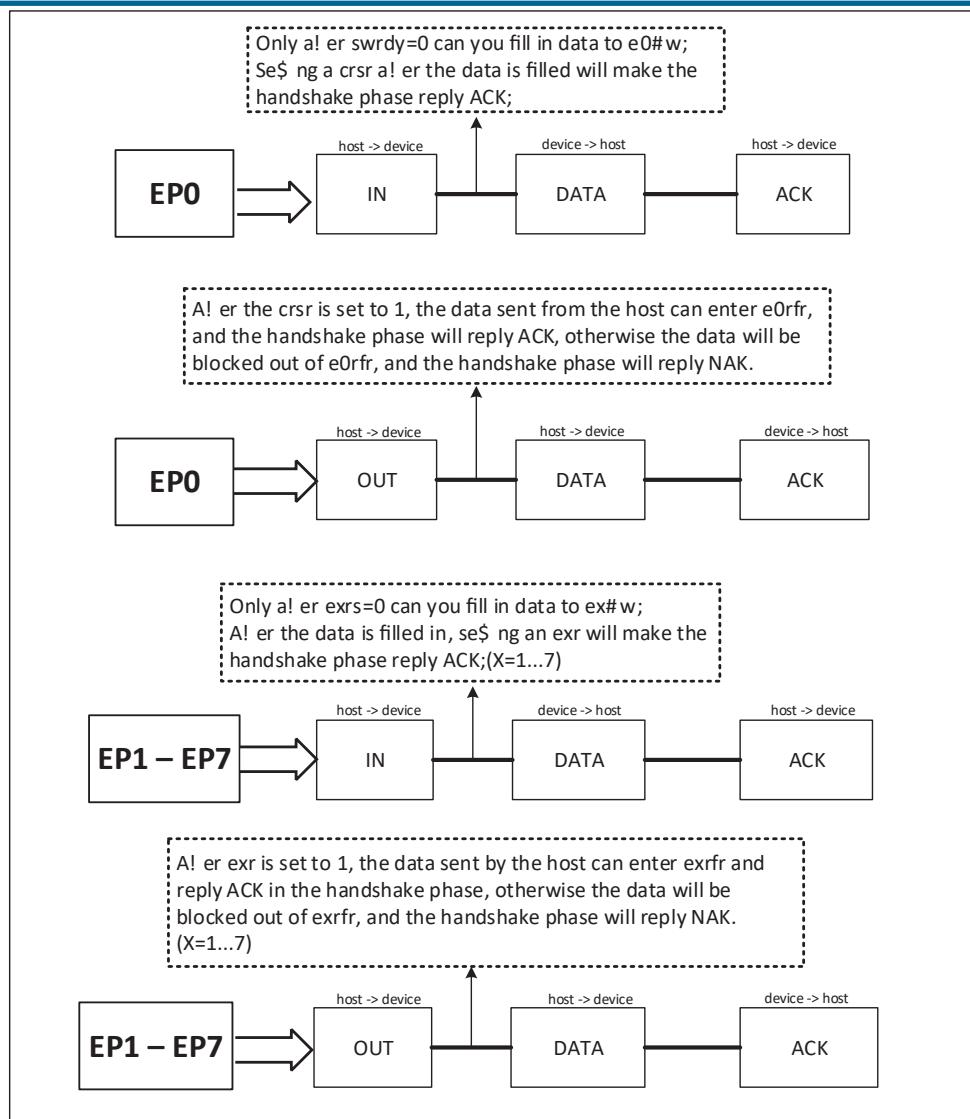


Fig. 18.2: USB communication method

Recommended configuration of internal transmitter register:

Table 18.1: Register configuration 1

usb_xcvr	value
usb_rcv	read only
usb_vip	read only
usb_vim	read only
usb_bd	read only
pu_usb	0->1
usb_sus	0
usb_spd	1
usb_enum	0->1

Table 18.1: Register configuration 1(continued)

	value
usb_xcvr	
usb_data_convert	0
usb_oeb	read only
usb_oeb_reg	1
usb_oeb_sel	0
usb_rout_pmos	3
usb_rout_nmos	3
pu_usb_ldo	0
usb_ldo_vfb	3

Table 18.2: Register configuration 2

	value
usb_xcvr_config	
usb_slewrate_p_rise	4
usb_slewrate_p_fall	3
usb_slewrate_m_rise	4
usb_slewrate_m_fall	3
usb_res_pullup_tune	2
reg_usb_use_ctrl	0
usb_str_drv	1
reg_usb_use_xcvr	1
usb_bd_vth	7
usb_v_hys_p	1
usb_v_hys_m	1

Note: You need to set pu_usb and usb_enum to one when you are ready to open the internal transceiver.

18.4 Register description

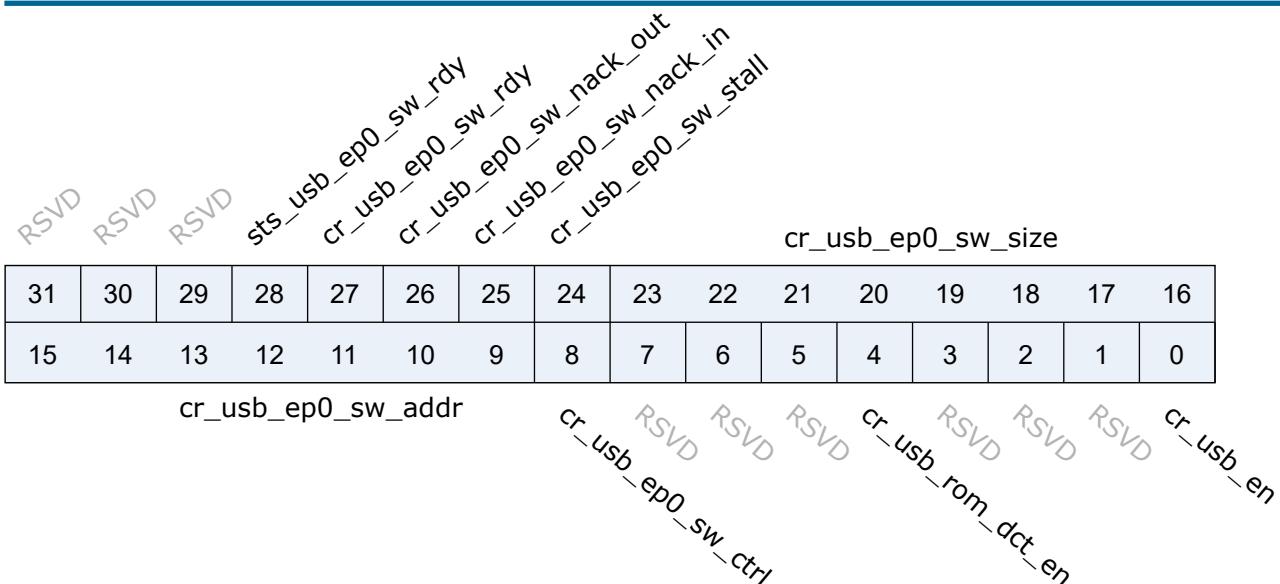
Name	Description
usb_config	USB configuration
usb_lpm_config	USB lpm configuration

Name	Description
usb_resume_config	USB resume configuration
usb_frame_no	USB frame number
usb_error	USB error
usb_int_en	USB interrupt enable
usb_int_sts	USB interrupt status
usb_int_mask	USB interrupt mask
usb_int_clear	USB interrupt clear
ep1_config	EP1 configuration
ep2_config	EP2 configuration
ep3_config	EP3 configuration
ep4_config	EP4 configuration
ep5_config	EP5 configuration
ep6_config	EP6 configuration
ep7_config	EP7 configuration
ep0_fifo_config	EP0 fifo configuration
ep0_fifo_status	EP0 fifo status
ep0_tx_fifo_wdata	EP0 tx fifo write data
ep0_rx_fifo_rdata	EP0 rx fifo write data
ep1_fifo_config	EP1 fifo configuration
ep1_fifo_status	EP1 fifo status
ep1_tx_fifo_wdata	EP1 tx fifo write data
ep1_rx_fifo_rdata	EP1 rx fifo write data
ep2_fifo_config	EP2 fifo configuration
ep2_fifo_status	EP2 fifo status
ep2_tx_fifo_wdata	EP2 tx fifo write data
ep2_rx_fifo_rdata	EP2 rx fifo write data
ep3_fifo_config	EP3 fifo configuration
ep3_fifo_status	EP3 fifo status
ep3_tx_fifo_wdata	EP3 tx fifo write data

Name	Description
ep3_rx_fifo_rdata	EP3 rx fifo write data
ep4_fifo_config	EP4 fifo configuration
ep4_fifo_status	EP4 fifo status
ep4_tx_fifo_wdata	EP4 tx fifo write data
ep4_rx_fifo_rdata	EP4 rx fifo write data
ep5_fifo_config	EP5 fifo configuration
ep5_fifo_status	EP5 fifo status
ep5_tx_fifo_wdata	EP5 tx fifo write data
ep5_rx_fifo_rdata	EP5 rx fifo read data
ep6_fifo_config	EP6 fifo configuration
ep6_fifo_status	EP6 fifo status
ep6_tx_fifo_wdata	EP6 tx fifo write data
ep6_rx_fifo_rdata	EP6 rx fifo read data
ep7_fifo_config	EP7 fifo configuration
ep7_fifo_status	EP7 fifo status
ep7_tx_fifo_wdata	EP7 tx fifo write data
ep7_rx_fifo_rdata	EP7 rx fifo read data
xcvr_if_config	

18.4.1 usb_config

Address: 0x4000d800

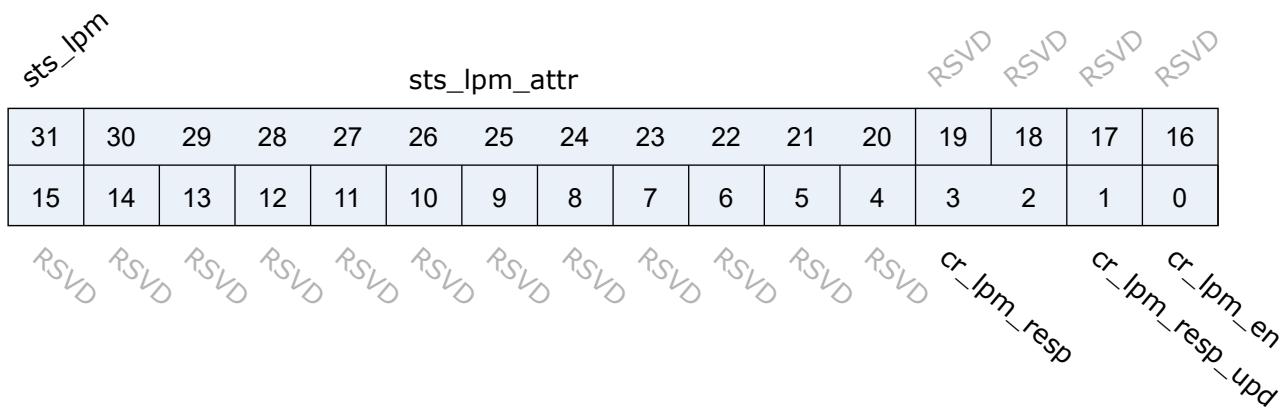


Bits	Name	Type	Reset	Description
31:29	RSVD			
28	sts_usb_ep0_sw_rdy	r	1'b0	EP0 transaction ready status bit. Asserted with sw_rdy, and de-asserted when ACK is sent/received.
27	cr_usb_ep0_sw_rdy	w1c	1'b0	EP0 transaction ready. When NACK is enabled, asserting this bit will allow one packet to be transferred even if NACK is asserted
26	cr_usb_ep0_sw_nack_out	r/w	1'b0	EP0 OUT/SETUP transaction nack response (SW control mode) Note: Should NOT enable both ep0_sw_nack_out and ep0_sw_stall at the same time
25	cr_usb_ep0_sw_nack_in	r/w	1'b1	EP0 IN transaction nack response (SW control mode) Note: Should NOT enable both ep0_sw_nack_in and ep0_sw_stall at the same time
24	cr_usb_ep0_sw_stall	w1c	1'b0	EP0 stall response (SW control mode) Note: Should NOT enable both ep0_sw_nack_in/out and ep0_sw_stall at the same time
23:16	cr_usb_ep0_sw_size	r/w	8'd0	EP0 transfer size (SW control mode)
15:9	cr_usb_ep0_sw_addr	r/w	7'd0	EP0 address (SW control mode)
8	cr_usb_ep0_sw_ctrl	r/w	1'b0	EP0 software control enable 1'b1: EP0 IN/OUT transaction is fully controlled by SW 1'b0: EP0 IN/OUT transaction is controlled by HW
7:5	RSVD			

Bits	Name	Type	Reset	Description
4	cr_usb_rom_dct_en	r/w	1'b1	Enable signal of ROM-based descriptors (don't care if ep0_sw_ctrl is asserted) 1'b1: USB descriptors stored in ROM will be used 1'b0: SW should prepare the descriptors requested by HOST
3:1	RSVD			
0	cr_usb_en	r/w	1'b0	Enable signal of USB function

18.4.2 usb_lpm_config

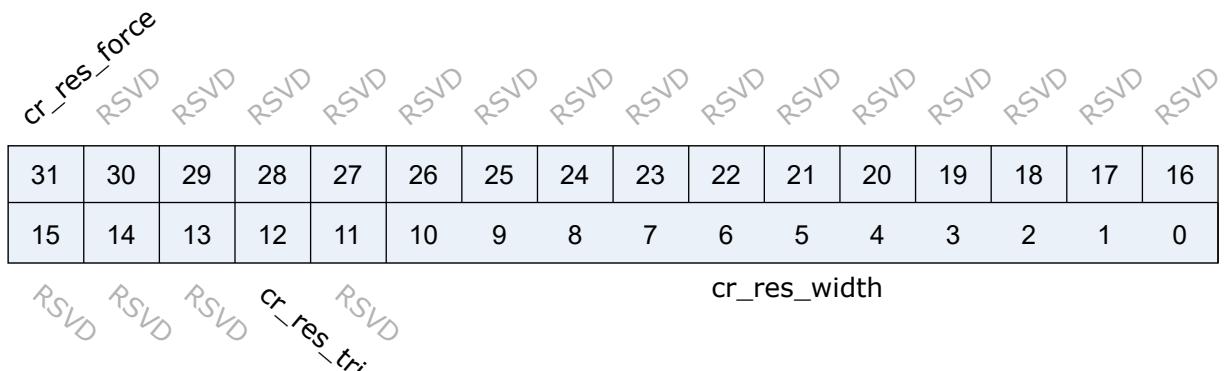
Address: 0x4000d804



Bits	Name	Type	Reset	Description
31	sts_lpm	r	1'b0	LPM status bit
30:20	sts_lpm_attr	r	11'h0	LPM attributes received in LPM packet
19:4	RSVD			
3:2	cr_lpm_resp	r/w	2'd2	Response when LPM packet is received 2'd3: NYET 2'd2: STALL 2'd1: NACK 2'd0: ACK
1	cr_lpm_resp_upd	w1c	1'b0	Response update signal (for async concern) Assert this bit when cr_lpm_resp is updated
0	cr_lpm_en	w1c	1'b0	LPM enable signal

18.4.3 usb_resume_config

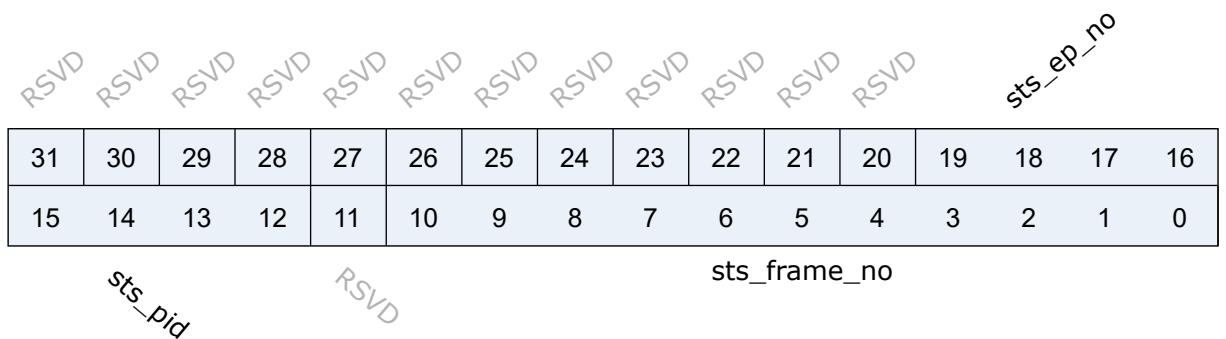
Address: 0x4000d808



Bits	Name	Type	Reset	Description
31	cr_res_force	r/w	1'b0	Force to output K-state
30:13	RSVD			
12	cr_res_trig	w1c	1'b0	Resume K-state trigger
11	RSVD			
10:0	cr_res_width	r/w	11'd26	Resume K-state width (unit: 2.67us)

18.4.4 usb_frame_no

Address: 0x4000d818



Bits	Name	Type	Reset	Description
31:20	RSVD			
19:16	sts_ep_no	r	4'd0	Endpoint number of the current transaction
15:12	sts_pid	r	4'd0	PID value of the current transaction
11	RSVD			

Bits	Name	Type	Reset	Description
10:0	sts_frame_no	r	11'h0	Current frame number

18.4.5 usb_error

Address: 0x4000d81c

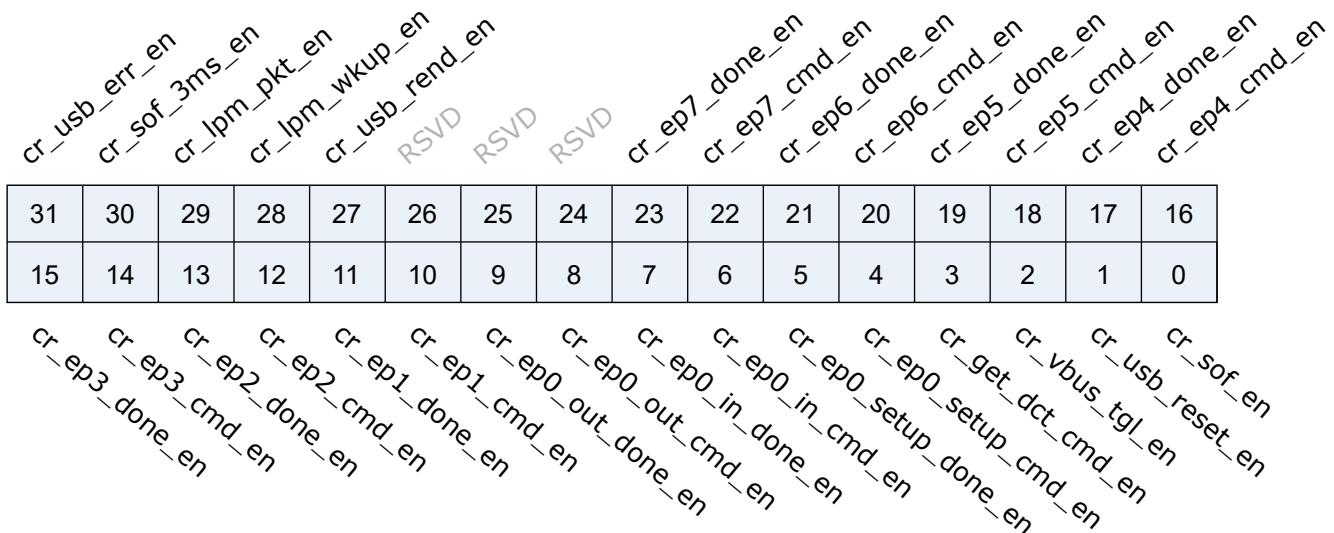
| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | | |

RSVD RSVD

Bits	Name	Type	Reset	Description
31:7	RSVD			
6	crc16_err	r	1'b0	Data CRC error occurs, cleared by cr_usb_err_clr
5	crc5_err	r	1'b0	Token CRC error occurs, cleared by cr_usb_err_clr
4	pid_cks_err	r	1'b0	PID check sum error occurs, cleared by cr_usb_err_clr
3	pid_seq_err	r	1'b0	PID sequence error occurs, cleared by cr_usb_err_clr
2	ivld_ep_err	r	1'b0	Invalid endpoint error occurs, cleared by cr_usb_err_clr
1	xfer_to_err	r	1'b0	Transfer time-out error occurs, cleared by cr_usb_err_clr
0	utmi_rx_err	r	1'b0	UTMI I/F RX error occurs, cleared by cr_usb_err_clr

18.4.6 usb_int_en

Address: 0x4000d820

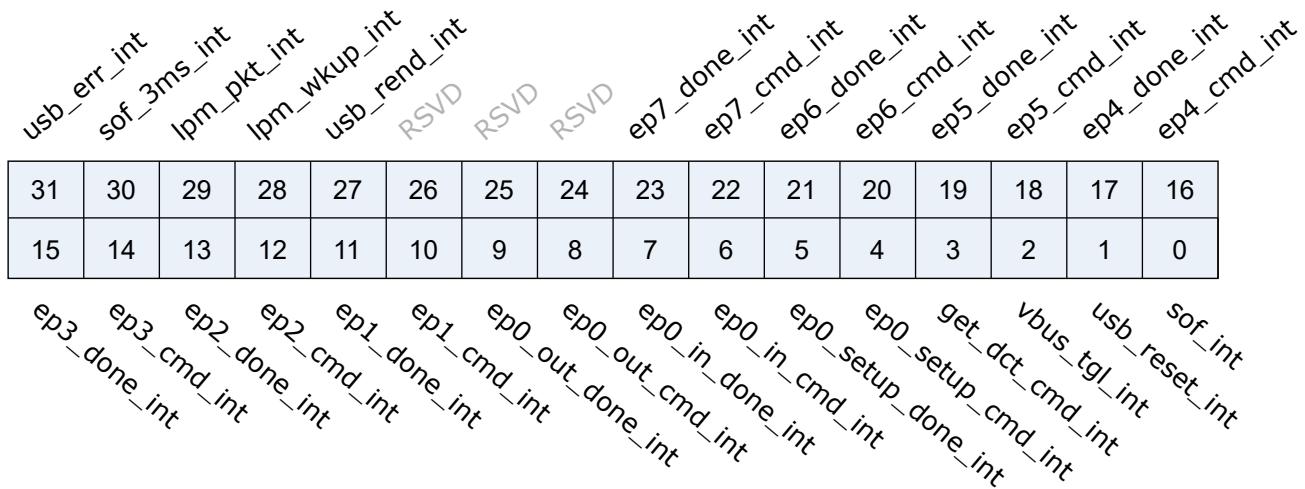


Bits	Name	Type	Reset	Description
31	cr_usb_err_en	r/w	1'b1	Interrupt enable of usb_err_int
30	cr_sof_3ms_en	r/w	1'b0	Interrupt enable of sof_3ms_int
29	cr_lpm_pkt_en	r/w	1'b0	Interrupt enable of lpm_pkt_int
28	cr_lpm_wkup_en	r/w	1'b0	Interrupt enable of lpm_wkup_int
27	cr_usb_rend_en	r/w	1'b0	Interrupt enable of usb_rend_int
26:24	RSVD			
23	cr_ep7_done_en	r/w	1'b1	Interrupt enable of ep7_done_int
22	cr_ep7_cmd_en	r/w	1'b1	Interrupt enable of ep7_cmd_int
21	cr_ep6_done_en	r/w	1'b1	Interrupt enable of ep6_done_int
20	cr_ep6_cmd_en	r/w	1'b1	Interrupt enable of ep6_cmd_int
19	cr_ep5_done_en	r/w	1'b1	Interrupt enable of ep5_done_int
18	cr_ep5_cmd_en	r/w	1'b1	Interrupt enable of ep5_cmd_int
17	cr_ep4_done_en	r/w	1'b1	Interrupt enable of ep4_done_int
16	cr_ep4_cmd_en	r/w	1'b1	Interrupt enable of ep4_cmd_int
15	cr_ep3_done_en	r/w	1'b1	Interrupt enable of ep3_done_int
14	cr_ep3_cmd_en	r/w	1'b1	Interrupt enable of ep3_cmd_int
13	cr_ep2_done_en	r/w	1'b1	Interrupt enable of ep2_done_int
12	cr_ep2_cmd_en	r/w	1'b1	Interrupt enable of ep2_cmd_int
11	cr_ep1_done_en	r/w	1'b1	Interrupt enable of ep1_done_int

Bits	Name	Type	Reset	Description
10	cr_ep1_cmd_en	r/w	1'b1	Interrupt enable of ep1_cmd_int
9	cr_ep0_out_done_en	r/w	1'b1	Interrupt enable of ep0_out_done_int
8	cr_ep0_out_cmd_en	r/w	1'b1	Interrupt enable of ep0_out_cmd_int
7	cr_ep0_in_done_en	r/w	1'b1	Interrupt enable of ep0_in_done_int
6	cr_ep0_in_cmd_en	r/w	1'b1	Interrupt enable of ep0_in_cmd_int
5	cr_ep0_setup_done_en	r/w	1'b1	Interrupt enable of ep0_setup_done_int
4	cr_ep0_setup_cmd_en	r/w	1'b1	Interrupt enable of ep0_setup_cmd_int
3	cr_get_dct_cmd_en	r/w	1'b1	Interrupt enable of get_dct_cmd_int
2	cr_vbus_tgl_en	r/w	1'b1	Interrupt enable of vbus_tgl_int
1	cr_usb_reset_en	r/w	1'b1	Interrupt enable of usb_reset_int
0	cr_sof_en	r/w	1'b1	Interrupt enable of sof_int

18.4.7 usb_int_sts

Address: 0x4000d824

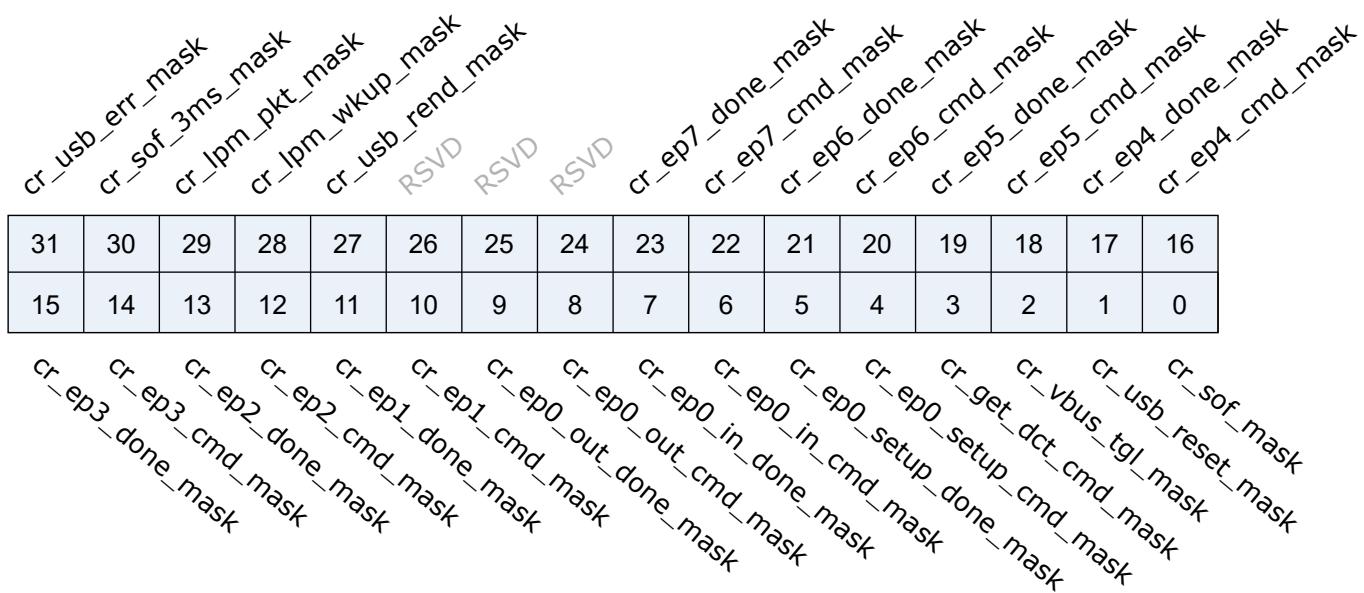


Bits	Name	Type	Reset	Description
31	usb_err_int	r	1'b0	USB error occurs, check <code>usb_error</code> for detailed error type
30	sof_3ms_int	r	1'b0	SOF is absent for 3 ms
29	lpm_pkt_int	r	1'b0	LPM packet is received
28	lpm_wkup_int	r	1'b0	LPM resume (wakeup) signal is received
27	usb rend int	r	1'b0	USB reset de-assert is triggered

Bits	Name	Type	Reset	Description
26:24	RSVD			
23	ep7_done_int	r	1'b0	EP7 IN or OUT command is finished
22	ep7_cmd_int	r	1'b0	EP7 IN or OUT command is received
21	ep6_done_int	r	1'b0	EP6 IN or OUT command is finished
20	ep6_cmd_int	r	1'b0	EP6 IN or OUT command is received
19	ep5_done_int	r	1'b0	EP5 IN or OUT command is finished
18	ep5_cmd_int	r	1'b0	EP5 IN or OUT command is received
17	ep4_done_int	r	1'b0	EP4 IN or OUT command is finished
16	ep4_cmd_int	r	1'b0	EP4 IN or OUT command is received
15	ep3_done_int	r	1'b0	EP3 IN or OUT command is finished
14	ep3_cmd_int	r	1'b0	EP3 IN or OUT command is received
13	ep2_done_int	r	1'b0	EP2 IN or OUT command is finished
12	ep2_cmd_int	r	1'b0	EP2 IN or OUT command is received
11	ep1_done_int	r	1'b0	EP1 IN or OUT command is finished
10	ep1_cmd_int	r	1'b0	EP1 IN or OUT command is received
9	ep0_out_done_int	r	1'b0	EP0 OUT command is finished
8	ep0_out_cmd_int	r	1'b0	EP0 OUT command is received
7	ep0_in_done_int	r	1'b0	EP0 IN command is finished
6	ep0_in_cmd_int	r	1'b0	EP0 IN command is received
5	ep0_setup_done_int	r	1'b0	EP0 SETUP command is finished
4	ep0_setup_cmd_int	r	1'b0	EP0 SETUP command is received
3	get_dct_cmd_int	r	1'b0	GET_DESCRIPTOR command is received
2	vbus_tgl_int	r	1'b0	VBUS detection is toggled, check 0x1FC[31] for vbus_detect status
1	usb_reset_int	r	1'b0	USB reset is triggered
0	sof_int	r	1'b0	SOF is received

18.4.8 usb_int_mask

Address: 0x4000d828

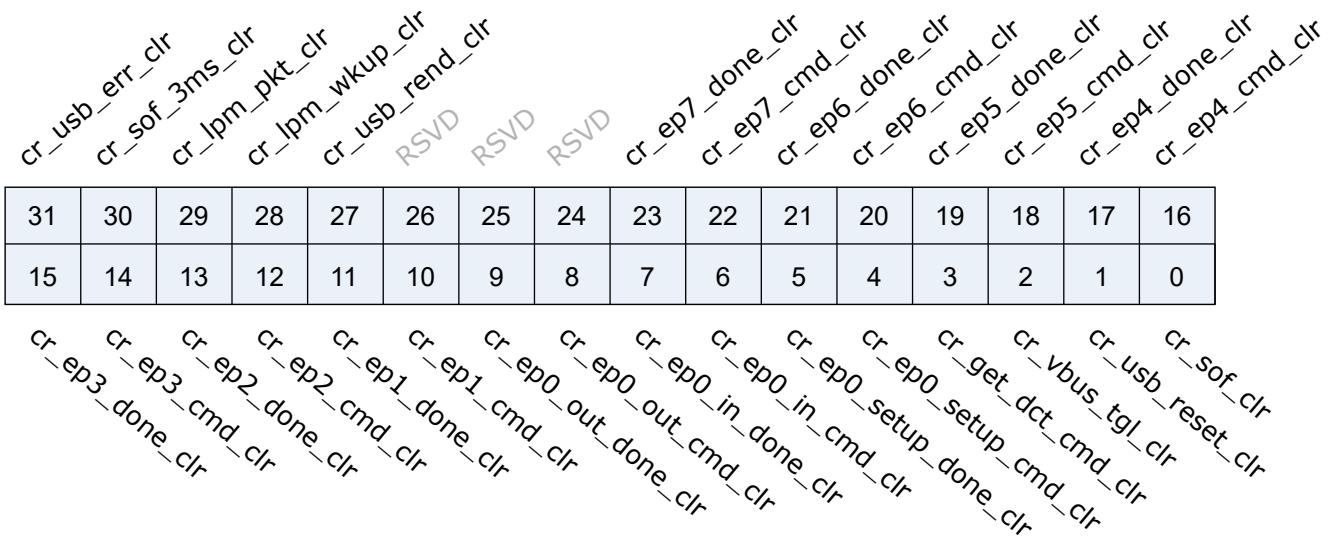


Bits	Name	Type	Reset	Description
31	cr_usb_err_mask	r/w	1'b1	Interrupt mask of usb_err_int
30	cr_sof_3ms_mask	r/w	1'b1	Interrupt mask of sof_3ms_int
29	cr_lpm_pkt_mask	r/w	1'b1	Interrupt mask of lpm_pkt_int
28	cr_lpm_wkup_mask	r/w	1'b1	Interrupt mask of lpm_wkup_int
27	cr_usb_rend_mask	r/w	1'b1	Interrupt mask of usb_rend_int
26:24	RSVD			
23	cr_ep7_done_mask	r/w	1'b1	Interrupt mask of ep7_done_int
22	cr_ep7_cmd_mask	r/w	1'b1	Interrupt mask of ep7_cmd_int
21	cr_ep6_done_mask	r/w	1'b1	Interrupt mask of ep6_done_int
20	cr_ep6_cmd_mask	r/w	1'b1	Interrupt mask of ep6_cmd_int
19	cr_ep5_done_mask	r/w	1'b1	Interrupt mask of ep5_done_int
18	cr_ep5_cmd_mask	r/w	1'b1	Interrupt mask of ep5_cmd_int
17	cr_ep4_done_mask	r/w	1'b1	Interrupt mask of ep4_done_int
16	cr_ep4_cmd_mask	r/w	1'b1	Interrupt mask of ep4_cmd_int
15	cr_ep3_done_mask	r/w	1'b1	Interrupt mask of ep3_done_int
14	cr_ep3_cmd_mask	r/w	1'b1	Interrupt mask of ep3_cmd_int
13	cr_ep2_done_mask	r/w	1'b1	Interrupt mask of ep2_done_int
12	cr_ep2_cmd_mask	r/w	1'b1	Interrupt mask of ep2_cmd_int

Bits	Name	Type	Reset	Description
11	cr_ep1_done_mask	r/w	1'b1	Interrupt mask of ep1_done_int
10	cr_ep1_cmd_mask	r/w	1'b1	Interrupt mask of ep1_cmd_int
9	cr_ep0_out_done_mask	r/w	1'b1	Interrupt mask of ep0_out_done_int
8	cr_ep0_out_cmd_mask	r/w	1'b1	Interrupt mask of ep0_out_cmd_int
7	cr_ep0_in_done_mask	r/w	1'b1	Interrupt mask of ep0_in_done_int
6	cr_ep0_in_cmd_mask	r/w	1'b1	Interrupt mask of ep0_in_cmd_int
5	cr_ep0_setup_done_mask	r/w	1'b1	Interrupt mask of ep0_setup_done_int
4	cr_ep0_setup_cmd_mask	r/w	1'b1	Interrupt mask of ep0_setup_cmd_int
3	cr_get_dct_cmd_mask	r/w	1'b1	Interrupt mask of get_dct_cmd_int
2	cr_vbus_tgl_mask	r/w	1'b1	Interrupt mask of vbus_tgl_int
1	cr_usb_reset_mask	r/w	1'b1	Interrupt mask of usb_reset_int
0	cr_sof_mask	r/w	1'b1	Interrupt mask of sof_int

18.4.9 usb_int_clear

Address: 0x4000d82c

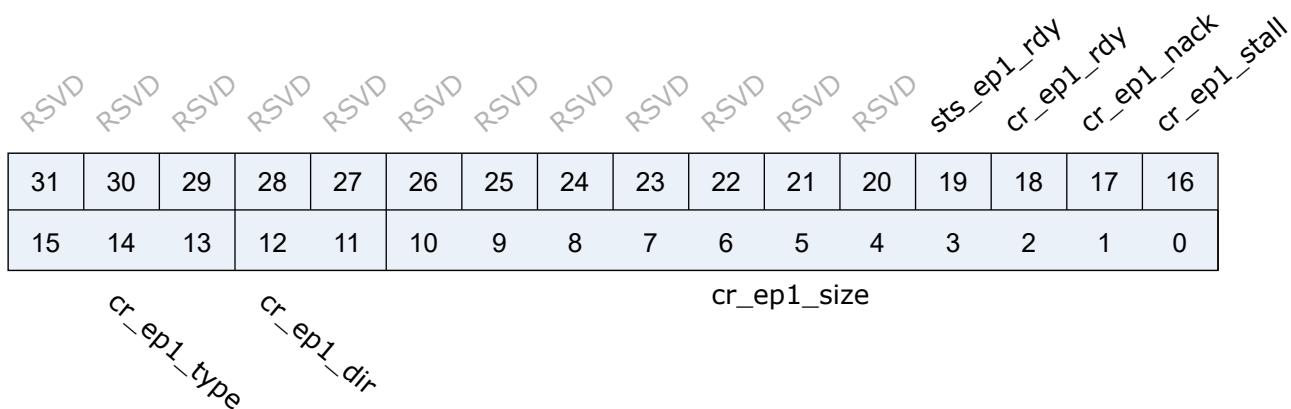


Bits	Name	Type	Reset	Description
31	cr_usb_err_clr	w1c	1'b0	Interrupt clear of usb_err_int
30	cr_sof_3ms_clr	w1c	1'b0	Interrupt clear of sof_3ms_int
29	cr_lpm_pkt_clr	w1c	1'b0	Interrupt clear of lpm_pkt_int

Bits	Name	Type	Reset	Description
28	cr_lpm_wkup_clr	w1c	1'b0	Interrupt clear of lpm_wkup_int
27	cr_usb_rend_clr	w1c	1'b0	Interrupt clear of usb_rend_int
26:24	RSVD			
23	cr_ep7_done_clr	w1c	1'b0	Interrupt clear of ep7_done_int
22	cr_ep7_cmd_clr	w1c	1'b0	Interrupt clear of ep7_cmd_int
21	cr_ep6_done_clr	w1c	1'b0	Interrupt clear of ep6_done_int
20	cr_ep6_cmd_clr	w1c	1'b0	Interrupt clear of ep6_cmd_int
19	cr_ep5_done_clr	w1c	1'b0	Interrupt clear of ep5_done_int
18	cr_ep5_cmd_clr	w1c	1'b0	Interrupt clear of ep5_cmd_int
17	cr_ep4_done_clr	w1c	1'b0	Interrupt clear of ep4_done_int
16	cr_ep4_cmd_clr	w1c	1'b0	Interrupt clear of ep4_cmd_int
15	cr_ep3_done_clr	w1c	1'b0	Interrupt clear of ep3_done_int
14	cr_ep3_cmd_clr	w1c	1'b0	Interrupt clear of ep3_cmd_int
13	cr_ep2_done_clr	w1c	1'b0	Interrupt clear of ep2_done_int
12	cr_ep2_cmd_clr	w1c	1'b0	Interrupt clear of ep2_cmd_int
11	cr_ep1_done_clr	w1c	1'b0	Interrupt clear of ep1_done_int
10	cr_ep1_cmd_clr	w1c	1'b0	Interrupt clear of ep1_cmd_int
9	cr_ep0_out_done_clr	w1c	1'b0	Interrupt clear of ep0_out_done_int
8	cr_ep0_out_cmd_clr	w1c	1'b0	Interrupt clear of ep0_out_cmd_int
7	cr_ep0_in_done_clr	w1c	1'b0	Interrupt clear of ep0_in_done_int
6	cr_ep0_in_cmd_clr	w1c	1'b0	Interrupt clear of ep0_in_cmd_int
5	cr_ep0_setup_done_clr	w1c	1'b0	Interrupt clear of ep0_setup_done_int
4	cr_ep0_setup_cmd_clr	w1c	1'b0	Interrupt clear of ep0_setup_cmd_int
3	cr_get_dct_cmd_clr	w1c	1'b0	Interrupt clear of get_dct_cmd_int
2	cr_vbus_tgl_clr	w1c	1'b0	Interrupt clear of vbus_tgl_int
1	cr_usb_reset_clr	w1c	1'b0	Interrupt clear of usb_reset_int
0	cr_sof_clr	w1c	1'b0	Interrupt clear of sof_int

18.4.10 ep1_config

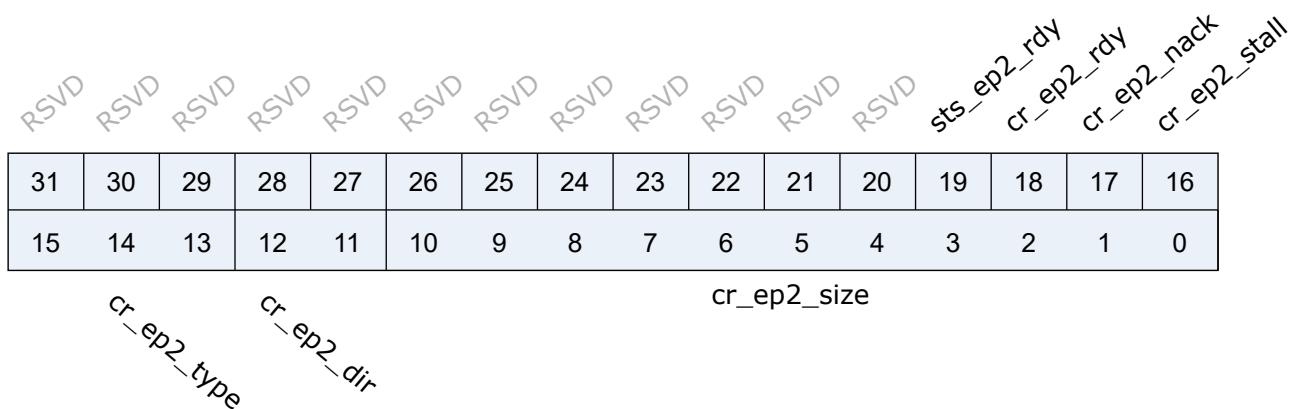
Address: 0x4000d840



Bits	Name	Type	Reset	Description
31:20	RSVD			
19	sts_ep1_rdy	r	1'b0	Endpoint ready status bit. Asserted with ep_rdy, and deasserted when ACK is sent/received.
18	cr_ep1_rdy	w1c	1'b0	Endpoint ready. When Endpoint NACK is enabled, asserting this bit will allow one packet to be transferred
17	cr_ep1_nack	r/w	1'b1	Endpoint NACK response enable, should not be enabled with STALL at the same time
16	cr_ep1_stall	r/w	1'b0	Endpoint STALL response enable, should not be enabled with NACK at the same time
15:13	cr_ep1_type	r/w	3'b100	Endpoint type 3'b101: CTRL 3'b010: ISO 3'b100: BULK 3'b000: INT Others: Reserved
12:11	cr_ep1_dir	r/w	2'b01	Endpoint direction 2'b00: Disabled 2'b01: IN 2'b10: OUT 2'b11: Reserved
10:0	cr_ep1_size	r/w	11'd64	Endpoint max packet size

18.4.11 ep2_config

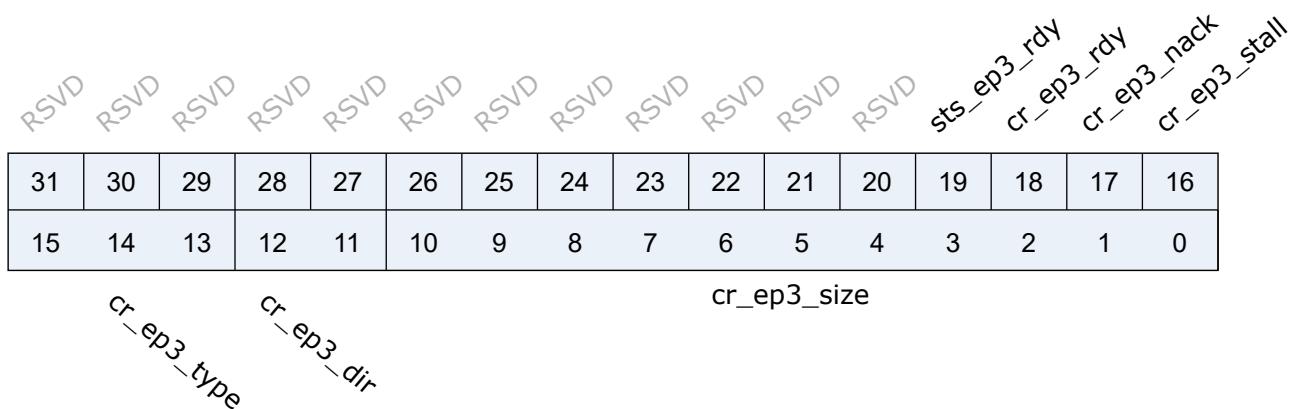
Address: 0x4000d844



Bits	Name	Type	Reset	Description
31:20	RSVD			
19	sts_ep2_rdy	r	1'b0	Endpoint ready status bit. Asserted with ep_rdy, and de-asserted when ACK is sent/received.
18	cr_ep2_rdy	w1c	1'b0	Endpoint ready. When Endpoint NACK is enabled, asserting this bit will allow one packet to be transferred
17	cr_ep2_nack	r/w	1'b1	Endpoint NACK response enable, should not be enabled with STALL at the same time
16	cr_ep2_stall	r/w	1'b0	Endpoint STALL response enable, should not be enabled with NACK at the same time
15:13	cr_ep2_type	r/w	3'b100	Endpoint type 3'b101: CTRL 3'b010: ISO 3'b100: BULK 3'b000: INT Others: Reserved
12:11	cr_ep2_dir	r/w	2'b01	Endpoint direction 2'b00: Disabled 2'b01: IN 2'b10: OUT 2'b11: Reserved
10:0	cr_ep2_size	r/w	11'd64	Endpoint max packet size

18.4.12 ep3_config

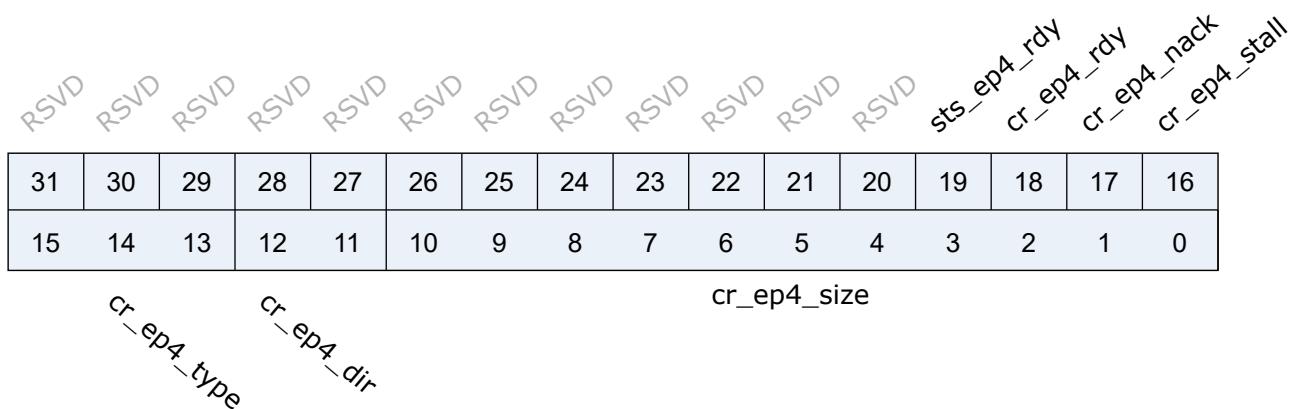
Address: 0x4000d848



Bits	Name	Type	Reset	Description
31:20	RSVD			
19	sts_ep3_rdy	r	1'b0	Endpoint ready status bit. Asserted with ep_rdy, and deasserted when ACK is sent/received.
18	cr_ep3_rdy	w1c	1'b0	Endpoint ready. When Endpoint NACK is enabled, asserting this bit will allow one packet to be transferred
17	cr_ep3_nack	r/w	1'b1	Endpoint NACK response enable, should not be enabled with STALL at the same time
16	cr_ep3_stall	r/w	1'b0	Endpoint STALL response enable, should not be enabled with NACK at the same time
15:13	cr_ep3_type	r/w	3'b100	Endpoint type 3'b101: CTRL 3'b010: ISO 3'b100: BULK 3'b000: INT Others: Reserved
12:11	cr_ep3_dir	r/w	2'b01	Endpoint direction 2'b00: Disabled 2'b01: IN 2'b10: OUT 2'b11: Reserved
10:0	cr_ep3_size	r/w	11'd64	Endpoint max packet size

18.4.13 ep4_config

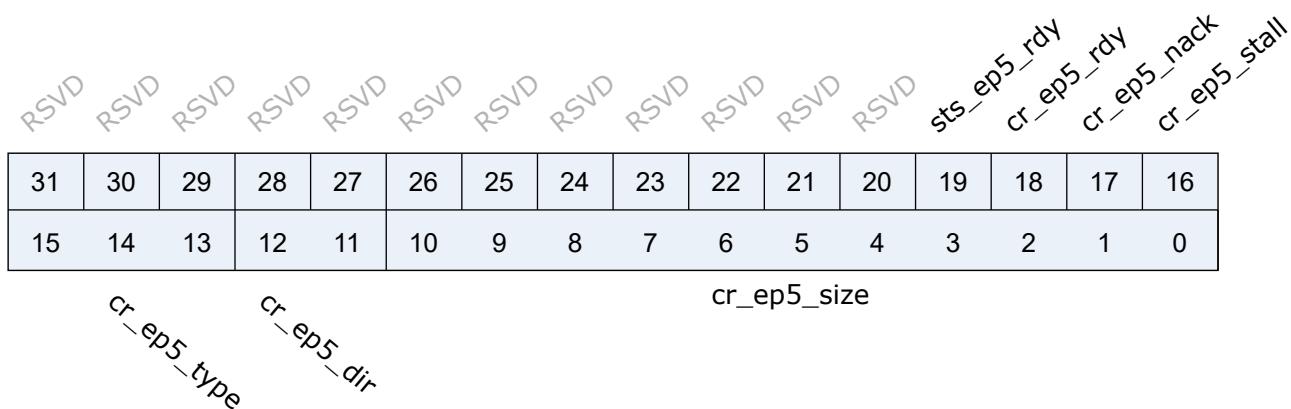
Address: 0x4000d84c



Bits	Name	Type	Reset	Description
31:20	RSVD			
19	sts_ep4_rdy	r	1'b0	Endpoint ready status bit. Asserted with ep_rdy, and deasserted when ACK is sent/received.
18	cr_ep4_rdy	w1c	1'b0	Endpoint ready. When Endpoint NACK is enabled, asserting this bit will allow one packet to be transferred
17	cr_ep4_nack	r/w	1'b1	Endpoint NACK response enable, should not be enabled with STALL at the same time
16	cr_ep4_stall	r/w	1'b0	Endpoint STALL response enable, should not be enabled with NACK at the same time
15:13	cr_ep4_type	r/w	3'b100	Endpoint type 3'b101: CTRL 3'b010: ISO 3'b100: BULK 3'b000: INT Others: Reserved
12:11	cr_ep4_dir	r/w	2'b01	Endpoint direction 2'b00: Disabled 2'b01: IN 2'b10: OUT 2'b11: Reserved
10:0	cr_ep4_size	r/w	11'd64	Endpoint max packet size

18.4.14 ep5_config

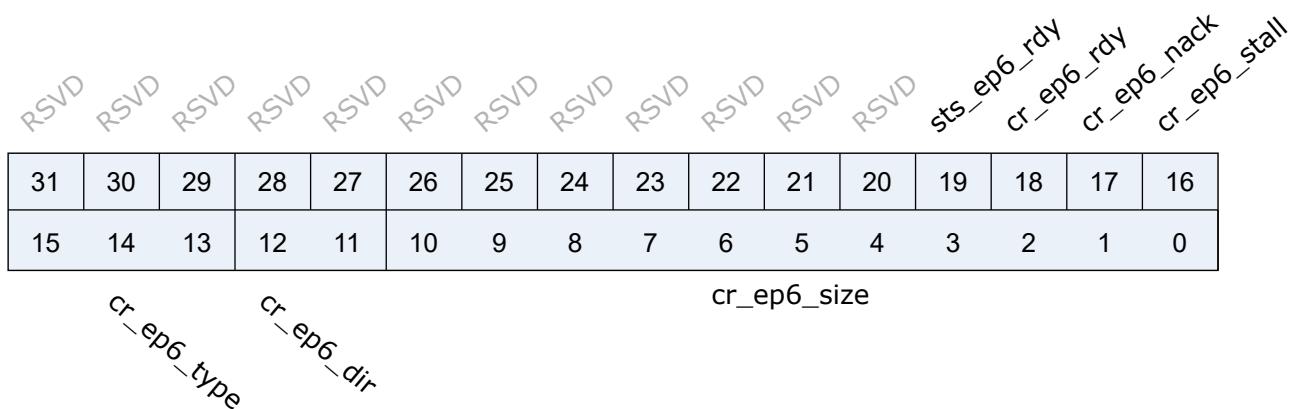
Address: 0x4000d850



Bits	Name	Type	Reset	Description
31:20	RSVD			
19	sts_ep5_rdy	r	1'b0	Endpoint ready status bit. Asserted with ep_rdy, and deasserted when ACK is sent/received.
18	cr_ep5_rdy	w1c	1'b0	Endpoint ready. When Endpoint NACK is enabled, asserting this bit will allow one packet to be transferred
17	cr_ep5_nack	r/w	1'b1	Endpoint NACK response enable, should not be enabled with STALL at the same time
16	cr_ep5_stall	r/w	1'b0	Endpoint STALL response enable, should not be enabled with NACK at the same time
15:13	cr_ep5_type	r/w	3'b100	Endpoint type 3'b101: CTRL 3'b010: ISO 3'b100: BULK 3'b000: INT Others: Reserved
12:11	cr_ep5_dir	r/w	2'b01	Endpoint direction 2'b00: Disabled 2'b01: IN 2'b10: OUT 2'b11: Reserved
10:0	cr_ep5_size	r/w	11'd64	Endpoint max packet size

18.4.15 ep6_config

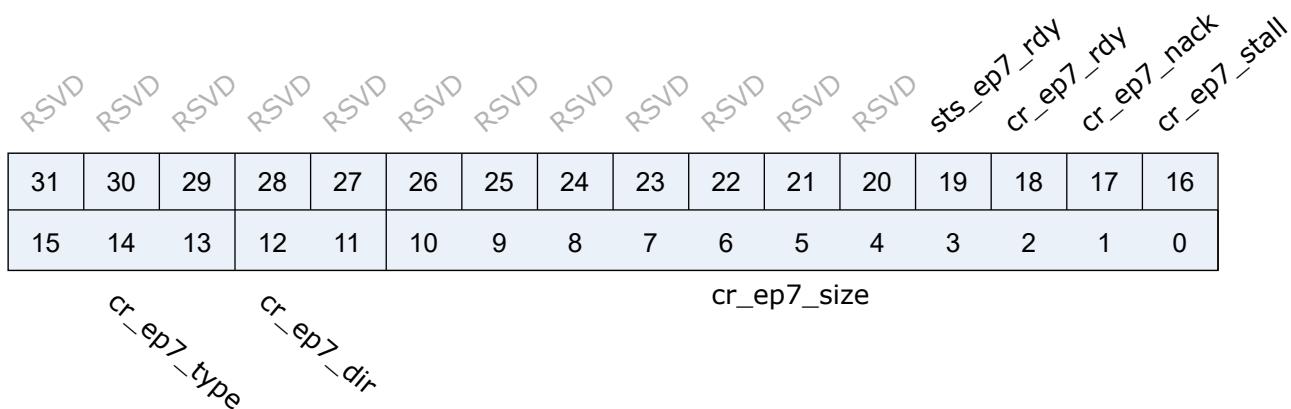
Address: 0x4000d854



Bits	Name	Type	Reset	Description
31:20	RSVD			
19	sts_ep6_rdy	r	1'b0	Endpoint ready status bit. Asserted with ep_rdy, and deasserted when ACK is sent/received.
18	cr_ep6_rdy	w1c	1'b0	Endpoint ready. When Endpoint NACK is enabled, asserting this bit will allow one packet to be transferred
17	cr_ep6_nack	r/w	1'b1	Endpoint NACK response enable, should not be enabled with STALL at the same time
16	cr_ep6_stall	r/w	1'b0	Endpoint STALL response enable, should not be enabled with NACK at the same time
15:13	cr_ep6_type	r/w	3'b100	Endpoint type 3'b101: CTRL 3'b010: ISO 3'b100: BULK 3'b000: INT Others: Reserved
12:11	cr_ep6_dir	r/w	2'b01	Endpoint direction 2'b00: Disabled 2'b01: IN 2'b10: OUT 2'b11: Reserved
10:0	cr_ep6_size	r/w	11'd64	Endpoint max packet size

18.4.16 ep7_config

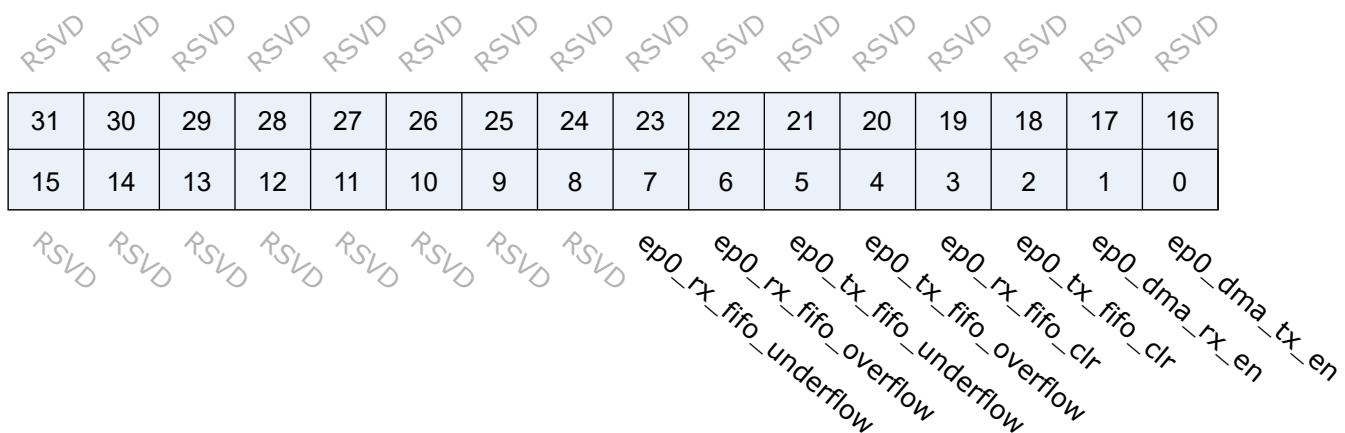
Address: 0x4000d858



Bits	Name	Type	Reset	Description
31:20	RSVD			
19	sts_ep7_rdy	r	1'b0	Endpoint ready status bit. Asserted with ep_rdy, and de-asserted when ACK is sent/received.
18	cr_ep7_rdy	w1c	1'b0	Endpoint ready. When Endpoint NACK is enabled, asserting this bit will allow one packet to be transferred
17	cr_ep7_nack	r/w	1'b1	Endpoint NACK response enable, should not be enabled with STALL at the same time
16	cr_ep7_stall	r/w	1'b0	Endpoint STALL response enable, should not be enabled with NACK at the same time
15:13	cr_ep7_type	r/w	3'b100	Endpoint type 3'b101: CTRL 3'b010: ISO 3'b100: BULK 3'b000: INT Others: Reserved
12:11	cr_ep7_dir	r/w	2'b01	Endpoint direction 2'b00: Disabled 2'b01: IN 2'b10: OUT 2'b11: Reserved
10:0	cr_ep7_size	r/w	11'd64	Endpoint max packet size

18.4.17 ep0_fifo_config

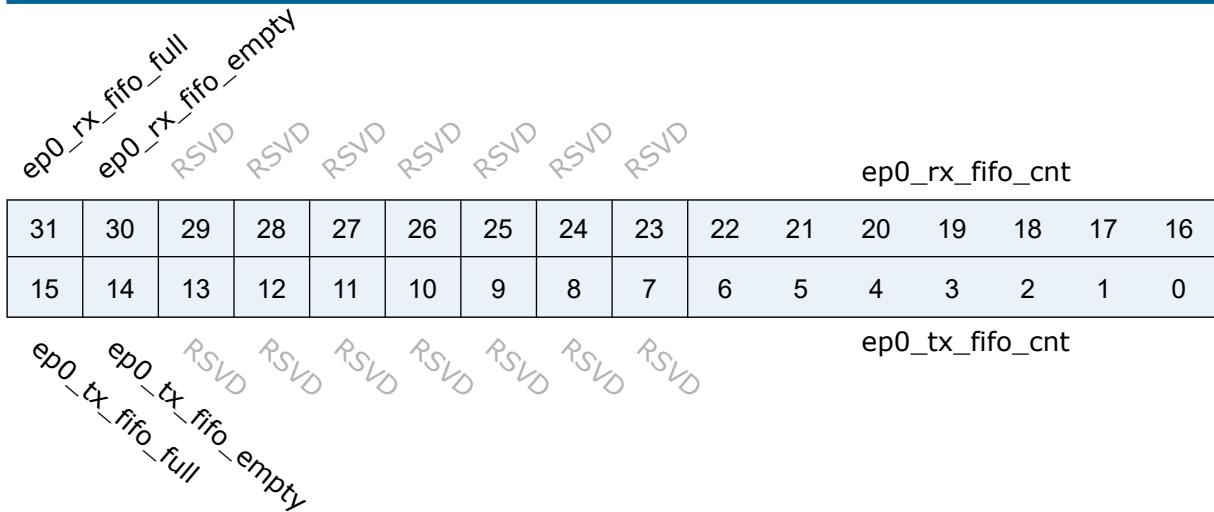
Address: 0x4000d900



Bits	Name	Type	Reset	Description
31:8	RSVD			
7	ep0_rx_fifo_underflow	r	1'b0	Underflow flag of RX FIFO, can be cleared by rx_fifo_clr
6	ep0_rx_fifo_overflow	r	1'b0	Overflow flag of RX FIFO, can be cleared by rx_fifo_clr
5	ep0_tx_fifo_underflow	r	1'b0	Underflow flag of TX FIFO, can be cleared by tx_fifo_clr
4	ep0_tx_fifo_overflow	r	1'b0	Overflow flag of TX FIFO, can be cleared by tx_fifo_clr
3	ep0_rx_fifo_clr	w1c	1'b0	Clear signal of RX FIFO
2	ep0_tx_fifo_clr	w1c	1'b0	Clear signal of TX FIFO
1	ep0_dma_rx_en	r/w	1'b0	Enable signal of dma_rx_req/ack interface
0	ep0_dma_tx_en	r/w	1'b0	Enable signal of dma_tx_req/ack interface

18.4.18 ep0_fifo_status

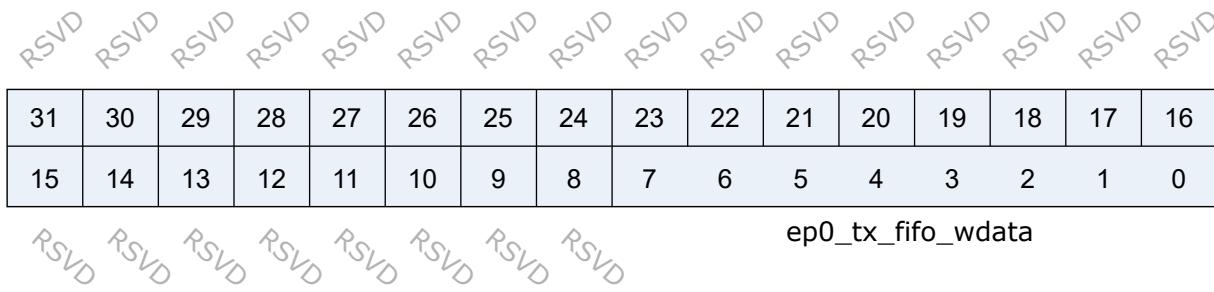
Address: 0x4000d904



Bits	Name	Type	Reset	Description
31	ep0_rx_fifo_full	r	1'b0	RX FIFO full flag
30	ep0_rx_fifo_empty	r	1'b1	RX FIFO empty flag
29:23	RSVD			
22:16	ep0_rx_fifo_cnt	r	7'd0	RX FIFO available count
15	ep0_tx_fifo_full	r	1'b0	TX FIFO full flag
14	ep0_tx_fifo_empty	r	1'b1	TX FIFO empty flag
13:7	RSVD			
6:0	ep0_tx_fifo_cnt	r	7'd64	TX FIFO available count

18.4.19 ep0_tx_fifo_wdata

Address: 0x4000d908



Bits	Name	Type	Reset	Description
31:8	RSVD			
7:0	ep0_tx_fifo_wdata	w	x	

18.4.20 ep0_rx_fifo_rdata

Address: 0x4000d90c

RSVD															
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

ep0_rx_fifo_rdata

Bits	Name	Type	Reset	Description
31:8	RSVD			
7:0	ep0_rx_fifo_rdata	r	8'h0	

18.4.21 ep1_fifo_config

Address: 0x4000d910

RSVD															
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

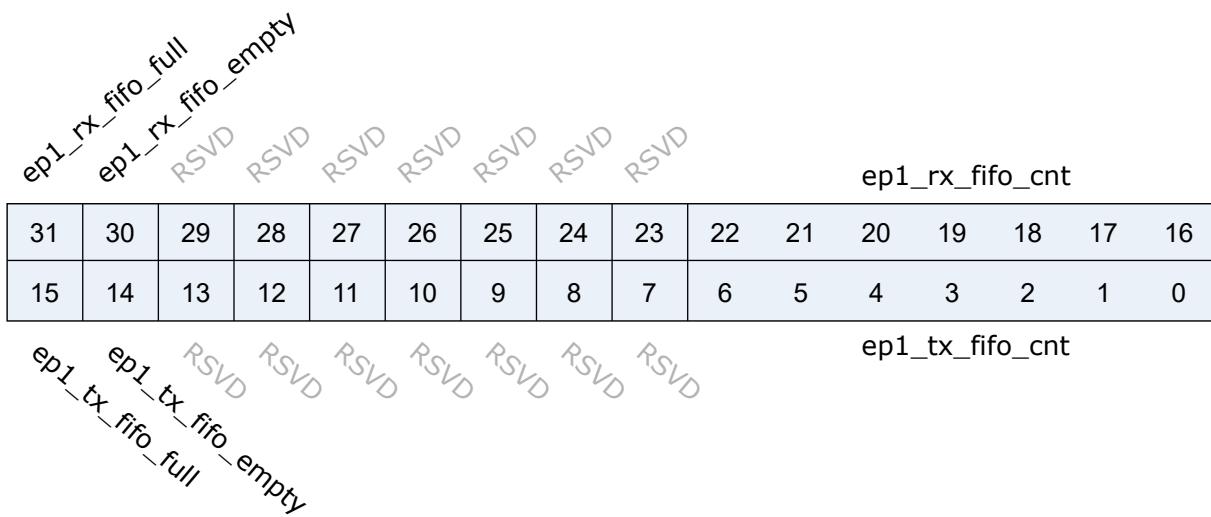
ep1_rx_fifo_underflow ep1_rx_fifo_overflow ep1_tx_fifo_underflow ep1_tx_fifo_overflow ep1_rx_fifo_clr ep1_tx_fifo_clr ep1_dma_rx_en ep1_dma_tx_en

Bits	Name	Type	Reset	Description
31:8	RSVD			
7	ep1_rx_fifo_underflow	r	1'b0	Underflow flag of RX FIFO, can be cleared by rx_fifo_clr
6	ep1_rx_fifo_overflow	r	1'b0	Overflow flag of RX FIFO, can be cleared by rx_fifo_clr
5	ep1_tx_fifo_underflow	r	1'b0	Underflow flag of TX FIFO, can be cleared by tx_fifo_clr
4	ep1_tx_fifo_overflow	r	1'b0	Overflow flag of TX FIFO, can be cleared by tx_fifo_clr

Bits	Name	Type	Reset	Description
3	ep1_rx_fifo_clr	w1c	1'b0	Clear signal of RX FIFO
2	ep1_tx_fifo_clr	w1c	1'b0	Clear signal of TX FIFO
1	ep1_dma_rx_en	r/w	1'b0	Enable signal of dma_rx_req/ack interface
0	ep1_dma_tx_en	r/w	1'b0	Enable signal of dma_tx_req/ack interface

18.4.22 ep1_fifo_status

Address: 0x4000d914



Bits	Name	Type	Reset	Description
31	ep1_rx_fifo_full	r	1'b0	RX FIFO full flag
30	ep1_rx_fifo_empty	r	1'b1	RX FIFO empty flag
29:23	RSVD			
22:16	ep1_rx_fifo_cnt	r	7'd0	RX FIFO available count
15	ep1_tx_fifo_full	r	1'b0	TX FIFO full flag
14	ep1_tx_fifo_empty	r	1'b1	TX FIFO empty flag
13:7	RSVD			
6:0	ep1_tx_fifo_cnt	r	7'd64	TX FIFO available count

18.4.23 ep1_tx_fifo_wdata

Address: 0x4000d918

RSVD																							
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								

ep1_tx_fifo_wdata

Bits	Name	Type	Reset	Description
31:8	RSVD			
7:0	ep1_tx_fifo_wdata	w	x	

18.4.24 ep1_rx_fifo_rdata

Address: 0x4000d91c

RSVD																							
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								

ep1_rx_fifo_rdata

Bits	Name	Type	Reset	Description
31:8	RSVD			
7:0	ep1_rx_fifo_rdata	r	8'h0	

18.4.25 ep2_fifo_config

Address: 0x4000d920

RSVD															
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

RSVD RSVD RSVD RSVD RSVD RSVD RSVD ep2_rx_fifo_overflow ep2_tx_fifo_overflow ep2_tx_fifo_clr ep2_dma_rx_en ep2_dma_tx_en
 ep2_rx_fifo_underflow ep2_tx_fifo_underflow ep2_tx_fifo_clr ep2_rx_fifo_clr ep2_rx_fifo_clr ep2_tx_fifo_clr

Bits	Name	Type	Reset	Description
31:8	RSVD			
7	ep2_rx_fifo_underflow	r	1'b0	Underflow flag of RX FIFO, can be cleared by rx_fifo_clr
6	ep2_rx_fifo_overflow	r	1'b0	Overflow flag of RX FIFO, can be cleared by rx_fifo_clr
5	ep2_tx_fifo_underflow	r	1'b0	Underflow flag of TX FIFO, can be cleared by tx_fifo_clr
4	ep2_tx_fifo_overflow	r	1'b0	Overflow flag of TX FIFO, can be cleared by tx_fifo_clr
3	ep2_rx_fifo_clr	w1c	1'b0	Clear signal of RX FIFO
2	ep2_tx_fifo_clr	w1c	1'b0	Clear signal of TX FIFO
1	ep2_dma_rx_en	r/w	1'b0	Enable signal of dma_rx_req/ack interface
0	ep2_dma_tx_en	r/w	1'b0	Enable signal of dma_tx_req/ack interface

18.4.26 ep2_fifo_status

Address: 0x4000d924

RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD
ep2_rx_fifo_full	ep2_rx_fifo_empty														ep2_rx_fifo_cnt
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

RSVD
 ep2_tx_fifo_full ep2_tx_fifo_empty ep2_tx_fifo_full ep2_tx_fifo_empty RSVD
 ep2_tx_fifo_cnt

Bits	Name	Type	Reset	Description
31	ep2_rx_fifo_full	r	1'b0	RX FIFO full flag
30	ep2_rx_fifo_empty	r	1'b1	RX FIFO empty flag
29:23	RSVD			
22:16	ep2_rx_fifo_cnt	r	7'd0	RX FIFO available count
15	ep2_tx_fifo_full	r	1'b0	TX FIFO full flag
14	ep2_tx_fifo_empty	r	1'b1	TX FIFO empty flag
13:7	RSVD			
6:0	ep2_tx_fifo_cnt	r	7'd64	TX FIFO available count

18.4.27 ep2_tx_fifo_wdata

Address: 0x4000d928

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | |

ep2_tx_fifo_wdata

Bits	Name	Type	Reset	Description
31:8	RSVD			
7:0	ep2_tx_fifo_wdata	w	x	

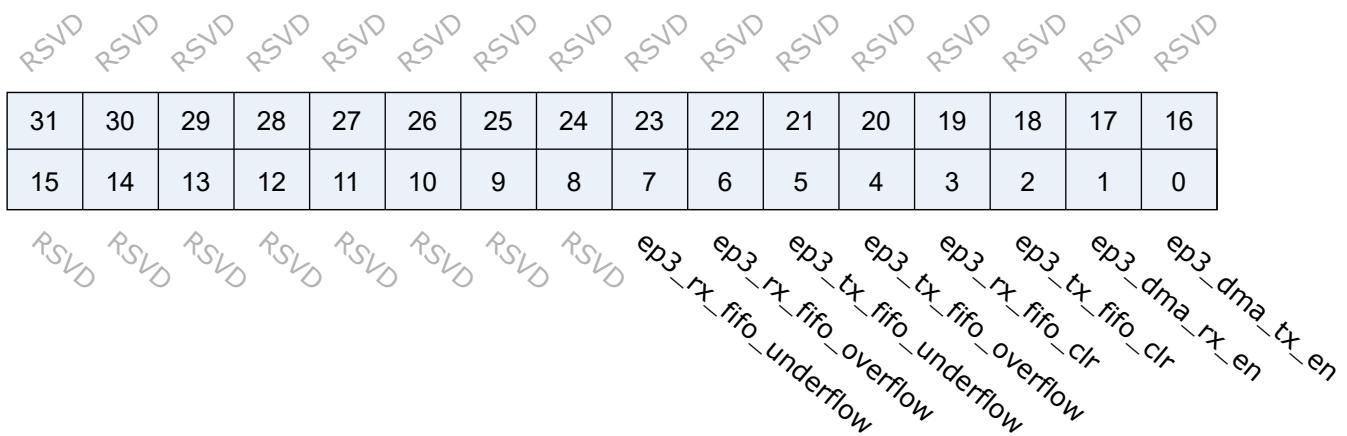
18.4.28 ep2_rx_fifo_rdata

Address: 0x4000d92c

Bits	Name	Type	Reset	Description
31:8	RSVD			
7:0	ep2_rx_fifo_rdata	r	8'h0	

18.4.29 ep3_fifo_config

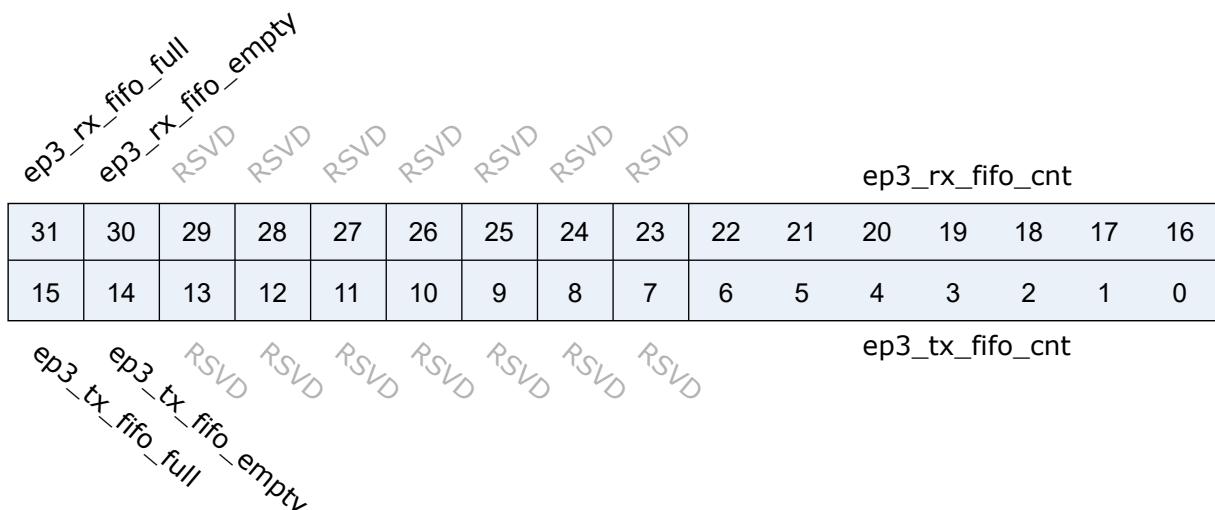
Address: 0x4000d930



Bits	Name	Type	Reset	Description
31:8	RSVD			
7	ep3_rx_fifo_underflow	r	1'b0	Underflow flag of RX FIFO, can be cleared by rx_fifo_clr
6	ep3_rx_fifo_overflow	r	1'b0	Overflow flag of RX FIFO, can be cleared by rx_fifo_clr
5	ep3_tx_fifo_underflow	r	1'b0	Underflow flag of TX FIFO, can be cleared by tx_fifo_clr
4	ep3_tx_fifo_overflow	r	1'b0	Overflow flag of TX FIFO, can be cleared by tx_fifo_clr
3	ep3_rx_fifo_clr	w1c	1'b0	Clear signal of RX FIFO
2	ep3_tx_fifo_clr	w1c	1'b0	Clear signal of TX FIFO
1	ep3_dma_rx_en	r/w	1'b0	Enable signal of dma_rx_req/ack interface
0	ep3_dma_tx_en	r/w	1'b0	Enable signal of dma_tx_req/ack interface

18.4.30 ep3_fifo_status

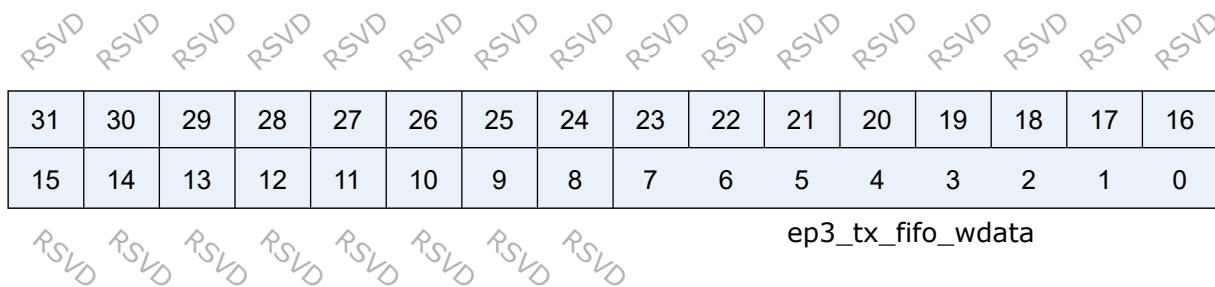
Address: 0x4000d934



Bits	Name	Type	Reset	Description
31	ep3_rx_fifo_full	r	1'b0	RX FIFO full flag
30	ep3_rx_fifo_empty	r	1'b1	RX FIFO empty flag
29:23	RSVD			
22:16	ep3_rx_fifo_cnt	r	7'd0	RX FIFO available count
15	ep3_tx_fifo_full	r	1'b0	TX FIFO full flag
14	ep3_tx_fifo_empty	r	1'b1	TX FIFO empty flag
13:7	RSVD			
6:0	ep3_tx_fifo_cnt	r	7'd64	TX FIFO available count

18.4.31 ep3_tx_fifo_wdata

Address: 0x4000d938



Bits	Name	Type	Reset	Description
31:8	RSVD			
7:0	ep3_tx_fifo_wdata	w	x	

18.4.32 ep3_rx_fifo_rdata

Address: 0x4000d93c

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

RSVD RSVD

ep3_rx_fifo_rdata

Bits	Name	Type	Reset	Description
31:8	RSVD			
7:0	ep3_rx_fifo_rdata	r	8'h0	

18.4.33 ep4_fifo_config

Address: 0x4000d940

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

RSVD RSVD

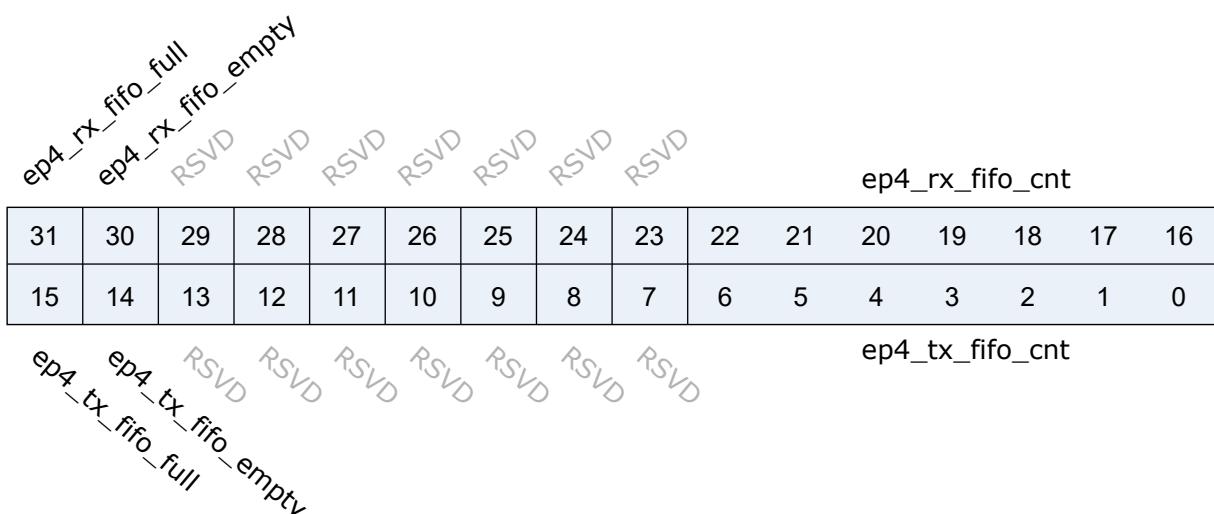
ep4_rx_fifo_underflow ep4_tx_fifo_overflow ep4_tx_fifo_clr ep4_tx_fifo_clr ep4_dma_tx_en ep4_dma_tx_en

Bits	Name	Type	Reset	Description
31:8	RSVD			
7	ep4_rx_fifo_underflow	r	1'b0	Underflow flag of RX FIFO, can be cleared by rx_fifo_clr

Bits	Name	Type	Reset	Description
6	ep4_rx_fifo_overflow	r	1'b0	Overflow flag of RX FIFO, can be cleared by rx_fifo_clr
5	ep4_tx_fifo_underflow	r	1'b0	Underflow flag of TX FIFO, can be cleared by tx_fifo_clr
4	ep4_tx_fifo_overflow	r	1'b0	Overflow flag of TX FIFO, can be cleared by tx_fifo_clr
3	ep4_rx_fifo_clr	w1c	1'b0	Clear signal of RX FIFO
2	ep4_tx_fifo_clr	w1c	1'b0	Clear signal of TX FIFO
1	ep4_dma_rx_en	r/w	1'b0	Enable signal of dma_rx_req/ack interface
0	ep4_dma_tx_en	r/w	1'b0	Enable signal of dma_tx_req/ack interface

18.4.34 ep4_fifo_status

Address: 0x4000d944



Bits	Name	Type	Reset	Description
31	ep4_rx_fifo_full	r	1'b0	RX FIFO full flag
30	ep4_rx_fifo_empty	r	1'b1	RX FIFO empty flag
29:23	RSVD			
22:16	ep4_rx_fifo_cnt	r	7'd0	RX FIFO available count
15	ep4_tx_fifo_full	r	1'b0	TX FIFO full flag
14	ep4_tx_fifo_empty	r	1'b1	TX FIFO empty flag
13:7	RSVD			
6:0	ep4_tx_fifo_cnt	r	7'd64	TX FIFO available count

18.4.35 ep4_tx_fifo_wdata

Address: 0x4000d948

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | |

ep4_tx_fifo_wdata

Bits	Name	Type	Reset	Description
31:8	RSVD			
7:0	ep4_tx_fifo_wdata	w	x	

18.4.36 ep4_rx_fifo_rdata

Address: 0x4000d94c

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | |

ep4_rx_fifo_rdata

Bits	Name	Type	Reset	Description
31:8	RSVD			
7:0	ep4_rx_fifo_rdata	r	8'h0	

18.4.37 ep5_fifo_config

Address: 0x4000d950

RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD								
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
RSVD	ep5_rx_fifo_overflow	ep5_tx_fifo_overflow	ep5_tx_fifo_clr	ep5_rx_fifo_clr	ep5_dma_rx_en	ep5_dma_tx_en									
								ep5_rx_fifo_underrflow	ep5_tx_fifo_underrflow	ep5_tx_fifo_clr	ep5_rx_fifo_clr				

Bits	Name	Type	Reset	Description
31:8	RSVD			
7	ep5_rx_fifo_underflow	r	1'b0	Underflow flag of RX FIFO, can be cleared by rx_fifo_clr
6	ep5_rx_fifo_overflow	r	1'b0	Overflow flag of RX FIFO, can be cleared by rx_fifo_clr
5	ep5_tx_fifo_underflow	r	1'b0	Underflow flag of TX FIFO, can be cleared by tx_fifo_clr
4	ep5_tx_fifo_overflow	r	1'b0	Overflow flag of TX FIFO, can be cleared by tx_fifo_clr
3	ep5_rx_fifo_clr	w1c	1'b0	Clear signal of RX FIFO
2	ep5_tx_fifo_clr	w1c	1'b0	Clear signal of TX FIFO
1	ep5_dma_rx_en	r/w	1'b0	Enable signal of dma_rx_req/ack interface
0	ep5_dma_tx_en	r/w	1'b0	Enable signal of dma_tx_req/ack interface

18.4.38 ep5_fifo_status

Address: 0x4000d954

RSVD	ep5_rx_fifo_full	ep5_rx_fifo_empty	ep5_rx_fifo_cnt	RSVD	RSVD	RSVD	RSVD	ep5_tx_fifo_full	ep5_tx_fifo_empty	ep5_tx_fifo_cnt							
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		

Bits	Name	Type	Reset	Description
31	ep5_rx_fifo_full	r	1'b0	RX FIFO full flag
30	ep5_rx_fifo_empty	r	1'b1	RX FIFO empty flag
29:23	RSVD			
22:16	ep5_rx_fifo_cnt	r	7'd0	RX FIFO available count
15	ep5_tx_fifo_full	r	1'b0	TX FIFO full flag
14	ep5_tx_fifo_empty	r	1'b1	TX FIFO empty flag
13:7	RSVD			
6:0	ep5_tx_fifo_cnt	r	7'd64	TX FIFO available count

18.4.39 ep5_tx_fifo_wdata

Address: 0x4000d958

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | |

ep5_tx_fifo_wdata

Bits	Name	Type	Reset	Description
31:8	RSVD			
7:0	ep5_tx_fifo_wdata	w	x	

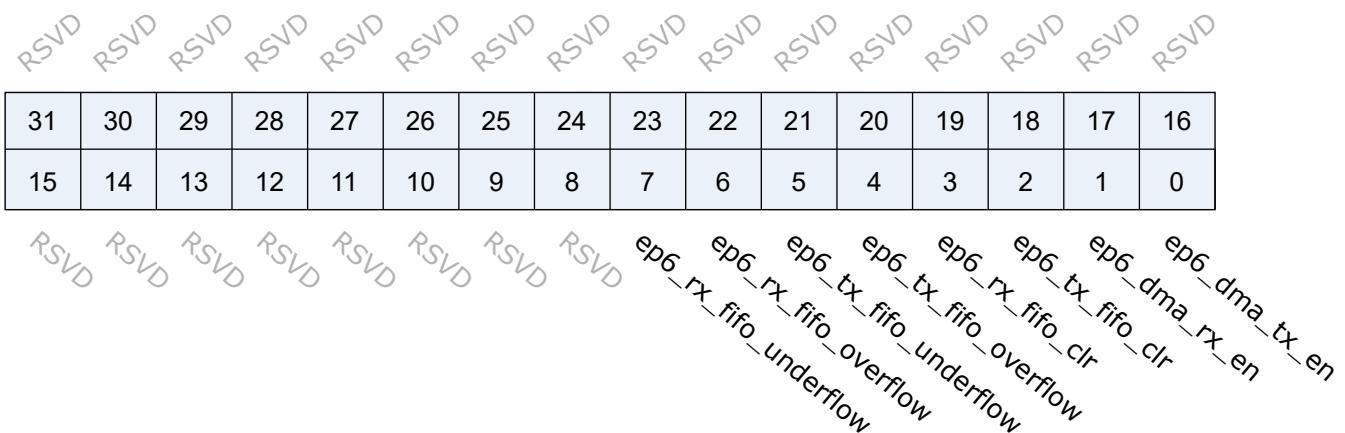
18.4.40 ep5_rx_fifo_rdata

Address: 0x4000d95c

Bits	Name	Type	Reset	Description
31:8	RSVD			
7:0	ep5_rx_fifo_rdata	r	8'h0	

18.4.41 ep6_fifo_config

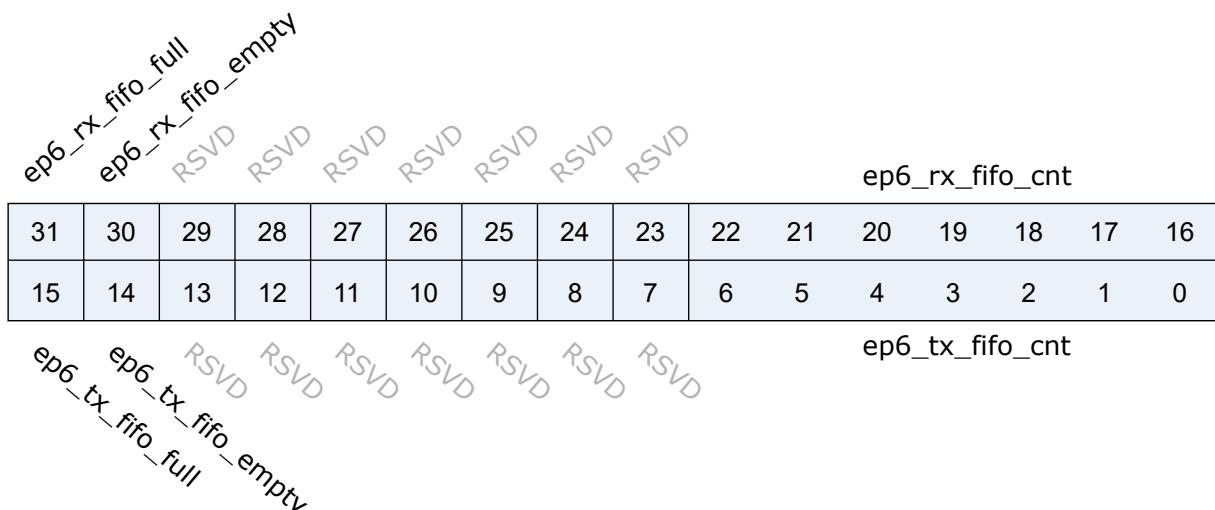
Address: 0x4000d960



Bits	Name	Type	Reset	Description
31:8	RSVD			
7	ep6_rx_fifo_underflow	r	1'b0	Underflow flag of RX FIFO, can be cleared by rx_fifo_clr
6	ep6_rx_fifo_overflow	r	1'b0	Overflow flag of RX FIFO, can be cleared by rx_fifo_clr
5	ep6_tx_fifo_underflow	r	1'b0	Underflow flag of TX FIFO, can be cleared by tx_fifo_clr
4	ep6_tx_fifo_overflow	r	1'b0	Overflow flag of TX FIFO, can be cleared by tx_fifo_clr
3	ep6_rx_fifo_clr	w1c	1'b0	Clear signal of RX FIFO
2	ep6_tx_fifo_clr	w1c	1'b0	Clear signal of TX FIFO
1	ep6_dma_rx_en	r/w	1'b0	Enable signal of dma_rx_req/ack interface
0	ep6_dma_tx_en	r/w	1'b0	Enable signal of dma_tx_req/ack interface

18.4.42 ep6_fifo_status

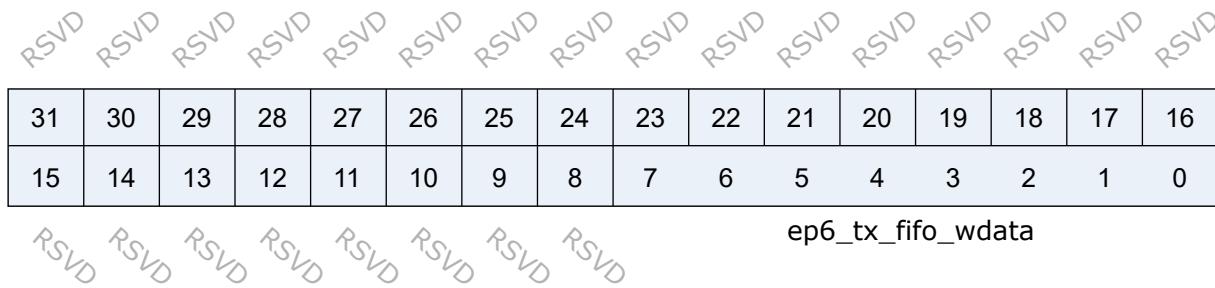
Address: 0x4000d964



Bits	Name	Type	Reset	Description
31	ep6_rx_fifo_full	r	1'b0	RX FIFO full flag
30	ep6_rx_fifo_empty	r	1'b1	RX FIFO empty flag
29:23	RSVD			
22:16	ep6_rx_fifo_cnt	r	7'd0	RX FIFO available count
15	ep6_tx_fifo_full	r	1'b0	TX FIFO full flag
14	ep6_tx_fifo_empty	r	1'b1	TX FIFO empty flag
13:7	RSVD			
6:0	ep6_tx_fifo_cnt	r	7'd64	TX FIFO available count

18.4.43 ep6_tx_fifo_wdata

Address: 0x4000d968



Bits	Name	Type	Reset	Description
31:8	RSVD			
7:0	ep6_tx_fifo_wdata	w	x	

18.4.44 ep6_rx_fifo_rdata

Address: 0x4000d96c

RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD
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		
ep6_rx_fifo_rdata																	

Bits	Name	Type	Reset	Description
31:8	RSVD			
7:0	ep6_rx_fifo_rdata	r	8'h0	

18.4.45 ep7_fifo_config

Address: 0x4000d970

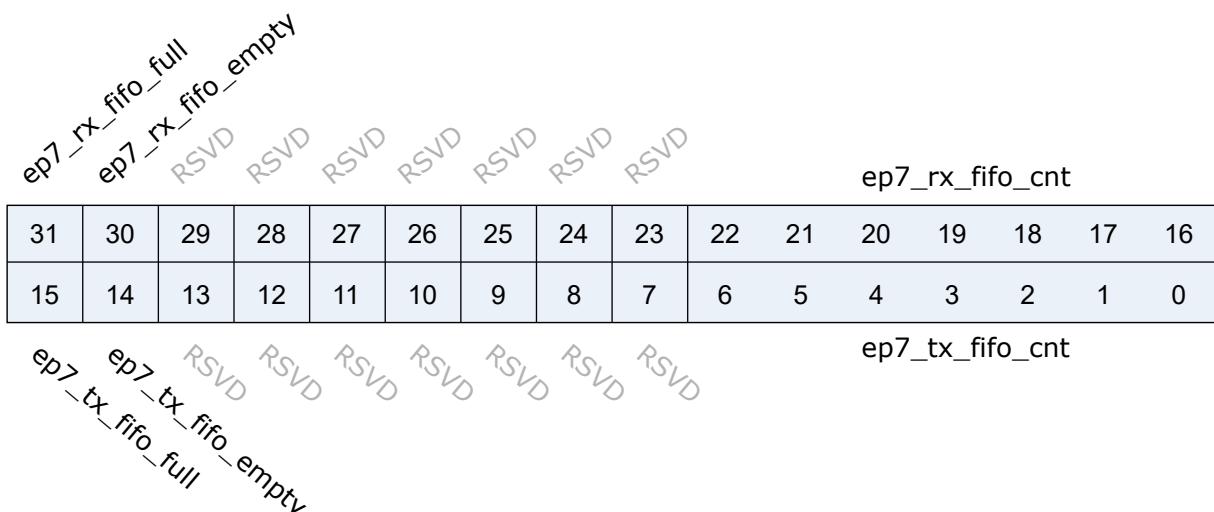
RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
ep7_rx_fifo_underflow																	

Bits	Name	Type	Reset	Description
31:8	RSVD			
7	ep7_rx_fifo_underflow	r	1'b0	Underflow flag of RX FIFO, can be cleared by rx_fifo_clr

Bits	Name	Type	Reset	Description
6	ep7_rx_fifo_overflow	r	1'b0	Overflow flag of RX FIFO, can be cleared by rx_fifo_clr
5	ep7_tx_fifo_underflow	r	1'b0	Underflow flag of TX FIFO, can be cleared by tx_fifo_clr
4	ep7_tx_fifo_overflow	r	1'b0	Overflow flag of TX FIFO, can be cleared by tx_fifo_clr
3	ep7_rx_fifo_clr	w1c	1'b0	Clear signal of RX FIFO
2	ep7_tx_fifo_clr	w1c	1'b0	Clear signal of TX FIFO
1	ep7_dma_rx_en	r/w	1'b0	Enable signal of dma_rx_req/ack interface
0	ep7_dma_tx_en	r/w	1'b0	Enable signal of dma_tx_req/ack interface

18.4.46 ep7_fifo_status

Address: 0x4000d974



Bits	Name	Type	Reset	Description
31	ep7_rx_fifo_full	r	1'b0	RX FIFO full flag
30	ep7_rx_fifo_empty	r	1'b1	RX FIFO empty flag
29:23	RSVD			
22:16	ep7_rx_fifo_cnt	r	7'd0	RX FIFO available count
15	ep7_tx_fifo_full	r	1'b0	TX FIFO full flag
14	ep7_tx_fifo_empty	r	1'b1	TX FIFO empty flag
13:7	RSVD			
6:0	ep7_tx_fifo_cnt	r	7'd64	TX FIFO available count

18.4.47 ep7_tx_fifo_wdata

Address: 0x4000d978

| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 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 | | |

ep7_tx_fifo_wdata

Bits	Name	Type	Reset	Description
31:8	RSVD			
7:0	ep7_tx_fifo_wdata	w	x	

18.4.48 ep7_rx_fifo_rdata

Address: 0x4000d97c

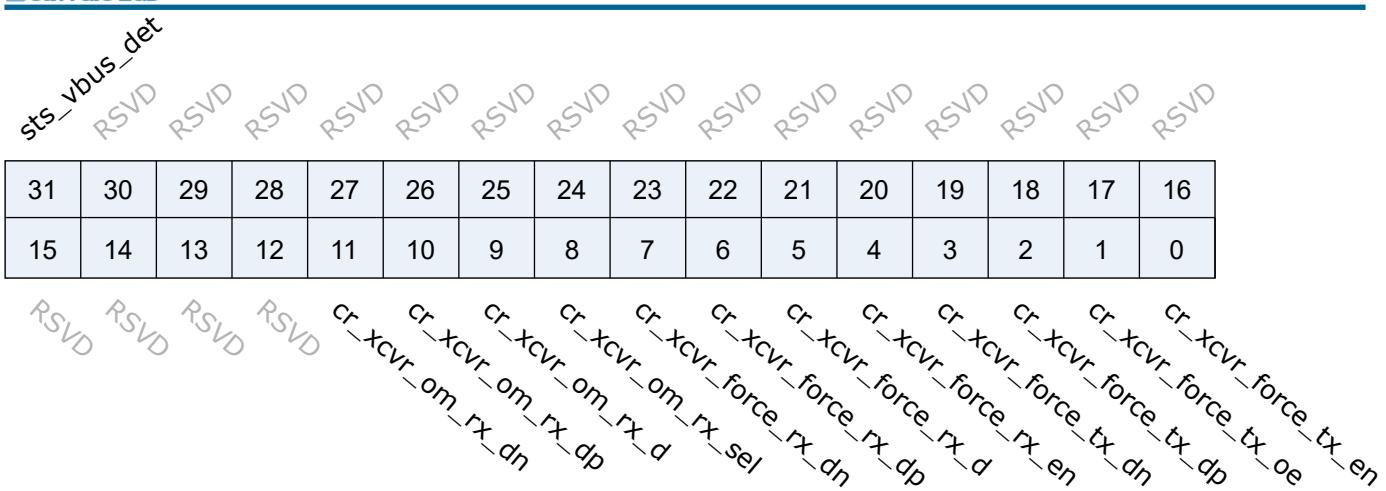
| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |

ep7_rx_fifo_rdata

Bits	Name	Type	Reset	Description
31:8	RSVD			
7:0	ep7_rx_fifo_rdata	r	8'h0	

18.4.49 xcvr_if_config

Address: 0x4000d9fc



Bits	Name	Type	Reset	Description
31	sts_vbus_det	r	1'b0	Transceiver VBUS detection status
30:12	RSVD			
11	cr_xcvr_om_rx_dn	r/w	1'b0	Transceiver RX signals output mode CR value
10	cr_xcvr_om_rx_dp	r/w	1'b1	Transceiver RX signals output mode CR value
9	cr_xcvr_om_rx_d	r/w	1'b1	Transceiver RX signals output mode CR value
8	cr_xcvr_om_rx_sel	r/w	1'b0	Select signal of transceiver RX signals in output mode (tx_oe# = 0) 1'b0: rx_d/dp/dn is directly from transceiver 1'b1: rx_d/dp/dn is controlled by CR (cr_xcvr_om_rx_d/dp/dn)
7	cr_xcvr_force_rx_dn	r/w	1'b0	Transceiver RX signals force mode value
6	cr_xcvr_force_rx_dp	r/w	1'b1	Transceiver RX signals force mode value
5	cr_xcvr_force_rx_d	r/w	1'b1	Transceiver RX signals force mode value
4	cr_xcvr_force_rx_en	r/w	1'b0	Enable signal of transceiver RX signals force mode 1'b0: rx_d/dp/dn is from Transceiver 1'b1: tx_d/dp/dn is controlled by CR (cr_xcvr_force_rx_d/dp/dn)
3	cr_xcvr_force_tx_dn	r/w	1'b0	Transceiver TX signals force mode value
2	cr_xcvr_force_tx_dp	r/w	1'b1	Transceiver TX signals force mode value
1	cr_xcvr_force_tx_oe	r/w	1'b0	Transceiver TX signals force mode value
0	cr_xcvr_force_tx_en	r/w	1'b0	Enable signal of transceiver TX signals force mode 1'b0: tx_oe/dp/dn is controlled by HW 1'b1: tx_oe/dp/dn is controlled by CR (cr_xcvr_force_tx_oe/dp/dn)

Revision history

Table 19.1: Document revision history

Date	Revision	Changes
2020/9/9	1.0	Initial release
2021/3/8	1.1	Modify the GPIO port corresponding to the DAC
2021/9/30	1.2	Update KeyScan register description, introduction of low-power wake-up source