



# BL616/BL618

## 参考手册

*Version: 0.91*

*Copyright @ 2022*

[www.bouffalolab.com](http://www.bouffalolab.com)

# 目录

1 系统和存储器概述 . . . . .	27
1.1 系统架构 . . . . .	27
1.2 处理器 . . . . .	28
1.2.1 主要功能特点 . . . . .	28
1.2.2 扩展功能 . . . . .	29
1.2.3 实现标准 . . . . .	29
1.3 启动 . . . . .	29
1.4 地址映射 . . . . .	30
1.5 中断源 . . . . .	31
1.6 外设概述 . . . . .	33
2 复位和时钟 . . . . .	34
2.1 简介 . . . . .	34
2.2 复位管理 . . . . .	34
2.3 时钟源 . . . . .	35
3 GLB . . . . .	39
3.1 简介 . . . . .	39
3.2 功能描述 . . . . .	39
3.2.1 时钟管理 . . . . .	39
3.2.2 复位管理 . . . . .	39
3.2.3 总线管理 . . . . .	40
3.2.4 内存管理 . . . . .	40
3.2.5 LDO 管理 . . . . .	40
4 GPIO . . . . .	41
4.1 简介 . . . . .	41
4.2 主要特征 . . . . .	41
4.3 GPIO 功能列表 . . . . .	41

4.4	GPIO 输入设置 . . . . .	43
4.5	GPIO 输出设置 . . . . .	43
4.5.1	普通输出模式 . . . . .	43
4.5.2	Set/Clear 输出模式 . . . . .	43
4.6	I/O FIFO . . . . .	44
4.7	I/O 中断 . . . . .	44
4.8	寄存器描述 . . . . .	45
4.8.1	gpio_cfg0 . . . . .	47
4.8.2	gpio_cfg1 . . . . .	48
4.8.3	gpio_cfg2 . . . . .	50
4.8.4	gpio_cfg3 . . . . .	52
4.8.5	gpio_cfg4 . . . . .	54
4.8.6	gpio_cfg5 . . . . .	55
4.8.7	gpio_cfg6 . . . . .	57
4.8.8	gpio_cfg7 . . . . .	59
4.8.9	gpio_cfg8 . . . . .	61
4.8.10	gpio_cfg9 . . . . .	62
4.8.11	gpio_cfg10 . . . . .	64
4.8.12	gpio_cfg11 . . . . .	66
4.8.13	gpio_cfg12 . . . . .	68
4.8.14	gpio_cfg13 . . . . .	70
4.8.15	gpio_cfg14 . . . . .	72
4.8.16	gpio_cfg15 . . . . .	74
4.8.17	gpio_cfg16 . . . . .	76
4.8.18	gpio_cfg17 . . . . .	78
4.8.19	gpio_cfg18 . . . . .	80
4.8.20	gpio_cfg19 . . . . .	82
4.8.21	gpio_cfg20 . . . . .	84
4.8.22	gpio_cfg21 . . . . .	86
4.8.23	gpio_cfg22 . . . . .	88
4.8.24	gpio_cfg23 . . . . .	90
4.8.25	gpio_cfg24 . . . . .	92
4.8.26	gpio_cfg25 . . . . .	94
4.8.27	gpio_cfg26 . . . . .	96
4.8.28	gpio_cfg27 . . . . .	98
4.8.29	gpio_cfg28 . . . . .	100
4.8.30	gpio_cfg29 . . . . .	102
4.8.31	gpio_cfg30 . . . . .	104
4.8.32	gpio_cfg31 . . . . .	106

4.8.33	gpio_cfg32 . . . . .	108
4.8.34	gpio_cfg33 . . . . .	110
4.8.35	gpio_cfg34 . . . . .	112
4.8.36	gpio_cfg128 . . . . .	114
4.8.37	gpio_cfg129 . . . . .	116
4.8.38	gpio_cfg136 . . . . .	116
4.8.39	gpio_cfg137 . . . . .	118
4.8.40	gpio_cfg138 . . . . .	118
4.8.41	gpio_cfg139 . . . . .	121
4.8.42	gpio_cfg141 . . . . .	125
4.8.43	gpio_cfg142 . . . . .	125
4.8.44	gpio_cfg143 . . . . .	126
4.8.45	gpio_cfg144 . . . . .	128
5	ADC . . . . .	129
5.1	简介 . . . . .	129
5.2	主要特点 . . . . .	129
5.3	功能描述 . . . . .	130
5.3.1	ADC 引脚和内部信号 . . . . .	131
5.3.2	ADC 通道 . . . . .	131
5.3.3	ADC 时钟 . . . . .	132
5.3.4	ADC 转换模式 . . . . .	133
5.3.5	ADC 结果 . . . . .	133
5.3.6	ADC 中断 . . . . .	134
5.3.7	ADC FIFO . . . . .	134
5.3.8	ADC 设置流程 . . . . .	135
5.3.9	VBAT 测量 . . . . .	135
5.3.10	TSEN 测量 . . . . .	135
5.4	寄存器描述 . . . . .	136
5.4.1	gpadc_config . . . . .	136
5.4.2	gpadc_dma_rdata . . . . .	137
6	DAC . . . . .	138
6.1	简介 . . . . .	138
6.2	主要特点 . . . . .	138
6.3	功能描述 . . . . .	138
6.3.1	DAC 通道使能 . . . . .	139
6.3.2	DAC 数据格式 . . . . .	139
6.3.3	DAC 输出电压 . . . . .	140
6.3.4	DAC 转换 . . . . .	140
6.3.4.1	CPU 方式 . . . . .	140

6.3.4.2	DMA 方式 . . . . .	141
6.4	寄存器描述 . . . . .	141
6.4.1	gpdac_config . . . . .	142
6.4.2	gpdac_dma_config . . . . .	143
6.4.3	gpdac_dma_wdata . . . . .	143
7	DMA . . . . .	145
7.1	简介 . . . . .	145
7.2	主要特征 . . . . .	145
7.3	功能描述 . . . . .	146
7.3.1	工作原理 . . . . .	146
7.3.2	DMA 通道配置 . . . . .	148
7.3.3	外设支持 . . . . .	148
7.3.4	链表模式 . . . . .	150
7.3.5	DMA 中断 . . . . .	151
7.4	传输模式 . . . . .	152
7.4.1	内存到内存 . . . . .	152
7.4.2	内存到外设 . . . . .	152
7.4.3	外设到内存 . . . . .	152
7.5	寄存器描述 . . . . .	153
7.5.1	DMA_IntStatus . . . . .	154
7.5.2	DMA_IntTCStatus . . . . .	155
7.5.3	DMA_IntTCClear . . . . .	155
7.5.4	DMA_IntErrorStatus . . . . .	155
7.5.5	DMA_IntErrClr . . . . .	156
7.5.6	DMA_RawIntTCStatus . . . . .	156
7.5.7	DMA_RawIntErrorStatus . . . . .	157
7.5.8	DMA_EnbldChns . . . . .	157
7.5.9	DMA_SoftBReq . . . . .	158
7.5.10	DMA_SoftSReq . . . . .	158
7.5.11	DMA_SoftLBReq . . . . .	158
7.5.12	DMA_SoftLSReq . . . . .	159
7.5.13	DMA_Config . . . . .	159
7.5.14	DMA_Sync . . . . .	159
7.5.15	DMA_C0SrcAddr . . . . .	160
7.5.16	DMA_C0DstAddr . . . . .	160
7.5.17	DMA_C0LLI . . . . .	160
7.5.18	DMA_C0Control . . . . .	161
7.5.19	DMA_C0Config . . . . .	162
7.5.20	DMA_C0RSVD . . . . .	163

7.5.21	DMA_C1SrcAddr . . . . .	164
7.5.22	DMA_C1DstAddr . . . . .	164
7.5.23	DMA_C1LLI . . . . .	165
7.5.24	DMA_C1Control . . . . .	165
7.5.25	DMA_C1Config . . . . .	166
7.5.26	DMA_C1RSVD . . . . .	167
7.5.27	DMA_C2SrcAddr . . . . .	168
7.5.28	DMA_C2DstAddr . . . . .	168
7.5.29	DMA_C2LLI . . . . .	168
7.5.30	DMA_C2Control . . . . .	169
7.5.31	DMA_C2Config . . . . .	170
7.5.32	DMA_C2RSVD . . . . .	171
7.5.33	DMA_C3SrcAddr . . . . .	171
7.5.34	DMA_C3DstAddr . . . . .	172
7.5.35	DMA_C3LLI . . . . .	172
7.5.36	DMA_C3Control . . . . .	172
7.5.37	DMA_C3Config . . . . .	174
7.5.38	DMA_C3RSVD . . . . .	174
8	IR . . . . .	176
8.1	简介 . . . . .	176
8.2	主要特征 . . . . .	176
8.3	功能描述 . . . . .	176
8.3.1	固定协议接收 . . . . .	176
8.3.2	脉冲宽度接收 . . . . .	178
8.3.3	IR 中断 . . . . .	178
8.4	寄存器描述 . . . . .	178
8.4.1	irrx_config . . . . .	179
8.4.2	irrx_int_sts . . . . .	180
8.4.3	irrx_pw_config . . . . .	181
8.4.4	irrx_data_count . . . . .	181
8.4.5	irrx_data_word0 . . . . .	181
8.4.6	irrx_data_word1 . . . . .	182
8.4.7	ir_fifo_config_0 . . . . .	182
8.4.8	ir_fifo_config_1 . . . . .	183
8.4.9	ir_fifo_rdata . . . . .	183
9	SPI . . . . .	184
9.1	简介 . . . . .	184
9.2	主要特征 . . . . .	184

9.3 功能描述 . . . . .	185
9.3.1 时钟控制 . . . . .	185
9.3.2 主设备持续传输模式 . . . . .	186
9.3.3 主从设备传输接收数据 . . . . .	186
9.3.4 接收忽略功能 . . . . .	186
9.3.5 濾波功能 . . . . .	187
9.3.6 可配置 MSB/LSB 传输 . . . . .	187
9.3.7 可调整字节传输顺序 . . . . .	188
9.3.8 从模式超时机制 . . . . .	188
9.3.9 I/O 传输模式 . . . . .	188
9.3.10 DMA 传输模式 . . . . .	188
9.3.11 SPI 中断 . . . . .	188
9.4 寄存器描述 . . . . .	189
9.4.1 spi_config . . . . .	189
9.4.2 spi_int_sts . . . . .	191
9.4.3 spi_bus_busy . . . . .	192
9.4.4 spi_prd_0 . . . . .	193
9.4.5 spi_prd_1 . . . . .	193
9.4.6 spi_rxd_ignr . . . . .	193
9.4.7 spi_sto_value . . . . .	194
9.4.8 spi_fifo_config_0 . . . . .	195
9.4.9 spi_fifo_config_1 . . . . .	195
9.4.10 spi_fifo_wdata . . . . .	196
9.4.11 spi_fifo_rdata . . . . .	197
9.4.12 backup_io_en . . . . .	197
10 UART . . . . .	198
10.1 简介 . . . . .	198
10.2 主要特征 . . . . .	198
10.3 功能描述 . . . . .	199
10.3.1 数据格式描述 . . . . .	199
10.3.2 基本架构 . . . . .	200
10.3.3 发送器 . . . . .	200
10.3.4 接收器 . . . . .	201
10.3.5 波特率设定 . . . . .	201
10.3.6 濾波 . . . . .	202
10.3.7 自动波特率检测 . . . . .	202
10.3.8 硬件流控 . . . . .	203
10.3.9 DMA 传输 . . . . .	204
10.3.10 LIN 总线支持 . . . . .	204

10.3.11 RS485 模式 . . . . .	206
10.4 UART 中断 . . . . .	206
10.4.1 TX/RX 传输完成中断 . . . . .	207
10.4.2 TX/RX FIFO 请求中断 . . . . .	207
10.4.3 RX 超时中断 . . . . .	208
10.4.4 RX 奇偶校验错误中断 . . . . .	208
10.4.5 TX/RX FIFO 溢出中断 . . . . .	208
10.4.6 RX BCR 中断 . . . . .	208
10.4.7 LIN 同步错误中断 . . . . .	208
10.4.8 通用/固定字符模式自动波特率检测中断 . . . . .	209
10.5 寄存器描述 . . . . .	209
10.5.1 utx_config . . . . .	210
10.5.2 urx_config . . . . .	211
10.5.3 uart_bit_prd . . . . .	212
10.5.4 data_config . . . . .	213
10.5.5 utx_ir_position . . . . .	213
10.5.6 urx_ir_position . . . . .	214
10.5.7 urx_rto_timer . . . . .	214
10.5.8 uart_sw_mode . . . . .	214
10.5.9 uart_int_sts . . . . .	215
10.5.10 uart_int_mask . . . . .	216
10.5.11 uart_int_clear . . . . .	217
10.5.12 uart_int_en . . . . .	218
10.5.13 uart_status . . . . .	219
10.5.14 sts_urx_abr_prd . . . . .	219
10.5.15 urx_abr_prd_b01 . . . . .	220
10.5.16 urx_abr_prd_b23 . . . . .	220
10.5.17 urx_abr_prd_b45 . . . . .	221
10.5.18 urx_abr_prd_b67 . . . . .	221
10.5.19 urx_abr_pw_tol . . . . .	221
10.5.20 urx_bcr_int_cfg . . . . .	222
10.5.21 utx_rs485_cfg . . . . .	222
10.5.22 uart_fifo_config_0 . . . . .	223
10.5.23 uart_fifo_config_1 . . . . .	224
10.5.24 uart_fifo_wdata . . . . .	224
10.5.25 uart_fifo_rdata . . . . .	225
11 I2C . . . . .	226
11.1 简介 . . . . .	226
11.2 主要特征 . . . . .	226

11.3 功能描述 . . . . .	226
11.3.1 起始和停止条件 . . . . .	227
11.3.2 数据传输格式 . . . . .	227
11.3.3 仲裁 . . . . .	230
11.4 I2C 时钟设定 . . . . .	230
11.5 I2C 配置流程 . . . . .	230
11.5.1 配置项 . . . . .	230
11.5.2 读写标志位 . . . . .	231
11.5.3 从设备地址 . . . . .	231
11.5.4 从设备寄存器地址 . . . . .	231
11.5.5 从设备寄存器地址长度 . . . . .	231
11.5.6 数据 . . . . .	231
11.5.7 数据长度 . . . . .	231
11.5.8 使能信号 . . . . .	232
11.6 FIFO 管理 . . . . .	232
11.7 DMA 功能 . . . . .	233
11.7.1 DMA 发送流程 . . . . .	233
11.7.2 DMA 接收流程 . . . . .	233
11.8 中断 . . . . .	234
11.9 寄存器描述 . . . . .	234
11.9.1 i2c_config . . . . .	235
11.9.2 i2c_int_sts . . . . .	236
11.9.3 i2c_sub_addr . . . . .	237
11.9.4 i2c_bus_busy . . . . .	238
11.9.5 i2c_prd_start . . . . .	238
11.9.6 i2c_prd_stop . . . . .	239
11.9.7 i2c_prd_data . . . . .	239
11.9.8 i2c_fifo_config_0 . . . . .	240
11.9.9 i2c_fifo_config_1 . . . . .	240
11.9.10 i2c_fifo_wdata . . . . .	241
11.9.11 i2c_fifo_rdata . . . . .	241
12 PWM . . . . .	242
12.1 简介 . . . . .	242
12.2 主要特征 . . . . .	242
12.3 功能描述 . . . . .	243
12.3.1 时钟与分频器 . . . . .	243
12.3.2 有效电平 . . . . .	243
12.3.3 脉冲产生原理 . . . . .	243
12.3.4 刹车 . . . . .	244

12.3.5	死区 . . . . .	244
12.3.6	周期与占空比计算 . . . . .	245
12.3.7	PWM 中断 . . . . .	245
12.3.8	ADC 联动 . . . . .	245
12.4	寄存器描述 . . . . .	245
12.4.1	pwm_int_config . . . . .	246
12.4.2	pwm_mc0_config0 . . . . .	247
12.4.3	pwm_mc0_config1 . . . . .	248
12.4.4	pwm_mc0_period . . . . .	249
12.4.5	pwm_mc0_dead_time . . . . .	250
12.4.6	pwm_mc0_ch0_thre . . . . .	251
12.4.7	pwm_mc0_ch1_thre . . . . .	252
12.4.8	pwm_mc0_ch2_thre . . . . .	252
12.4.9	pwm_mc0_ch3_thre . . . . .	252
12.4.10	pwm_mc0_int_sts . . . . .	253
12.4.11	pwm_mc0_int_mask . . . . .	254
12.4.12	pwm_mc0_int_clear . . . . .	255
12.4.13	pwm_mc0_int_en . . . . .	256
13	TIMER . . . . .	257
13.1	简介 . . . . .	257
13.2	主要特征 . . . . .	258
13.3	功能描述 . . . . .	258
13.3.1	通用定时器工作原理 . . . . .	259
13.3.2	看门狗定时器工作原理 . . . . .	261
13.3.3	报警设定 . . . . .	261
13.3.4	看门狗报警 . . . . .	261
13.4	寄存器描述 . . . . .	262
13.4.1	TCCR . . . . .	264
13.4.2	TMR2_0 . . . . .	264
13.4.3	TMR2_1 . . . . .	265
13.4.4	TMR2_2 . . . . .	265
13.4.5	TMR3_0 . . . . .	265
13.4.6	TMR3_1 . . . . .	266
13.4.7	TMR3_2 . . . . .	266
13.4.8	TCR2 . . . . .	266
13.4.9	TCR3 . . . . .	267
13.4.10	TSR2 . . . . .	267
13.4.11	TSR3 . . . . .	268
13.4.12	TIER2 . . . . .	268

13.4.13 TIER3 . . . . .	269
13.4.14 TPLVR2 . . . . .	269
13.4.15 TPLVR3 . . . . .	269
13.4.16 TPLCR2 . . . . .	270
13.4.17 TPLCR3 . . . . .	270
13.4.18 WMER . . . . .	271
13.4.19 WMR . . . . .	271
13.4.20 WVR . . . . .	272
13.4.21 WSR . . . . .	272
13.4.22 TICR2 . . . . .	273
13.4.23 TICR3 . . . . .	273
13.4.24 WICR . . . . .	274
13.4.25 TCER . . . . .	274
13.4.26 TCMR . . . . .	275
13.4.27 TILR2 . . . . .	275
13.4.28 TILR3 . . . . .	276
13.4.29 WCR . . . . .	276
13.4.30 WFAR . . . . .	277
13.4.31 WSAR . . . . .	277
13.4.32 TCVWR2 . . . . .	277
13.4.33 TCVWR3 . . . . .	278
13.4.34 TCVSYN2 . . . . .	278
13.4.35 TCVSYN3 . . . . .	278
13.4.36 TCDR . . . . .	279
13.4.37 GPIO . . . . .	279
13.4.38 GPIO_LAT1 . . . . .	280
13.4.39 GPIO_LAT2 . . . . .	280
14 I2S . . . . .	281
14.1 简介 . . . . .	281
14.2 主要特征 . . . . .	281
14.3 功能描述 . . . . .	282
14.4 功能描述 . . . . .	283
14.4.1 数据格式描述 . . . . .	283
14.4.2 基本架构图 . . . . .	285
14.4.3 时钟源 . . . . .	285
14.4.4 I2S 中断 . . . . .	285
14.5 寄存器描述 . . . . .	286
14.5.1 i2s_config . . . . .	286
14.5.2 i2s_int_sts . . . . .	288

14.5.3 i2s_bclk_config . . . . .	289
14.5.4 i2s_fifo_config_0 . . . . .	289
14.5.5 i2s_fifo_config_1 . . . . .	290
14.5.6 i2s_fifo_wdata . . . . .	291
14.5.7 i2s_fifo_rdata . . . . .	291
14.5.8 i2s_io_config . . . . .	292
<b>15 AudioPWM . . . . .</b>	<b>293</b>
15.1 简介 . . . . .	293
15.2 主要特征 . . . . .	293
15.3 时钟树 . . . . .	293
15.4 功能描述 . . . . .	294
15.4.1 AudioPWM 中断 . . . . .	294
15.4.2 FIFO 格式控制 . . . . .	295
15.4.3 FIFO 的启动与 DMA 搬运 . . . . .	296
15.4.4 音频声道选择器 . . . . .	296
15.4.5 音量控制 . . . . .	297
15.4.6 ZeroDetect . . . . .	297
15.5 配置流程 . . . . .	297
15.6 寄存器描述 . . . . .	297
15.6.1 aupwm_0 . . . . .	298
15.6.2 aupwm_status . . . . .	299
15.6.3 aupwm_s0 . . . . .	300
15.6.4 aupwm_s0_misc . . . . .	302
15.6.5 aupwm_zd_0 . . . . .	303
15.6.6 aupwm_1 . . . . .	303
15.6.7 aupwm_rsvd . . . . .	304
15.6.8 aupwm_test_0 . . . . .	304
15.6.9 aupwm_test_1 . . . . .	305
15.6.10 aupwm_test_2 . . . . .	305
15.6.11 aupwm_test_3 . . . . .	306
15.6.12 aupwm_fifo_ctrl . . . . .	306
15.6.13 aupwm_fifo_status . . . . .	308
15.6.14 aupwm_fifo_data . . . . .	309
<b>16 AudioADC . . . . .</b>	<b>310</b>
16.1 简介 . . . . .	310
16.2 主要特征 . . . . .	310
16.3 时钟树 . . . . .	311
16.4 功能描述 . . . . .	311
16.4.1 PDM 左右声道选择 . . . . .	312

16.4.2 AUADC 中断 . . . . .	312
16.4.3 FIFO 格式控制 . . . . .	312
16.4.4 FIFO 的启动与 DMA 搬运 . . . . .	313
16.5 配置流程 . . . . .	313
16.6 寄存器描述 . . . . .	314
16.6.1 audpdm_top . . . . .	314
16.6.2 audpdm_if . . . . .	315
16.6.3 pdm_adc_0 . . . . .	316
16.6.4 pdm_adc_1 . . . . .	316
16.6.5 pdm_dac_0 . . . . .	317
16.6.6 pdm_pdm_0 . . . . .	317
16.6.7 pdm_dbg_0 . . . . .	318
16.6.8 pdm_dbg_1 . . . . .	318
16.6.9 pdm_dbg_2 . . . . .	319
16.6.10 pdm_adc_s0 . . . . .	319
16.6.11 audadc_ana_cfg1 . . . . .	320
16.6.12 audadc_ana_cfg2 . . . . .	321
16.6.13 audadc_cmd . . . . .	323
16.6.14 audadc_data . . . . .	325
16.6.15 audadc_rx_fifo_ctrl . . . . .	326
16.6.16 audadc_rx_fifo_status . . . . .	329
16.6.17 audadc_rx_fifo_data . . . . .	330
17 Emac . . . . .	331
17.1 简介 . . . . .	331
17.2 主要特征 . . . . .	331
17.3 功能描述 . . . . .	332
17.4 时钟 . . . . .	333
17.5 收发缓冲描述符 (BD, Buffer Descriptor) . . . . .	333
17.6 PHY 交互 . . . . .	334
17.7 编程流程 . . . . .	335
17.7.1 PHY 初始化 . . . . .	335
17.7.2 发送数据帧 . . . . .	336
17.7.3 接收数据帧 . . . . .	336
18 USB . . . . .	338
18.1 简介 . . . . .	338
18.2 主要特征 . . . . .	338
18.3 功能描述 . . . . .	339
18.3.1 USB 使用步骤 . . . . .	339
18.3.2 部分寄存器配置及功能描述 . . . . .	339

18.3.3 USB 枚举阶段中断处理流程 . . . . .	339
19 ISO11898 . . . . .	340
19.1 简介 . . . . .	340
19.2 主要特征 . . . . .	340
19.3 功能介绍 . . . . .	340
19.3.1 发送缓冲区 (TXB) . . . . .	340
19.3.2 接收缓冲区 (RXB, RXFIFO) . . . . .	341
19.3.3 接收过滤器 (ACF) . . . . .	341
19.3.4 位流处理器 (BSP) . . . . .	341
19.3.5 位时序逻辑 (BTL) . . . . .	341
19.3.6 错误管理逻辑 (EML) . . . . .	341
19.4 功能描述 . . . . .	341
19.4.1 模式 . . . . .	341
19.4.1.1 自测模式 . . . . .	341
19.4.1.2 静默模式 . . . . .	342
19.4.1.3 复位模式 . . . . .	342
19.4.2 发送处理 . . . . .	343
19.4.2.1 发送流程 . . . . .	343
19.4.2.2 终止发送 . . . . .	344
19.4.2.3 自发自收 . . . . .	344
19.4.2.4 注意点 . . . . .	344
19.4.3 接收处理 . . . . .	344
19.4.3.1 接收流程 . . . . .	344
19.4.3.2 消息数量 . . . . .	345
19.4.3.3 接收缓冲区 . . . . .	345
19.4.4 标识符过滤 . . . . .	345
19.4.4.1 单滤波器配置 . . . . .	345
19.4.4.2 双滤波器配置 . . . . .	347
19.4.5 出错管理 . . . . .	350
19.4.5.1 仲裁失败 . . . . .	350
19.4.5.2 错误捕获 . . . . .	352
19.4.5.3 接收错误计数器 (RXERR) . . . . .	353
19.4.5.4 发送错误计数器 (TXERR) . . . . .	353
19.4.5.5 错误限值设定 . . . . .	354
19.4.6 位时序 . . . . .	354
19.4.6.1 波特率分频器 (BRP) . . . . .	354
19.4.6.2 同步跳转宽度 (SJW) . . . . .	355
19.4.6.3 采样 (SAM) . . . . .	355
19.4.6.4 时间段 (TSEG) . . . . .	355

20 CAM . . . . .	356
20.1 简介 . . . . .	356
20.2 主要特征 . . . . .	356
20.3 功能描述 . . . . .	357
20.3.1 DVP(Digital Video Port) 信号与配置 . . . . .	357
20.3.2 YCbCr 格式 . . . . .	357
20.3.3 影片模式/照片模式 . . . . .	358
20.3.4 图像矩形裁剪 . . . . .	358
20.3.5 帧取舍功能 . . . . .	358
20.3.6 行帧同步信号完整性检测 . . . . .	359
20.3.7 缓存图像信息 . . . . .	359
20.3.8 支持多种中断信息(可独立开关配置) . . . . .	359
20.4 寄存器描述 . . . . .	360
20.4.1 dvp2axi_configue . . . . .	361
20.4.2 dvp2axi_addr_start . . . . .	363
20.4.3 dvp2axi_mem_bcnt . . . . .	363
20.4.4 dvp_status_and_error . . . . .	363
20.4.5 dvp2axi_frame_bcnt . . . . .	364
20.4.6 dvp_frame_fifo_pop . . . . .	365
20.4.7 dvp2axi_frame_vld . . . . .	365
20.4.8 dvp2axi_frame_period . . . . .	366
20.4.9 dvp2axi_misc . . . . .	366
20.4.10 dvp2axi_hsync_crop . . . . .	367
20.4.11 dvp2axi_vsync_crop . . . . .	367
20.4.12 dvp2axi_fram_exm . . . . .	368
20.4.13 frame_start_addr0 . . . . .	368
20.4.14 frame_start_addr1 . . . . .	368
20.4.15 frame_start_addr2 . . . . .	369
20.4.16 frame_start_addr3 . . . . .	369
20.4.17 frame_id_sts01 . . . . .	369
20.4.18 frame_id_sts23 . . . . .	370
20.4.19 dvp_debug . . . . .	370
20.4.20 dvp_dummy_reg . . . . .	371
21 MJPEG . . . . .	372
21.1 简介 . . . . .	372
21.2 主要特征 . . . . .	372
21.3 功能描述 . . . . .	373
21.3.1 输入配置 . . . . .	373
21.3.2 量化系数表 . . . . .	373

21.3.3 软件模式和连动模式 . . . . .	373
21.3.4 swap 模式 . . . . .	374
21.3.5 Kick 模式 . . . . .	374
21.3.6 jpg 功能 . . . . .	374
21.3.7 缓存图片信息 . . . . .	374
21.3.8 支持多种中断信息(可独立开关配置) . . . . .	374
<b>22 DBI . . . . .</b>	<b>375</b>
<b>22.1 简介 . . . . .</b>	<b>375</b>
<b>22.2 主要特征 . . . . .</b>	<b>375</b>
<b>22.3 功能描述 . . . . .</b>	<b>376</b>
<b>22.3.1 DBI Type B . . . . .</b>	<b>377</b>
<b>22.3.1.1 写时序 . . . . .</b>	<b>377</b>
<b>22.3.1.2 读时序 . . . . .</b>	<b>379</b>
<b>22.3.1.3 输出 RGB565 . . . . .</b>	<b>379</b>
<b>22.3.1.4 输出 RGB666 . . . . .</b>	<b>380</b>
<b>22.3.1.5 输出 RGB888 . . . . .</b>	<b>381</b>
<b>22.3.2 DBI Type C 3-Line . . . . .</b>	<b>382</b>
<b>22.3.2.1 写时序 . . . . .</b>	<b>382</b>
<b>22.3.2.2 读时序 . . . . .</b>	<b>383</b>
<b>22.3.2.3 输出 RGB565 . . . . .</b>	<b>384</b>
<b>22.3.2.4 输出 RGB666 . . . . .</b>	<b>384</b>
<b>22.3.2.5 输出 RGB888 . . . . .</b>	<b>385</b>
<b>22.3.3 DBI Type C 4-Line . . . . .</b>	<b>385</b>
<b>22.3.3.1 写时序 . . . . .</b>	<b>385</b>
<b>22.3.3.2 读时序 . . . . .</b>	<b>386</b>
<b>22.3.3.3 输出 RGB565 . . . . .</b>	<b>386</b>
<b>22.3.3.4 输出 RGB666 . . . . .</b>	<b>387</b>
<b>22.3.3.5 输出 RGB888 . . . . .</b>	<b>388</b>
<b>22.3.4 QSPI . . . . .</b>	<b>388</b>
<b>22.3.4.1 写时序 . . . . .</b>	<b>388</b>
<b>22.3.4.2 读时序 . . . . .</b>	<b>390</b>
<b>22.3.4.3 1线命令、1线地址、1线数据模式 . . . . .</b>	<b>390</b>
<b>22.3.4.4 1线命令、1线地址、4线数据模式 . . . . .</b>	<b>392</b>
<b>22.3.4.5 1线命令、4线地址、4线数据模式 . . . . .</b>	<b>395</b>
<b>22.3.5 输入像素格式 . . . . .</b>	<b>396</b>
<b>22.3.6 CS 信号拉高释放条件配置 . . . . .</b>	<b>398</b>
<b>22.3.7 中断 . . . . .</b>	<b>398</b>
<b>22.3.8 DMA . . . . .</b>	<b>398</b>

22.4 寄存器描述 . . . . .	398
22.4.1 dbi_config . . . . .	399
22.4.2 qspi_config . . . . .	401
22.4.3 dbi_pix_cnt . . . . .	402
22.4.4 dbi_prd . . . . .	403
22.4.5 dbi_cmd . . . . .	403
22.4.6 dbi_qspi_adr . . . . .	404
22.4.7 dbi_rdata_0 . . . . .	404
22.4.8 dbi_rdata_1 . . . . .	404
22.4.9 dbi_int_sts . . . . .	405
22.4.10 dbi_yuv_rgb_config_0 . . . . .	406
22.4.11 dbi_yuv_rgb_config_1 . . . . .	406
22.4.12 dbi_yuv_rgb_config_2 . . . . .	407
22.4.13 dbi_yuv_rgb_config_3 . . . . .	407
22.4.14 dbi_yuv_rgb_config_4 . . . . .	408
22.4.15 dbi_yuv_rgb_config_5 . . . . .	409
22.4.16 dbi_fifo_config_0 . . . . .	409
22.4.17 dbi_fifo_config_1 . . . . .	410
22.4.18 dbi_fifo_wdata . . . . .	411
22.4.19 dbi_dummy . . . . .	411
23 SDH . . . . .	412
23.1 简介 . . . . .	412
23.2 主要特征 . . . . .	412
23.3 功能描述 . . . . .	412
23.3.1 SDH 总体结构 . . . . .	413
23.3.2 寄存器映射 . . . . .	414
23.3.3 多卡槽支持 . . . . .	415
23.3.4 支持 DMA . . . . .	416
23.3.5 SD 命令生成 . . . . .	416
23.3.6 挂起和恢复机制 . . . . .	417
23.3.7 缓冲区控制 . . . . .	417
23.3.7.1 缓冲区指针的控制 . . . . .	418
23.3.7.2 确定缓冲块长度 . . . . .	418
23.3.7.3 分割大数据传输 . . . . .	418
23.3.8 中断控制寄存器之间的关系 . . . . .	418
23.3.9 硬件框图和定时部分 . . . . .	419
23.3.10 自动 CMD12 . . . . .	419
23.3.11 控制 SDCLK . . . . .	420

23.3.12 高级 DMA . . . . .	420
23.3.12.1 ADMA2 的框图 . . . . .	421
23.3.12.2 数据地址和数据长度要求 . . . . .	422
23.3.12.3 描述符表 . . . . .	423
23.3.12.4 ADMA2 状态 . . . . .	424
23.3.13 测试寄存器 . . . . .	425
23.3.14 块计数 . . . . .	425
23.3.15 采样时钟调谐 . . . . .	425
23.3.16 SD 主机标准寄存器 . . . . .	426
23.3.16.1 SD 主机控制寄存器映射 . . . . .	426
23.3.16.2 配置寄存器类型 . . . . .	427
23.3.16.3 寄存器初始值 . . . . .	428
23.3.16.4 寄存器的保留位 . . . . .	428
24 SDIO . . . . .	429
24.1 简介 . . . . .	429
24.2 主要特征 . . . . .	429
24.3 功能描述 . . . . .	429
24.3.1 SDIO 信号定义 . . . . .	429
24.3.1.1 SDIO 卡类型 . . . . .	429
24.3.1.2 SDIO 卡模式 . . . . .	430
24.3.1.3 信号引脚 . . . . .	430
24.3.2 SDIO 卡初始化 . . . . .	430
24.3.2.1 I/O 卡初始化差异 . . . . .	430
24.3.2.2 IO_SEND_OP_COND 命令 (CMD5) . . . . .	432
24.3.2.3 IO_SEND_OP_COND 响应 (R4) . . . . .	432
24.3.2.4 组合卡的特殊初始化注意事项 . . . . .	434
24.3.3 与 SD 内存规范的差异 . . . . .	434
24.3.3.1 SDIO 命令列表 . . . . .	434
24.3.3.2 不支持的 SD 内存命令 . . . . .	437
24.3.3.3 改进的 R6 响应 . . . . .	438
24.3.3.4 SDIO 复位 . . . . .	439
24.3.3.5 总线宽度 . . . . .	439
24.3.3.6 卡检测电阻 . . . . .	439
24.3.3.7 数据传输块大小 . . . . .	439
24.3.3.8 数据传输中止 . . . . .	439
24.3.4 新的 I/O 读或写命令 . . . . .	440
24.3.4.1 IO_RW_DIRECT 命令 (CMD52) . . . . .	440
24.3.4.2 IO_RW_DIRECT 响应 (R5) . . . . .	441
24.3.4.3 IO_RW_EXTENDED 命令 (CMD53) . . . . .	442

24.3.5 SDIO 卡内部操作 . . . . .	443
24.3.5.1 概述 . . . . .	443
24.3.5.2 寄存器访问时间 . . . . .	445
24.3.5.3 中断 . . . . .	445
24.3.5.4 挂起/恢复 . . . . .	445
24.3.5.5 读等待 . . . . .	445
24.3.5.6 数据传输中的 CMD52 . . . . .	446
24.3.5.7 固定内部映射 . . . . .	446
24.3.5.8 公共 I/O 区 (CIA) . . . . .	446
24.3.5.9 卡通用控制寄存器 (CCCR) . . . . .	446
24.3.5.10 功能基础寄存器 (FBR) . . . . .	446
24.3.5.11 卡信息结构 (CIS) . . . . .	447
24.3.5.12 多功能 SDIO 卡 . . . . .	447
24.3.5.13 使用 CMD53 设置块大小 . . . . .	447
24.3.5.14 总线状态图 . . . . .	447
24.3.6 嵌入式 I/O 代码存储区 (CSA) . . . . .	449
24.3.6.1 CSA 访问 . . . . .	449
24.3.6.2 CSA 数据格式 . . . . .	449
24.3.7 SDIO 中断 . . . . .	449
24.3.7.1 SPI 和 SD 1 位模式中断 . . . . .	449
24.3.7.2 SD 4 位模式中断 . . . . .	450
24.3.7.3 中断清除时间 . . . . .	450
24.3.8 SDIO 挂起/恢复操作 . . . . .	450
24.3.9 SDIO 读取等待操作 . . . . .	450
25 LowPower . . . . .	451
25.1 概述 . . . . .	451
25.2 主要特征 . . . . .	451
25.3 功能描述 . . . . .	452
25.3.1 电源域 . . . . .	452
25.3.2 唤醒源 . . . . .	454
25.3.3 功耗模式 . . . . .	454
25.3.4 IO 保持 . . . . .	455
25.4 寄存器描述 . . . . .	456
25.4.1 PDS_CTL . . . . .	457
25.4.2 PDS_TIME1 . . . . .	459
25.4.3 PDS_INT . . . . .	459
25.4.4 PDS_CTL2 . . . . .	460
25.4.5 PDS_CTL3 . . . . .	461
25.4.6 PDS_CTL4 . . . . .	462

25.4.7 pds_stat . . . . .	464
25.4.8 pds_ram1 . . . . .	465
25.4.9 PDS_CTL5 . . . . .	467
25.4.10 PDS_RAM2 . . . . .	468
25.4.11 pds_gpio_i_set . . . . .	469
25.4.12 pds_gpio_pd_set . . . . .	469
25.4.13 pds_gpio_int . . . . .	471
25.4.14 pds_gpio_stat . . . . .	473
25.4.15 PDS_RAM3 . . . . .	473
25.4.16 PDS_RAM4 . . . . .	474
25.5 寄存器描述 . . . . .	475
25.5.1 HBN_CTL . . . . .	476
25.5.2 HBN_TIME_L . . . . .	477
25.5.3 HBN_TIME_H . . . . .	478
25.5.4 RTC_TIME_L . . . . .	478
25.5.5 RTC_TIME_H . . . . .	479
25.5.6 HBN_IRQ_MODE . . . . .	479
25.5.7 HBN_IRQ_STAT . . . . .	480
25.5.8 HBN_IRQ_CLR . . . . .	481
25.5.9 HBN_PIR_CFG . . . . .	481
25.5.10 HBN_PIR_VTH . . . . .	482
25.5.11 HBN_PIR_INTERVAL . . . . .	482
25.5.12 HBN_BOR_CFG . . . . .	483
25.5.13 HBN_GLB . . . . .	484
25.5.14 HBN_SRAM . . . . .	485
25.5.15 HBN_PAD_CTRL_0 . . . . .	486
25.5.16 HBN_PAD_CTRL_1 . . . . .	487
25.5.17 vbat_ldo . . . . .	488
25.5.18 rc32k_ctrl0 . . . . .	490
25.5.19 xtal32k . . . . .	492
26 SEC ENG . . . . .	493
26.1 简介 . . . . .	493
26.1.1 AES 简介 . . . . .	493
26.1.2 SHA 简介 . . . . .	493
26.1.3 CRC 简介 . . . . .	493
26.1.4 GMAC 简介 . . . . .	493
26.2 主要特征 . . . . .	494
26.3 原理描述 . . . . .	494
26.3.1 AES 加解密流程 . . . . .	494

26.3.2 SHA256 的实现 . . . . .	495
26.3.3 GMAC 的原理 . . . . .	496
26.4 功能描述 . . . . .	498
26.4.1 AES 加速器 . . . . .	498
26.4.2 SHA 加速器 . . . . .	499
26.4.3 GMAC(link 模式) . . . . .	500
26.4.4 真随机数发生器 . . . . .	501
27 版本信息 . . . . .	502

for Verimake

1.1 系统框图 . . . . .	28
2.1 系统时钟架构 . . . . .	36
2.2 模块时钟架构 . . . . .	37
2.3 外设时钟架构 . . . . .	38
5.1 ADC 基本框图 . . . . .	130
5.2 ADC 时钟 . . . . .	132
6.1 DAC 基本框图 . . . . .	139
7.1 DMA 框图 . . . . .	147
7.2 外设类型选择 . . . . .	149
7.3 LLI 框架 . . . . .	151
8.1 NEC 逻辑波形 . . . . .	177
8.2 NEC 协议波形 . . . . .	177
8.3 RC5 逻辑波形 . . . . .	177
8.4 RC5 协议波形 . . . . .	178
9.1 SPI 时序图 . . . . .	185
9.2 SPI Ignore 波形图 . . . . .	186
9.3 SPI 滤波波形图 . . . . .	187
10.1 UART 数据格式 . . . . .	199
10.2 UART 基本架构图 . . . . .	200
10.3 UART 采样波形图 . . . . .	201
10.4 UART 滤波波形图 . . . . .	202
10.5 UART 固定字符模式波形图 . . . . .	203
10.6 UART 硬件流控图 . . . . .	203

10.7 一个典型的 LIN 帧 . . . . .	205
10.8 LIN 的 Break 域 . . . . .	205
10.9 LIN 的 Sync 域 . . . . .	205
10.10 LIN 的 ID 域 . . . . .	206
11.1 I2C 起始和停止条件 . . . . .	227
11.2 主发送和从接收的数据格式 . . . . .	227
11.3 主接收和从发送的数据格式 . . . . .	228
11.4 主发送和从接收的时序 . . . . .	228
11.5 主接收和从发送的时序 . . . . .	228
11.6 主发送和从接收的数据格式 (10bit 从地址) . . . . .	229
11.7 主接收和从发送的数据格式 (10bit 从地址) . . . . .	229
11.8 同时传输数据波形示意图 . . . . .	230
13.1 定时器框图 . . . . .	257
13.2 看门狗定时器框图 . . . . .	258
13.3 定时器在 PreLoad 模式下工作时序 . . . . .	260
13.4 Watchdog 工作时序 . . . . .	261
13.5 看门狗报警机制 . . . . .	262
14.1 I2S 基本框图 . . . . .	282
14.2 标准 I2S 数据格式 . . . . .	283
14.3 I2S 左对齐/右对齐数据格式 . . . . .	284
14.4 I2S TDM64 模式六通道录音 . . . . .	285
14.5 I2S 时钟 . . . . .	285
15.1 时钟示意图 . . . . .	294
15.2 模块示意图 . . . . .	294
15.3 Mixer 示意图 . . . . .	296
16.1 时钟示意图 . . . . .	311
16.2 模块框图 . . . . .	311
17.1 EMAC 框图 . . . . .	332
20.1 Cam 框图 . . . . .	356
20.2 FIFO 框架 . . . . .	359
20.3 内存 . . . . .	360
22.1 DBI 基本框图 . . . . .	377
22.2 写时序 . . . . .	378
22.3 读时序 . . . . .	379
22.4 RGB565 输出 . . . . .	380
22.5 RGB666 输出 . . . . .	381

22.6 RGB888 输出 . . . . .	382
22.7 写时序 . . . . .	383
22.8 读时序 . . . . .	383
22.9 RGB565 输出 . . . . .	384
22.10 RGB666 输出 . . . . .	384
22.11 RGB888 输出 . . . . .	385
22.12 写时序 . . . . .	385
22.13 读时序 . . . . .	386
22.14 RGB565 输出 . . . . .	387
22.15 RGB666 输出 . . . . .	387
22.16 RGB888 输出 . . . . .	388
22.17 写时序 . . . . .	389
22.18 读时序 . . . . .	390
22.19 RGB565 输出 . . . . .	391
22.20 RGB666 输出 . . . . .	391
22.21 RGB888 输出 . . . . .	392
22.22 RGB565 输出 . . . . .	393
22.23 RGB666 输出 . . . . .	394
22.24 RGB888 输出 . . . . .	395
22.25 RGB565 输出 . . . . .	395
22.26 RGB666 输出 . . . . .	396
22.27 RGB888 输出 . . . . .	396
22.28 输入像素 RGB 格式 . . . . .	397
23.1 SDH 硬件和驱动结构图 . . . . .	413
23.2 标准寄存器映射分类图 . . . . .	414
23.3 多卡槽控制器的寄存器映射 . . . . .	415
23.4 用于生成 SD 命令的寄存器 . . . . .	416
23.5 挂起和恢复机制 . . . . .	417
23.6 主机控制器框图 . . . . .	419
23.7 通过 SD 总线电源和 SD 时钟启用控制 SDCLK . . . . .	420
23.8 ADMA2 的框图 . . . . .	421
23.9 32 位地址描述符表 . . . . .	423
23.10 ADMA2 的状态图 . . . . .	424
23.11 SD 主机控制寄存器映射 . . . . .	426
23.12 寄存器（和寄存器位字段）类型 . . . . .	427
24.1 2 个 4 位 SDIO 卡信号连接示意图 . . . . .	430
24.2 SDIO 对非 I/O 感知初始化的响应 . . . . .	431
24.3 IO_SEND_OP_COND 命令 . . . . .	432
24.4 SD 模式下的响应 R4 . . . . .	432
24.5 SPI 模式下的响应 R4 . . . . .	433

24.6 改进的 R1 响应 . . . . .	433
24.7 SD 模式命令列表 . . . . .	435
24.8 SPI 模式命令列表 . . . . .	436
24.9 不支持的 SD 内存命令 . . . . .	437
24.10 CMD3 的 R6 响应 . . . . .	438
24.11 SDIO R6 状态位 . . . . .	438
24.12 IO_RW_DIRECT 命令 . . . . .	440
24.13 SD 模式下的 IO_RW_DIRECT 响应 . . . . .	441
24.14 SPI 模式下的 IO_RW_DIRECT 响应 . . . . .	442
24.15 IO_RW_EXTENDED 命令 . . . . .	442
24.16 SDIO 内部映射 . . . . .	444
24.17 总线状态机状态图 . . . . .	448
25.1 低功耗模式 . . . . .	451
26.1 AES 加密流程图 . . . . .	494
26.2 消息认证码流程图 . . . . .	497
26.3 AES 运算模式图 . . . . .	498

## 表格

1.1 启动模式 . . . . .	29
1.2 地址映射 . . . . .	30
1.3 中断分配 . . . . .	31
1.4 外设列表 . . . . .	33
2.1 软件复位功能表 . . . . .	34
4.1 GPIO 功能列表 . . . . .	42
5.1 ADC 内部信号 . . . . .	131
5.2 ADC 外部引脚 . . . . .	131
5.3 ADC 转换结果含义 . . . . .	133
6.1 数据传输格式 . . . . .	140
6.2 内部参考电压输出电压 . . . . .	140
11.1 I2C 引脚 . . . . .	226
14.1 I2S 引脚 . . . . .	283
17.1 传输信号 . . . . .	334
25.1 电源模式 . . . . .	453
25.2 唤醒源 . . . . .	454
27.1 文档版本修改信息 . . . . .	502

## 系统和存储器概述

### 1.1 系统架构

BL616/BL618 系列芯片采用 RISC-V 32-bit CPU，搭载 16KB D-cache 和 32KB I-Cache，CPU 主频高达 320MHz，适用于物联网、嵌入式和人工智能等高性能应用领域。

主系统有以下部分组成：

- 总线接口：**AXI** 总线和 **AHB** 总线
  - CPU 通过 AXI 总线访问存储器
  - CPU 通过 AHB 总线访问外设
  - 具备低功耗，高性能等特性
- 存储器
  - 480KB SRAM 用于数据存储
  - 128KB ROM
  - 支持内嵌高速 PSRAM，扩展 RAM 容量
- 外设
  - 共有 30 个外设模块，其中包含 USB/SDH/SDU/EMAC/DVP/Display 等先进外设

片内集成 Wi-Fi 6/BLE/Zigbee 无线子系统，可以实现多种无线连接和数据传输，提供多样化的连接与传输体验。

BL616/BL618 系统架构如下图所示：

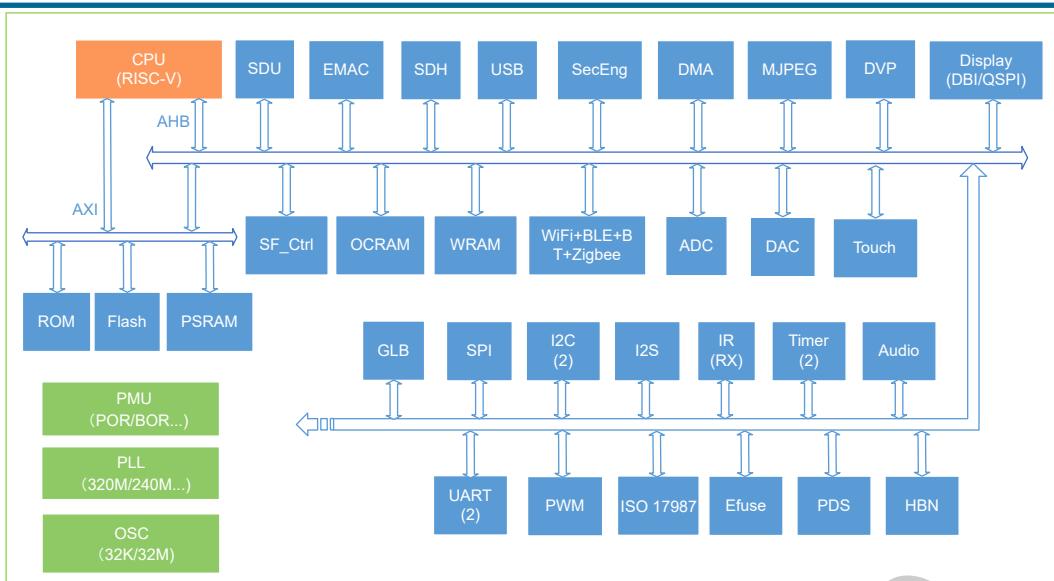


图 1.1: 系统框图

## 1.2 处理器

### 1.2.1 主要功能特点

BL616/BL618 内置一颗 32-bit RISC-V CPU，它采用 5 级流水线结构：取指、译码、执行、内存访问、写回。是一款高性能嵌入式微处理处理器。其主要特征如下：

- 32 位 RISC 处理器；
- 支持 RISC-V RV32IMAFCP 指令集；
- 支持 RISC-V 32/16 位混编指令集；
- 支持 RISC-V 机器模式和用户模式；
- 32 个 32 位整型通用寄存器，32 个 32 位/64 位浮点通用寄存器；
- 整型 5 级/浮点 7 级，单发射，顺序执行流水线；
- 支持 AXI4.0 主设备接口以及 AHB5.0 外设接口；
- 32KB 指令 cache，两路组相连结构；
- 16KB 数据 cache，两路组相连结构；
- 支持非对齐内存访问；
- 双周期硬件乘法器，基 4 硬件除法器；
- 支持 BHT (8K) 和 BTB；
- 支持扩展增强指令集；

- 支持 MCU 特性扩展技术，包括中断处理加速技术、MCU 扩展特性；
- 兼容 RISC-V CLIC 中断标准，支持中断嵌套，外部中断源数量 96 个，有 4 个 Bits 可以用于配置中断优先级；
- 兼容 RISC-V PMP 内存保护标准，8 个区域可配置；
- 支持性能监测单元 (HPM)；
- 支持 RISC-V Debug 协议标准；

### 1.2.2 扩展功能

- 扩展实现了位操作指令，算术运算指令，内存访问增强指令
- 扩展实现了 CACHE 操作指令，同步指令等方便软件人员编程
- 扩展实现了中断投机压栈技术和矢量中断咬尾技术，加速中断响应，中断响应延时为 20 个处理器时钟周期
- 扩展实现了如 NMI，深浅睡眠低功耗模式，低功耗唤醒事件，锁定等 MCU 领域常见的应用需求
- 支持非对齐内存访问，并可软件开关，方便软件人员编程和调试

### 1.2.3 实现标准

该处理器兼容 RISC-V 标准，具体版本为：

- The RISC-V Instruction Set Manual, Volume I: RISC-V User-Level ISA, Version 2.2
- The RISC-V Instruction Set Manual, Volume II: RISC-V Privileged Architecture, Version 1.10
- RISC-V Core-Local Interrupt Controller (CLIC) Version 0.8
- RISC-V External Debug Support Version 0.13.2
- RISC-V “P” Extension Proposal Version 0.9

## 1.3 启动

系统支持从 Flash/UART/USB/SDU 启动，各个启动模式说明如下：

表 1.1: 启动模式

启动引脚	电平	描述
GPIO2	1	从 UART(GPIO21/22)/USB/SDU 启动，该模式主要用于 Flash 烧写或者下载镜像到 RAM 执行 (无线透传场景)
	0	从 Flash 启动应用镜像

## 1.4 地址映射

表 1.2: 地址映射

模块	目标	开始地址	大小	描述
FLASH	Flash	0xA0000000	128MB	应用程序地址空间
PSRAM	pSRAM	0xA8000000	128MB	pSRAM 存储器地址空间 (可选项, 依赖芯片具体型号)
RAM	OCRAM	0x20FC0000	320KB	On Chip RAM 地址空间, 主要是供 CPU 应用程序数据使用的内存
	WRAM	0x21010000	160KB	Wireless RAM 地址空间, 主要是供无线网络数据使用的内存
	HBN RAM	0x20010000	4KB	HBN RAM, 主要用于超低功耗模式下的数据保存
Peripheral	USB	0x20072000	4KB	USB High Speed OTG 控制寄存器
	EMAC	0x20070000	4KB	EMAC 控制寄存器
	SDH	0x20060000	4KB	SDH 控制寄存器
	MJPEG	0x20059000	4KB	MJPEG 图像编码控制寄存器
	DVP	0x20057000	4KB	DVP 摄像头接口控制寄存器
	Efuse	0x20056000	4KB	Efuse 存储控制寄存器
	AUDIO PWM	0x20055000	4KB	Audio PWM 控制寄存器
	PSRAM_Ctrl	0x20052000	4KB	PSRAM 控制寄存器
	HBN	0x2000F000	4KB	深度睡眠控制 (休眠) 寄存器
	PDS	0x2000E000	4KB	睡眠控制 (掉电睡眠) 寄存器
	SDU	0x2000D000	4KB	SDU 控制寄存器
	DMA	0x2000C000	4KB	DMA 控制寄存器
	SF_Ctrl	0x2000B000	4KB	Serial Flash 控制寄存器
	Audio ADC	0x2000AC00	256B	Audio ADC 控制寄存器
	I2S	0x2000AB00	256B	I2S 控制寄存器
	ISO 17987	0x2000AA00	256B	ISO 17987 控制寄存器
	I2C1	0x2000A900	256B	I2C1 控制寄存器
	Display	0x2000A800	256B	Display 控制寄存器
	IRR	0x2000A600	256B	IR Receiver 控制寄存器
	TIMER	0x2000A500	256B	TIMER 控制寄存器
	PWM	0x2000A400	256B	PWM 控制寄存器
	I2C0	0x2000A300	256B	I2C0 控制寄存器
	SPI	0x2000A200	256B	SPI 控制寄存器
	UART1	0x2000A100	256B	UART1 控制寄存器
	UART0	0x2000A000	256B	UART0 控制寄存器
	TZ	0x20005000	4KB	TrustZone 控制寄存器
	SEC_ENG	0x20004000	4KB	安全引擎控制寄存器
	GPIP	0x20002000	1KB	通用 DAC/ADC/ACOMP 接口控制寄存器
	GLB	0x20000000	4KB	全局控制寄存器

表 1.2: 地址映射 (continued)

模块	目标	开始地址	大小	描述
ROM	ROM	0x90000000	128KB	Bootrom 区域地址空间

## 1.5 中断源

BL616/BL618 一共包含 64 个中断源，中断源与对应的中断号如下表所示：

表 1.3: 中断分配

中断源		中断号	描述
BMX	BUS Error	IRQ_NUM_BASE+0	BUS Error Responses Interrupt
	BUS Timeout	IRQ_NUM_BASE+1	BUS Responses Timeout Interrupt
Dispaly	Dispaly	IRQ_NUM_BASE+2	Dispaly All Interrupt
SDU	SDU Software Reset	IRQ_NUM_BASE+3	SDU Reset Triggered by Host
Audio	Audio	IRQ_NUM_BASE+4	Audio All Interrupt
RF	RF Interrupt0	IRQ_NUM_BASE+5	RF Interrupt0
	RF Interrupt1	IRQ_NUM_BASE+6	RF Interrupt1
SDU	SDU Side Interrupt	IRQ_NUM_BASE+7	SDU Side All Interrupt
WiFi	WiFi TBTT	IRQ_NUM_BASE+8	WiFi TBTT Interrupt
SecEng	Group0	IRQ_NUM_BASE+9	Group0 SHA/AES/TRNG/PKA/GMAC Interrupt
	Group1	IRQ_NUM_BASE+10	Group1 SHA/AES/TRNG/PKA/GMAC Interrupt
	Group0 CDET	IRQ_NUM_BASE+11	Group0 CDET Interrupt
	Group1 CDET	IRQ_NUM_BASE+12	Group1 CDET Interrupt
SF Ctrl	Group0	IRQ_NUM_BASE+13	SF_Ctrl Group0 Interrupt
	Group1	IRQ_NUM_BASE+14	SF_Ctrl Group1 Interrupt
DMA	DMA0_ALL	IRQ_NUM_BASE+15	DMA0 ALL Interrupt
DVP0	DVP2BUS0	IRQ_NUM_BASE+16	DVP2BUS0 Interrupt
SDH	SDH All Interrupt	IRQ_NUM_BASE+17	SDH All Interrupt
DVP1	DVP2BUS1	IRQ_NUM_BASE+18	DVP2BUS1 Interrupt
WiFi	TBTT	IRQ_NUM_BASE+19	WiFi TBTT Interrupt
IR	IRRX	IRQ_NUM_BASE+20	IR RX Interrupt
USB	USB	IRQ_NUM_BASE+21	USB Interrupt
Audio	Record	IRQ_NUM_BASE+22	Audio Record All Interrupt
MJPEG	Encoder	IRQ_NUM_BASE+23	MJPEG Encoder All Interrupt
EMAC	EMAC	IRQ_NUM_BASE+24	EMAC Interrupt
ADC	GPADC_DMA	IRQ_NUM_BASE+25	GPADC_DMA Interrupt

表 1.3: 中断分配 (continued)

中断源		中断号	描述
Efuse	Efuse	IRQ_NUM_BASE+26	Efuse Interrupt
SPI	SPI	IRQ_NUM_BASE+27	SPI Interrupt
UART	UART0	IRQ_NUM_BASE+28	UART0 Interrupt
	UART1	IRQ_NUM_BASE+29	UART1 Interrupt
ISO 17987	ISO 17987	IRQ_NUM_BASE+30	ISO 17987 Interrupt
GPIO	GPIO_DMA	IRQ_NUM_BASE+31	GPIO DMA Interrupt
I2C0	I2C0	IRQ_NUM_BASE+32	I2C0 Interrupt
PWM	PWM	IRQ_NUM_BASE+33	PWM Interrupt
TIMER0	TIMER0_CH0	IRQ_NUM_BASE+36	Timer0 Channel 0 Interrupt
	TIMER0_CH1	IRQ_NUM_BASE+37	Timer0 Channel 1 Interrupt
	TIMER0_WDT	IRQ_NUM_BASE+38	Timer0 Watch Dog Interrupt
I2C1	I2C1	IRQ_NUM_BASE+39	I2C1 Interrupt
I2S	I2S	IRQ_NUM_BASE+40	I2S Interrupt
	Reserved	IRQ_NUM_BASE+41	Reserved
	Reserved	IRQ_NUM_BASE+42	Reserved
XTAL	Xtal Ready	IRQ_NUM_BASE+43	Xtal Ready Interrupt
GPIO	GPIO_INT0	IRQ_NUM_BASE+44	GPIO Interrupt
DM	DM	IRQ_NUM_BASE+45	DM Interrupt
BT	BT	IRQ_NUM_BASE+46	BT Interrupt
MAC154	ENH Ack	IRQ_NUM_BASE+47	MAC154 ENH Ack Interrupt
	Others	IRQ_NUM_BASE+48	MAC154 Other Interrupt
	AES	IRQ_NUM_BASE+49	MAC154 AES Interrupt
PDS	PDS	IRQ_NUM_BASE+50	PDS Interrupt
HBN	HBN OUT0	IRQ_NUM_BASE+51	HBN Out 0 Interrupt
	HBN OUT1	IRQ_NUM_BASE+52	HBN Out 1 Interrupt
BOD	BOD	IRQ_NUM_BASE+53	Break Out Detect Interrupt
WiFi	WiFi	IRQ_NUM_BASE+54	WiFi Interrupt
BZ Phy	BZ Phy	IRQ_NUM_BASE+55	BZ Phy Interrupt
BLE	BLE	IRQ_NUM_BASE+56	BLE Interrupt
WiFi	MAC TR Timer	IRQ_NUM_BASE+57	MAC TX&RX Timer Interrupt
	MAC TR MISC	IRQ_NUM_BASE+58	MAC TX&RX Misc Interrupt
	MAC RX Trigger	IRQ_NUM_BASE+59	MAC RX Trigger Interrupt
	MAC TX Trigger	IRQ_NUM_BASE+60	MAC TX Trigger Interrupt
	MAC General	IRQ_NUM_BASE+61	MAC General Interrupt
	MAC Prot	IRQ_NUM_BASE+62	MAC Prot Interrupt

表 1.3: 中断分配 (continued)

中断源	中断号	描述
IPC	IRQ_NUM_BASE+63	MAC IPC Interrupt

注解: 其中 IRQ\_NUM\_BASE 为 16, 中断号 0-15 为 RISC-V 保留中断。

## 1.6 外设概述

表 1.4: 外设列表

外设	数量	备注
GPIO	19/35	QFN40 对应 19 GPIOs, QFN56 对应 35 GPIOs
UART	2	支持 RTS/CTS
SPI	1	支持 Master/Slave 模式
I2C	2	支持 Master 模式
ISO11898	1	AHB 总线
I2S	1	支持 Left-Justified/Right-Justified/Normal I2S/DSP 等数据格式
PWM	4	支持输出极性可调、双门限值设定
Timer	2	支持 FreeRun 模式和 PreLoad 模式
DMA	4	支持 LLI 链表功能
IR	1	支持接收, 协议包括 NEC 和 RC-5, 另外支持以脉冲宽度计数方式接收数据
Audio PWM	1	支持音频播放
Audio ADC	1	支持录音
EMAC	1	支持 10Mbps 和 100Mbps
CAM	2	支持图像矩形裁剪
MJPEG	1	支持任意量化表
DBI	1	支持 Type B/Type C 3-wire/Type C 4-wire, 另外还集成了 QSPI 模式
SDH	1	支持高速 SD 卡
SDU	1	支持 CCCR(function0) 与 function1, 支持 SDU 软复位, function1 有 16 个 port 接收缓冲区
SEC_ENG	1	支持 AES/SHA/GMAC/TRNG
USB	1	USB2.0

## 复位和时钟

### 2.1 简介

芯片内的时钟源：XTAL，PLL，RC。搭配分频等配置送至各模块。

### 2.2 复位管理

表 2.1: 软件复位功能表

BL616	RST_PIN/ Watch Dog/ PDS/ Software Power On (swrst_cfg2[0])	Software Reset (swrst_cfg1)	System Reset (swrst_cfg2[2])/ PDS	CPU Reset (swrst_cfg2[1])/ PDS
cpu(M0)	✓			✓
bus	✓		✓	
glb	✓	swrst_s1[0]		
mix	✓	swrst_s1[1]		
gpip	✓	swrst_s1[2]	✓	
sec_eng	✓	swrst_s1[4]	✓	
tz	✓		✓	
efuse	✓			
dma	✓	swrst_s1[12]	✓	
sdu/usb	✓	swrst_s1[13]		
mm_misc	✓	swrst_s1_ext[1]	✓	
psram_ctrl	✓	swrst_s1_ext[2]	✓	
usb	✓	swrst_s1_ext[3]	✓	
audio play	✓	swrst_s1_ext[5]	✓	
sdh	✓	swrst_s1_ext[6]	✓	
emac	✓	swrst_s1_ext[7]	✓	
dma2	✓	swrst_s1_ext[8]	✓	

表 2.1: 软件复位功能表 (continued)

BL616	RST_PIN/ Watch Dog/ PDS/ Software Power On (swrst_cfg2[0])	Software Reset (swrst_cfg1)	System Reset (swrst_cfg2[2])/ PDS	CPU Reset (swrst_cfg2[1])/ PDS
pds		swrst_s1[14]		
uart0	✓	swrst_s1a[0]	✓	
uart1	✓	swrst_s1a[1]	✓	
spi	✓	swrst_s1a[2]	✓	
i2c0	✓	swrst_s1a[3]	✓	
pwm	✓	swrst_s1a[4]	✓	
timer	✓	swrst_s1a[5]	✓	
irr	✓	swrst_s1a[6]	✓	
cks	✓	swrst_s1a[7]	✓	
dbi	✓	swrst_s1a[8]	✓	
i2c1	✓	swrst_s1a[9]	✓	
ISO11898_- fd	✓	swrst_s1a[10]	✓	
i2s	✓	swrst_s1a[11]	✓	
audio record	✓	swrst_s1a[12]	✓	
wifi	✓	swrst_s2[0]		
ble	✓	swrst_s3[0][2]		

## 2.3 时钟源

时钟源包含：

- XTAL：外部晶振时钟，视系统需求频率可选 24、32、38.4、40MHz
- XTAL32K：外部晶振时钟，频率 32kHz
- RC32K：RC 振荡器时钟，频率 32kHz，提供校准
- RC32M：RC 振荡器时钟，频率 32MHz，提供校准
- PLL：多个 PLL 模块，可以产生若干不同频率的时钟，以满足各个应用场景

时钟控制单元将来自振荡器的时钟分配给内核和外围设备。可通过选择系统时钟源，动态分频器，时钟配置，睡眠使用 32kHz 时钟，以达到低功耗时钟管理。

外围设备时钟包括: Flash、UART、I2C、SPI、PWM、IR-remote、ADC、DAC。

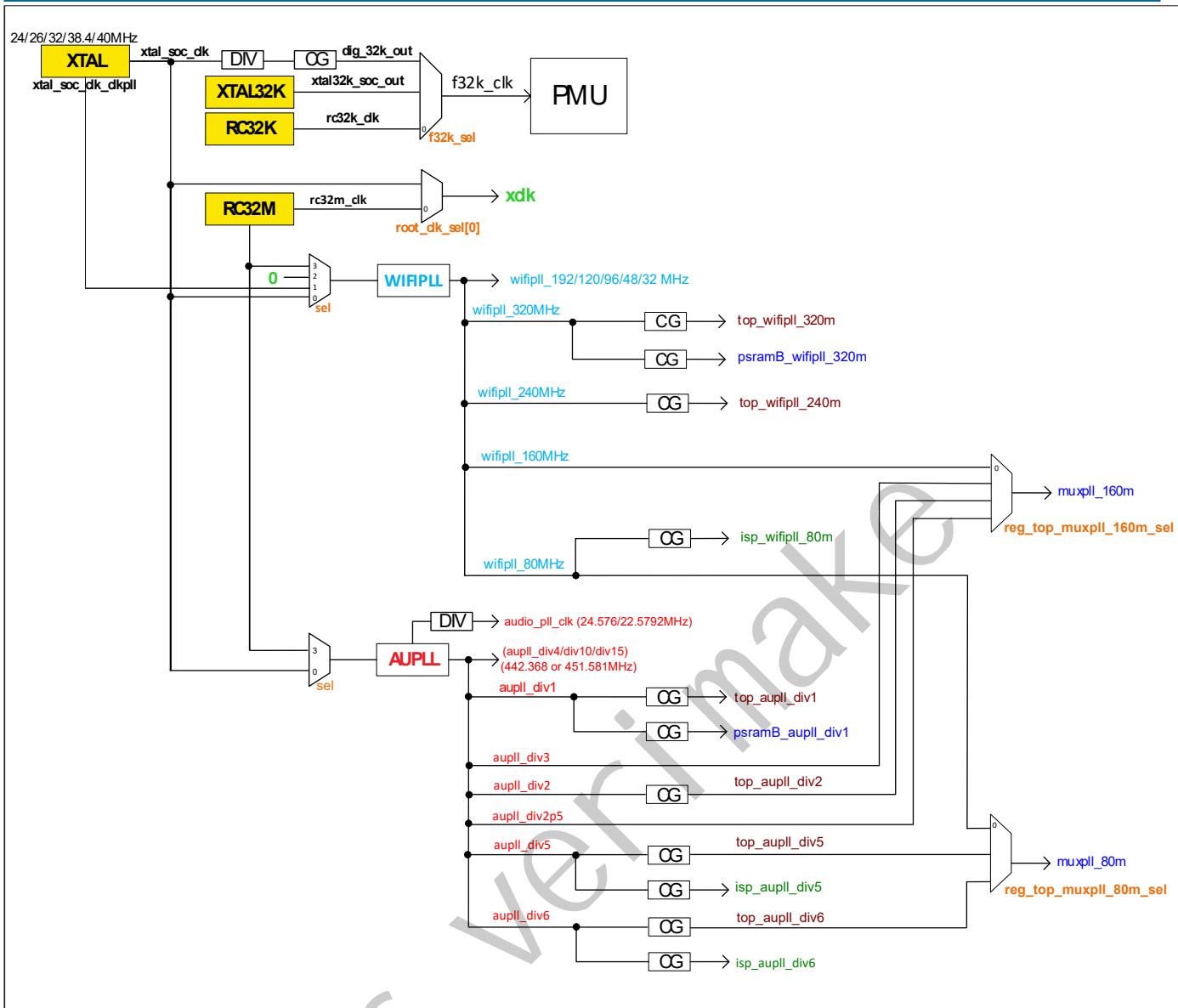


图 2.1: 系统时钟架构

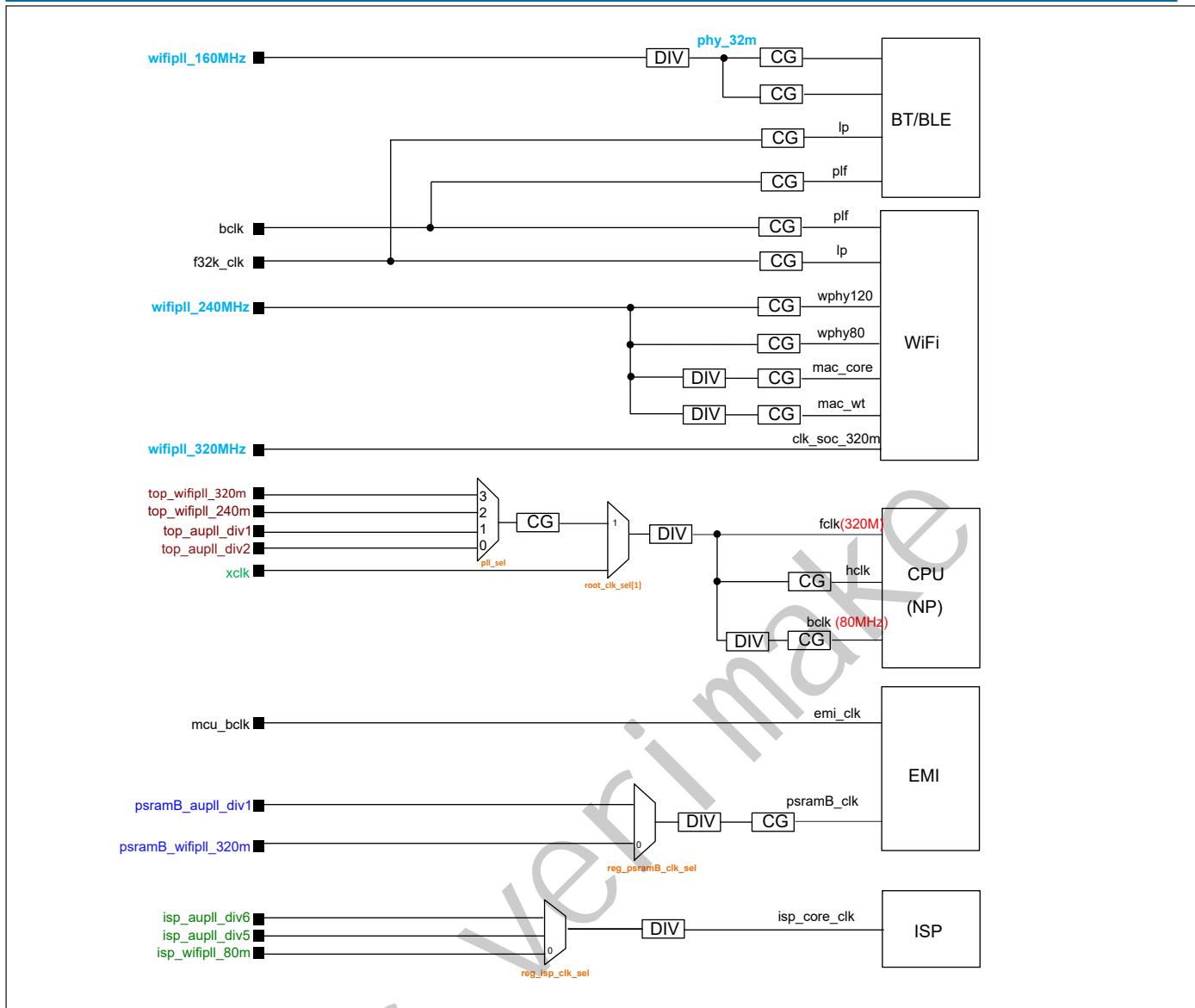


图 2.2: 模块时钟架构

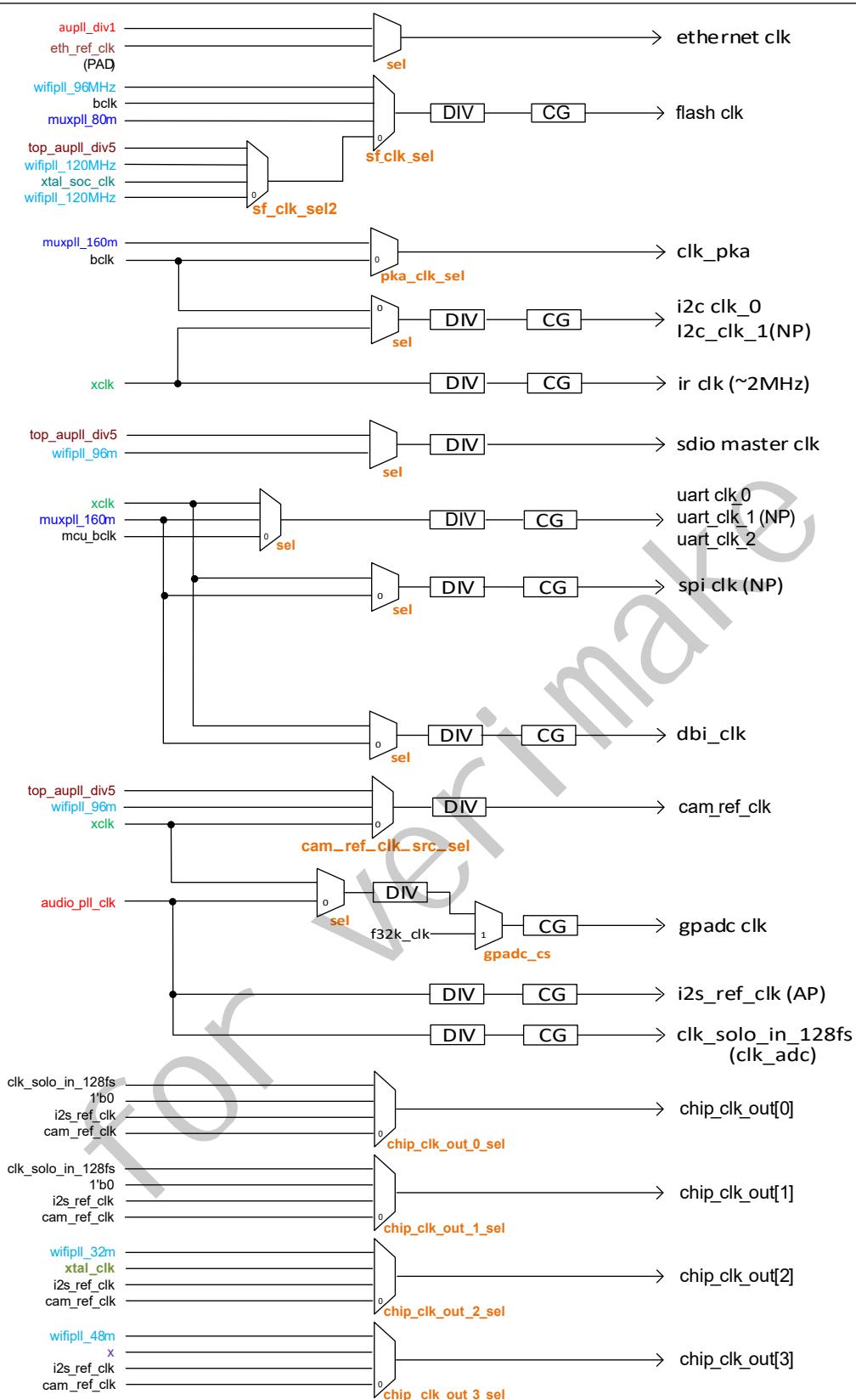


图 2.3: 外设时钟架构

## 3.1 简介

GLB(Global Register) 是芯片通用全局设定模块，主要包含了时钟管理、复位管理、总线管理、内存管理、LDO 管理以及 GPIO 管理等功能。

## 3.2 功能描述

### 3.2.1 时钟管理

时钟管理功能主要用于设定处理器、总线、各个外设的时钟，通过该模块可以设定上述模块工作的时钟源，时钟分频等，同时也可以实现对上述模块时钟的门控，以达到系统低功耗的目的。详细设定可以参考系统时钟相关的章节。

### 3.2.2 复位管理

复位管理提供各个外设模块的单独复位功能以及芯片复位功能。

芯片复位包括：

- CPU 复位：仅仅复位 CPU 模块，程序会重新运行，外设不会被复位
- 系统复位：各个外设和 CPU 会被复位，但是 AON 域的相关寄存器不会被复位
- 上电复位：整个系统包括 AON 域的相关寄存器都会被复位

应用程序可以根据需要，选择使用对应的复位方式。

### 3.2.3 总线管理

提供总线的仲裁设定以及总线出错设定，可以设定在总线出错时候是否产生中断，并提供出错总线地址信息，方便用户调试程序。

### 3.2.4 内存管理

提供各块 memory 区域 size 管理：

- em 管理：WRAM 有一块空间可以作为 EM 使用（默认有 32KB 作为 EM），此功能可以分配指定 size 的空间作为 EM，其余空间仍旧为 WRAM。

提供各个内存模块在芯片系统低功耗模式时的功耗管理，包含两种设定模式：

- retention 模式：在该模式下，内存上的数据可以保存，但是在退出低功耗模式之前，无法读写。
- sleep 模式：在该模式下，内存的数据会丢失，仅用于降低系统功耗。

### 3.2.5 LDO 管理

提供芯片内 LDO 管理：

- LDO 管理：可以对芯片内 LDO 输出电压进行一定调整，以降低功耗。

## 4.1 简介

GPIO(General Purpose I/O Ports) 是通用输入/输出端口，用户可将其与外部硬件设备连接达到控制外部硬件设备的目的。

## 4.2 主要特征

- 最多可达 35 个 I/O 引脚
- 每个 I/O 引脚可配置的功能多达 25 种
- 每个 I/O 引脚都可以配置为上拉、下拉或浮空模式
- 每个 I/O 引脚都可以配置为输入、输出或高阻态模式
- 每个 I/O 引脚的输出模式都有 4 种驱动能力可选择
- 每个 I/O 引脚的输入模式都可以设置是否开启施密特触发器
- 每个 I/O 引脚都支持 9 种外部中断模式
- FIFO 深度为 128 半字
- 可用 DMA 将数据从 RAM 搬至 I/O 引脚用于输出

## 4.3 GPIO 功能列表

GPIO\_CFGxx (其中 xx 表示 GPIO 引脚号) 寄存器的 <reg\_gpio\_xx\_func\_sel> 位用来设置 GPIO 的复用功能，复用功能编号如下表所示：

表 4.1: GPIO 功能列表

编号	功能
0	SDH
1	SPI0
2	FLASH
3	I2S0
4	PDM
5	I2C0
6	I2C1
7	UART0
8	EMAC
9	CAM
10	ANALOG
11	GPIO
12	SDIO
16	PWM0
17	JTAG
18	UART1
19	PWM1
20	SPI1
21	I2S1
22	DBI_B
23	DBI_C
24	QSPI
25	AUPWM
31	CLOCK_OUT

注解：若 GPIO 设置为外设功能，只需将 GPIO\_CFGxx 寄存器的 <reg\_gpio\_xx\_func\_sel> 位设置为外设对应的编号即可。

## 4.4 GPIO 输入设置

通过设置 GPIO\_CFGxx 寄存器，将通用的接口配置为输入模式，过程如下（其中 xx 表示 GPIO 引脚号）：

- 将 `<reg_gpio_xx_ie>` 设置为 1，使能 GPIO 输入模式
- 将 `<reg_gpio_xx_func_sel>` 设置为 11 即进入 SW-GPIO 模式
- 在 SW-GPIO 模式下，可通过 `<reg_gpio_xx_smt>` 设置是否使能施密特触发器，用于波形的整形
- 通过 `<reg_gpio_xx_pu>` 和 `<reg_gpio_xx_pd>` 设置是否使能内部上拉和下拉功能
- 如果需要中断则通过 `<reg_gpio_xx_int_mode_set>` 设定外部中断的类型，此时即可通过 `<reg_gpio_xx_i>` 读取 I/O 引脚的电平值

## 4.5 GPIO 输出设置

通过 GPIO\_CFGxx 寄存器设置 GPIO 不同的输出模式，共支持以下四种模式。

### 4.5.1 普通输出模式

- 将 `<reg_gpio_xx_oe>` 设置为 1，使能 GPIO 输出模式
- 将 `<reg_gpio_xx_func_sel>` 设置为 11，进入 SW-GPIO 模式
- 将 `<reg_gpio_xx_mode>` 设置为 0，使能 I/O 的普通输出功能
- 可通过 `<reg_gpio_xx_pu>` 和 `<reg_gpio_xx_pd>` 设置是否使能内部上拉和下拉功能，此时即可通过 `<reg_gpio_xx_o>` 设置 I/O 引脚的电平值。

### 4.5.2 Set/Clear 输出模式

- 将 `<reg_gpio_xx_oe>` 设置为 1，使能 GPIO 输出模式
- 将 `<reg_gpio_xx_func_sel>` 设置为 11，进入 SW-GPIO 模式
- 将 `<reg_gpio_xx_mode>` 设置为 1，使能 I/O 的 Set/Clear 输出功能
- 可通过 `<reg_gpio_xx_pu>` 和 `<reg_gpio_xx_pd>` 设置是否使能内部上拉和下拉功能

在 Set/Clear 输出模式下，可将 `<reg_gpio_xx_set>` 置 1，使对应 I/O 引脚设为高电平，将 `<reg_gpio_xx_clr>` 置 1 可将对应 I/O 引脚设为低电平，如果同时将 `<reg_gpio_xx_set>` 和 `<reg_gpio_xx_clr>` 置 1 则对应 I/O 引脚为高电平，对 `<reg_gpio_xx_set>` 和 `<reg_gpio_xx_clr>` 写 0 没有作用。

## 4.6 I/O FIFO

I/O FIFO 的深度为 128 半字，GPIO\_CFG143 寄存器中的位 `<gpio_tx_fifo_cnt>` 表示 FIFO 当前可用空间的大小，默认值是 128，每次向 GPIO\_CFG144 寄存器写入一个数值后，`<gpio_tx_fifo_cnt>` 的值都会递减 1，如果减至 0 后继续向 GPIO\_CFG144 寄存器写入数值，且 `<cr_gpio_tx_fer_en>` 为 1 即错误中断被使能，则会产生该中断。

当 GPIO\_CFG142 寄存器的位 `<cr_gpio_tx_en>` 为 1 时，I/O FIFO 的数据被逐个发送到 I/O 引脚，此时 `<gpio_tx_fifo_cnt>` 的值会递增，当递增到大于 `<cr_gpio_tx_fifo_th>` 时，且 `<cr_gpio_tx_fifo_en>` 为 1 即 FIFO 中断被使能，则会产生该中断。

如果 CR\_GPIO\_CFG143 寄存器的位 `<cr_gpio_dma_tx_en>` 为 1，则使能 DMA 发送数据，此时如果 `<cr_gpio_tx_fifo_th>` 小于 `<gpio_tx_fifo_cnt>`，则 DMA 会将数据从设定好的 RAM 中搬运至缓冲区，此时中断标志 `<r_gpio_tx_fifo_int>` 自动被清除。

## 4.7 I/O 中断

I/O 支持多种外部中断，将 GPIO\_CFGxx 寄存器的 `<reg_gpio_xx_int_mask>` 设置为 0，即可使能对应引脚的外部中断功能，`<reg_gpio_xx_int_mode_set>` 用于设置对应引脚的外部中断类型。支持的中断类型如下：

- 同步下降沿中断
  - 以 f32k\_clk 时钟为基准，在每个时钟上升沿采样一次输入引脚电平，若出现一个高电平后紧跟两个低电平，则此时产生同步下降沿中断
- 同步上升沿中断
  - 以 f32k\_clk 时钟为基准，在每个时钟上升沿采样一次输入引脚电平，若出现一个低电平后紧跟两个高电平，则此时产生同步上升沿中断
- 同步低电平中断
  - 以 f32k\_clk 时钟为基准，检测到低电平后，在第三个时钟上升沿处产生同步低电平中断
- 同步高电平中断
  - 以 f32k\_clk 时钟为基准，检测到高电平后，在第三个时钟上升沿处产生同步高电平中断
- 同步双边沿中断
  - 以 f32k\_clk 时钟为基准，若检测到高电平转换至低电平（低电平转换至高电平），会产生下降沿（上升沿）事件，事件产生后在第三个时钟上升沿处产生同步双边沿中断
- 异步下降沿中断
  - 当检测到高电平转换至低电平时，立即触发异步下降沿中断
- 异步上升沿中断
  - 当检测到低电平转换至高电平时，立即触发异步上升沿中断

- 异步低电平中断
  - 以 f32k\_clk 时钟为基准，在每个时钟上升沿采样一次输入引脚电平，若出现连续 3 次都为低电平，则触发异步低电平中断
- 异步高电平中断
  - 以 f32k\_clk 时钟为基准，在每个时钟上升沿采样一次输入引脚电平，若出现连续 3 次都为高电平，则触发异步高电平中断

在中断函数中可以通过 GPIO\_CFGxx 寄存器 <gpio\_xx\_int\_stat> 获取到产生中断的 GPIO 状态，同时可以通过 <reg\_gpio\_xx\_int\_clr> 清除对应的中断标志。

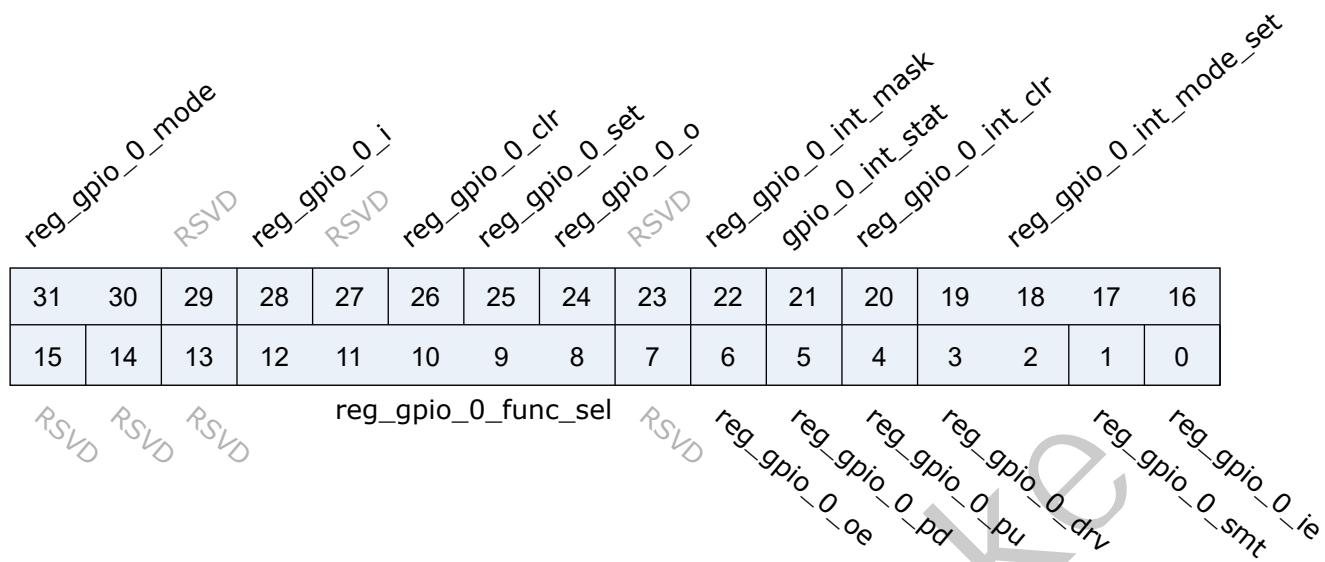
## 4.8 寄存器描述

名称	描述
gpio_cfg0	
gpio_cfg1	
gpio_cfg2	
gpio_cfg3	
gpio_cfg4	
gpio_cfg5	
gpio_cfg6	
gpio_cfg7	
gpio_cfg8	
gpio_cfg9	
gpio_cfg10	
gpio_cfg11	
gpio_cfg12	
gpio_cfg13	
gpio_cfg14	
gpio_cfg15	
gpio_cfg16	
gpio_cfg17	
gpio_cfg18	

名称	描述
gpio_cfg19	
gpio_cfg20	
gpio_cfg21	
gpio_cfg22	
gpio_cfg23	
gpio_cfg24	
gpio_cfg25	
gpio_cfg26	
gpio_cfg27	
gpio_cfg28	
gpio_cfg29	
gpio_cfg30	
gpio_cfg31	
gpio_cfg32	
gpio_cfg33	
gpio_cfg34	
gpio_cfg128	
gpio_cfg129	
gpio_cfg136	
gpio_cfg137	
gpio_cfg138	
gpio_cfg139	
gpio_cfg141	
gpio_cfg142	
gpio_cfg143	
gpio_cfg144	

#### 4.8.1 gpio\_cfg0

地址: 0x200008c4

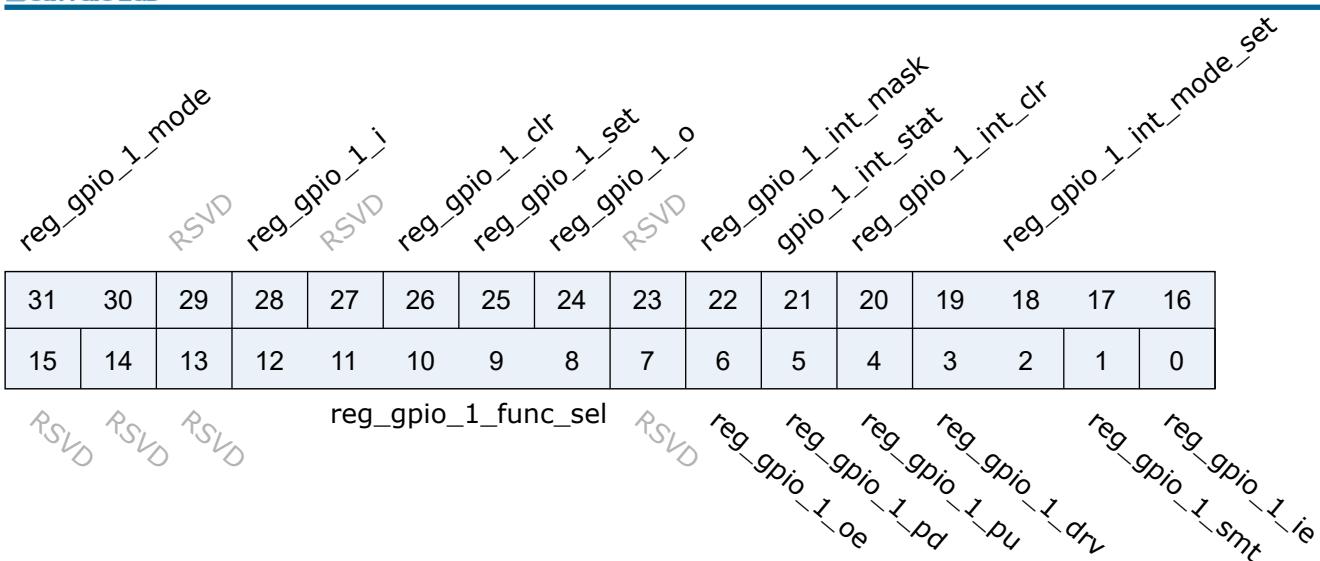


位	名称	权限	复位值	描述
31:30	reg_gpio_0_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_0_i	r	0	
27	RSVD			
26	reg_gpio_0_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_0_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_0_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			

位	名称	权限	复位值	描述
22	reg_gpio_0_int_mask	r/w	1	mask interrupt (1)
21	gpio_0_int_stat	r	0	interrupt status
20	reg_gpio_0_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_0_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_0_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)
7	RSVD			
6	reg_gpio_0_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
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	0	GPIO Input Enable

#### 4.8.2 gpio\_cfg1

地址: 0x200008c8

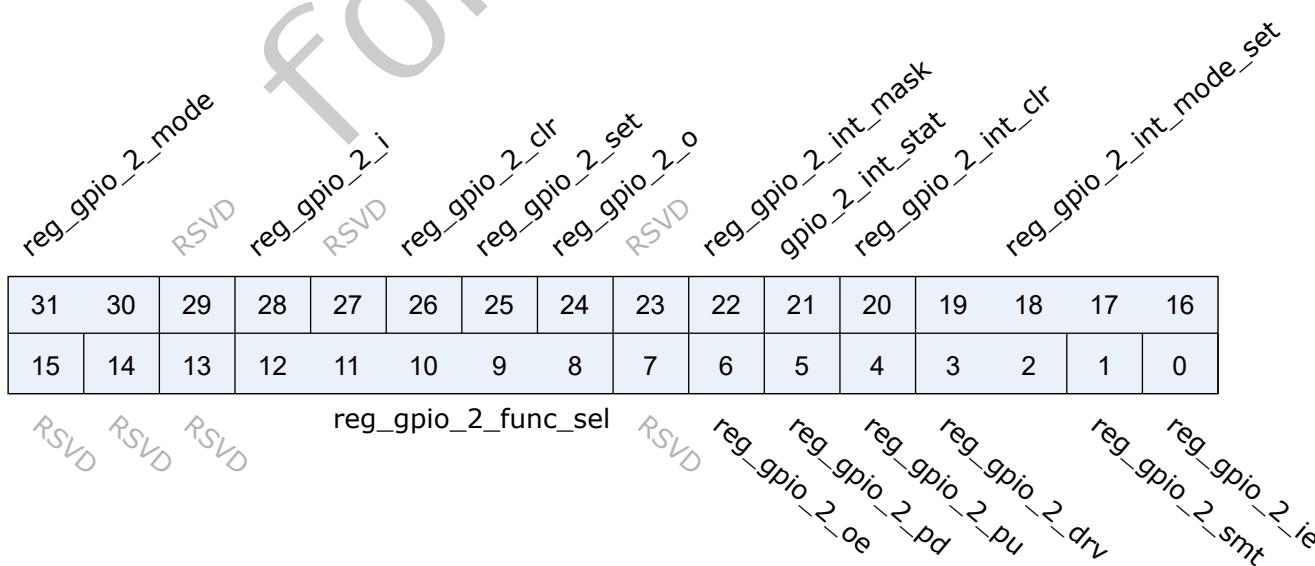


位	名称	权限	复位值	描述
31:30	reg_gpio_1_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_1_i	r	0	
27	RSVD			
26	reg_gpio_1_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_1_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_1_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_1_int_mask	r/w	1	mask interrupt (1)
21	gpio_1_int_stat	r	0	interrupt status

位	名称	权限	复位值	描述
20	reg_gpio_1_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_1_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_1_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)
7	RSVD			
6	reg_gpio_1_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
5	reg_gpio_1_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_1_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_1_drv	r/w	0	GPIO Driving Control
1	reg_gpio_1_smt	r/w	1	GPIO SMT Control
0	reg_gpio_1_ie	r/w	0	GPIO Input Enable

#### 4.8.3 gpio\_cfg2

地址: 0x2000008cc

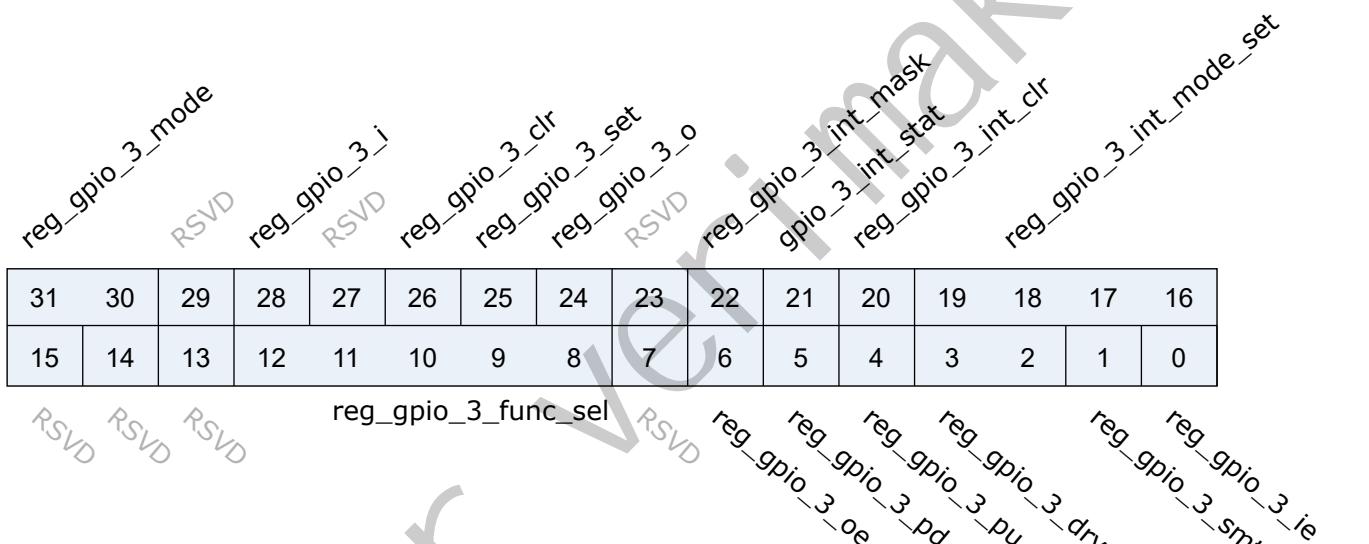


位	名称	权限	复位值	描述
31:30	reg_gpio_2_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_2_i	r	0	
27	RSVD			
26	reg_gpio_2_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_2_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_2_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_2_int_mask	r/w	1	mask interrupt (1)
21	gpio_2_int_stat	r	0	interrupt status
20	reg_gpio_2_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_2_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_2_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)
7	RSVD			

位	名称	权限	复位值	描述
6	reg_gpio_2_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
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	0	GPIO Input Enable

#### 4.8.4 gpio\_cfg3

地址: 0x200008d0

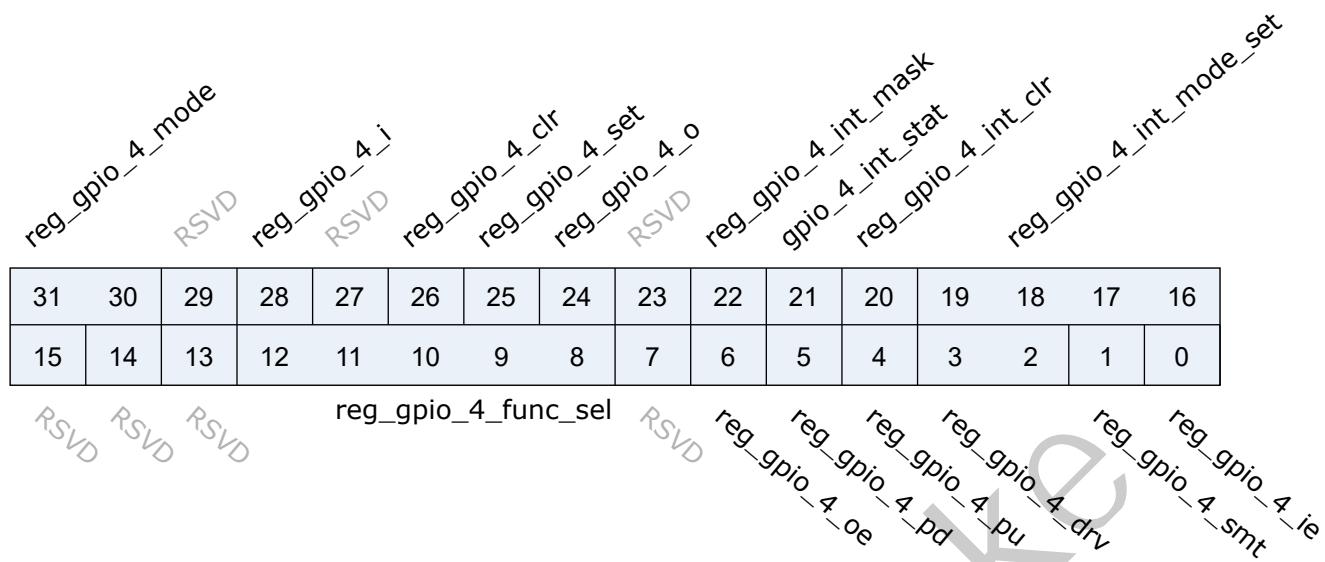


位	名称	权限	复位值	描述
31:30	reg_gpio_3_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			

位	名称	权限	复位值	描述
28	reg_gpio_3_i	r	0	
27	RSVD			
26	reg_gpio_3_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_3_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_3_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_3_int_mask	r/w	1	mask interrupt (1)
21	gpio_3_int_stat	r	0	interrupt status
20	reg_gpio_3_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_3_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_3_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)
7	RSVD			
6	reg_gpio_3_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
5	reg_gpio_3_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_3_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_3_drv	r/w	0	GPIO Driving Control
1	reg_gpio_3_smt	r/w	1	GPIO SMT Control
0	reg_gpio_3_ie	r/w	0	GPIO Input Enable

#### 4.8.5 gpio\_cfg4

地址: 0x200008d4

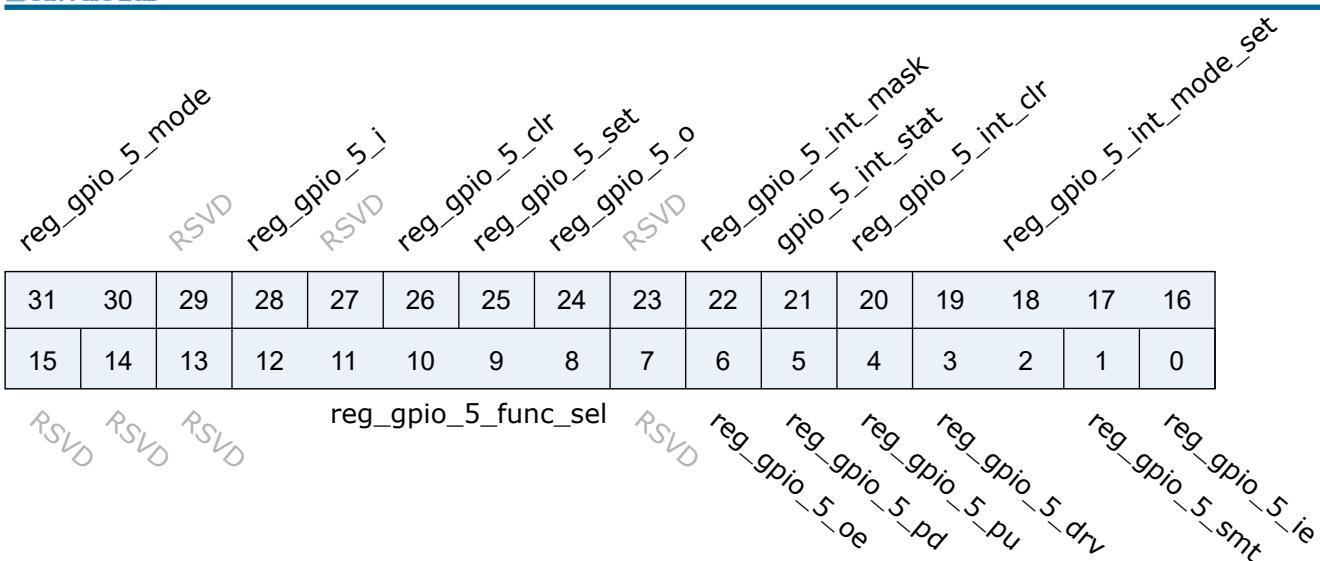


位	名称	权限	复位值	描述
31:30	reg_gpio_4_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_4_i	r	0	
27	RSVD			
26	reg_gpio_4_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_4_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_4_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			

位	名称	权限	复位值	描述
22	reg_gpio_4_int_mask	r/w	1	mask interrupt (1)
21	gpio_4_int_stat	r	0	interrupt status
20	reg_gpio_4_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_4_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_4_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)
7	RSVD			
6	reg_gpio_4_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
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	0	GPIO Input Enable

#### 4.8.6 gpio\_cfg5

地址: 0x200008d8

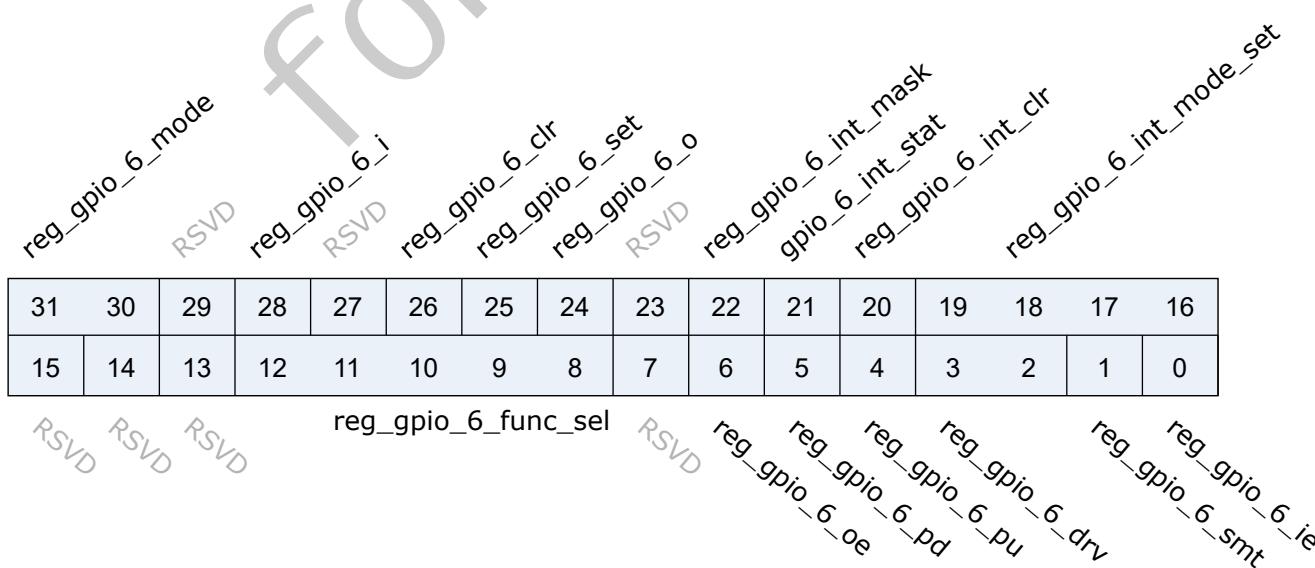


位	名称	权限	复位值	描述
31:30	reg_gpio_5_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_5_i	r	0	
27	RSVD			
26	reg_gpio_5_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_5_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_5_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_5_int_mask	r/w	1	mask interrupt (1)
21	gpio_5_int_stat	r	0	interrupt status

位	名称	权限	复位值	描述
20	reg_gpio_5_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_5_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_5_func_sel	r/w	5'hB	GPIO Function Select (Default : SWGPIO)
7	RSVD			
6	reg_gpio_5_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
5	reg_gpio_5_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_5_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_5_drv	r/w	0	GPIO Driving Control
1	reg_gpio_5_smt	r/w	1	GPIO SMT Control
0	reg_gpio_5_ie	r/w	0	GPIO Input Enable

#### 4.8.7 gpio\_cfg6

地址: 0x2000008dc

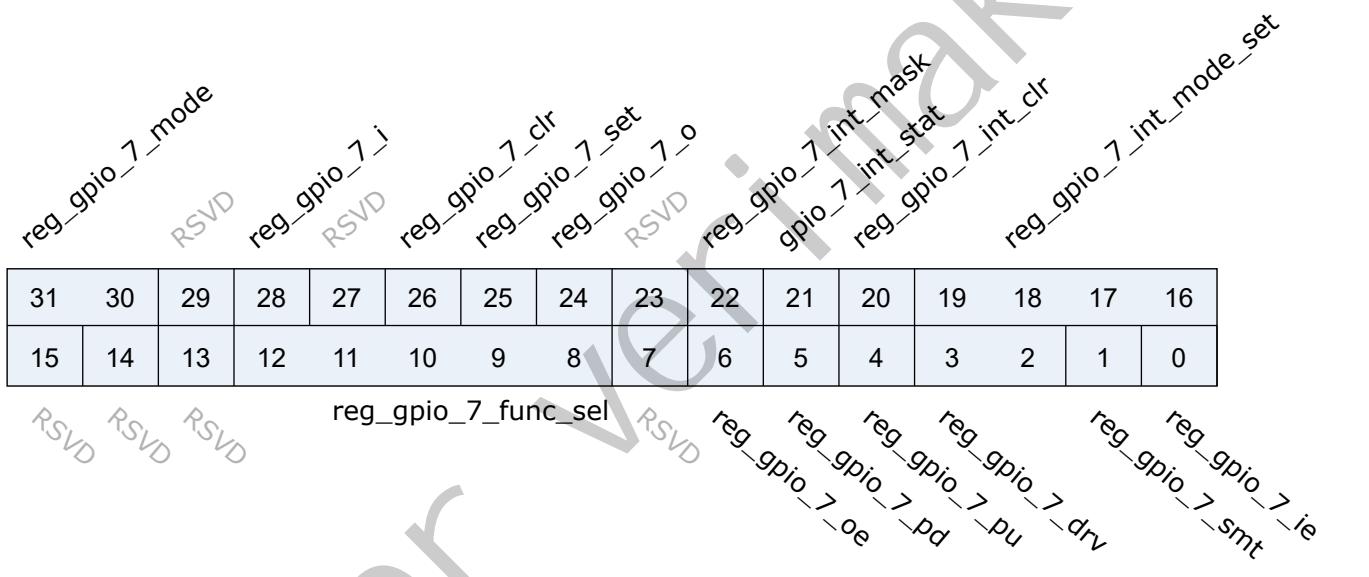


位	名称	权限	复位值	描述
31:30	reg_gpio_6_mode	r/w	0	<p>When GPIO Function Selected to SWGPIO</p> <p>00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value</p> <p>01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr</p> <p>10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o</p> <p>11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr</p>
29	RSVD			
28	reg_gpio_6_i	r	0	
27	RSVD			
26	reg_gpio_6_clr	w1p	0	<p>When SWGPIO @ Set/Clear Mode</p> <p>Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect</p>
25	reg_gpio_6_set	w1p	0	<p>When SWGPIO @ Set/Clear Mode</p> <p>Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect</p>
24	reg_gpio_6_o	r/w	0	<p>When SWGPIO @ Output Value Mode</p> <p>00 : GPIO Value changes according to this value</p> <p>01 : GPIO Value Set by this register and clr by clr_reg</p>
23	RSVD			
22	reg_gpio_6_int_mask	r/w	1	mask interrupt (1)
21	gpio_6_int_stat	r	0	interrupt status
20	reg_gpio_6_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_6_int_mode_set	r/w	0	<p>0000 : sync falling edge trigger</p> <p>0001 : sync rising edge trigger</p> <p>0010 : sync low level trigger</p> <p>0011 : sync high level trigger</p> <p>01xx : sync rising &amp; falling edge trigger</p> <p>1000 : async falling edge trigger</p> <p>1001 : async rising edge trigger</p> <p>1010 : async low level trigger</p> <p>1011 : async high level trigger</p>
15:13	RSVD			
12:8	reg_gpio_6_func_sel	r/w	5'hB	GPIO Function Select (Default : SW-GPIO)
7	RSVD			

位	名称	权限	复位值	描述
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_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	0	GPIO Input Enable

#### 4.8.8 gpio\_cfg7

地址: 0x200008e0

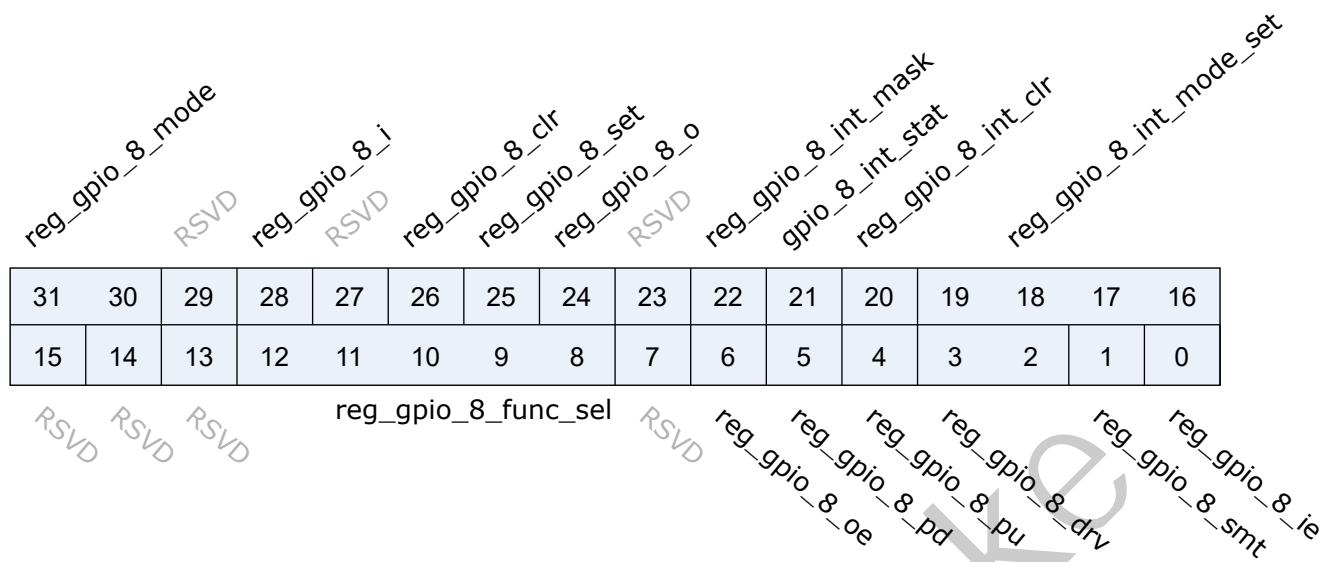


位	名称	权限	复位值	描述
31:30	reg_gpio_7_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			

位	名称	权限	复位值	描述
28	reg_gpio_7_i	r	0	
27	RSVD			
26	reg_gpio_7_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_7_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_7_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_7_int_mask	r/w	1	mask interrupt (1)
21	gpio_7_int_stat	r	0	interrupt status
20	reg_gpio_7_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_7_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_7_func_sel	r/w	5'hB	GPIO Function Select (Default : SW-GPIO)
7	RSVD			
6	reg_gpio_7_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
5	reg_gpio_7_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_7_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_7_drv	r/w	0	GPIO Driving Control
1	reg_gpio_7_smt	r/w	1	GPIO SMT Control
0	reg_gpio_7_ie	r/w	0	GPIO Input Enable

#### 4.8.9 gpio\_cfg8

地址: 0x200008e4

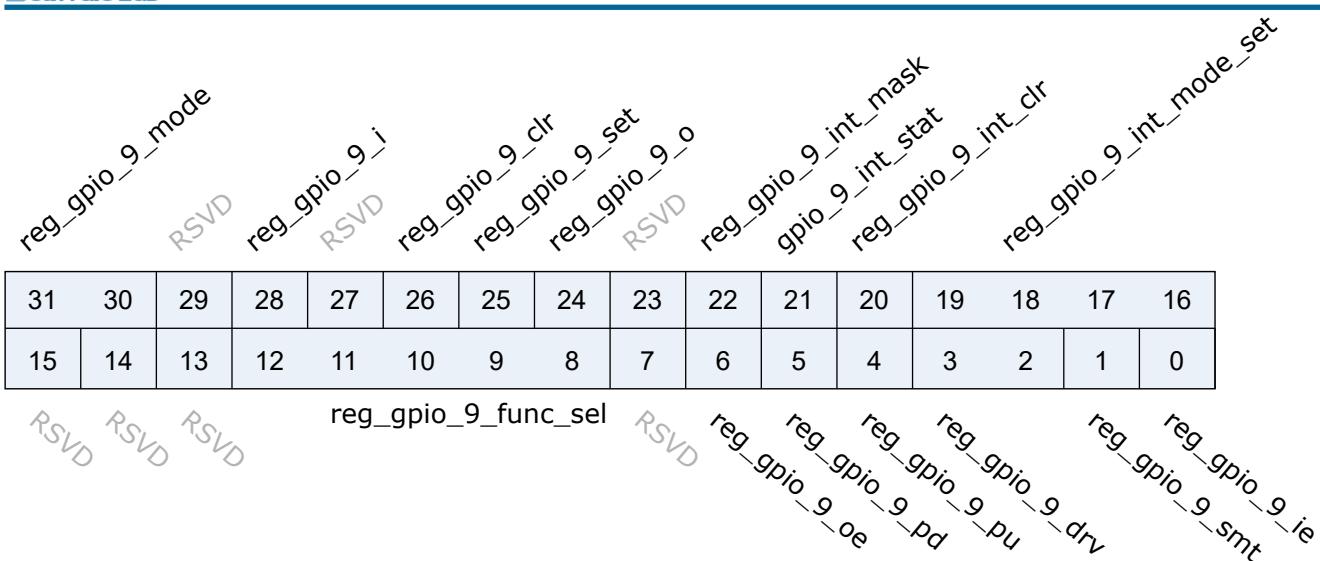


位	名称	权限	复位值	描述
31:30	reg_gpio_8_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_8_i	r	0	
27	RSVD			
26	reg_gpio_8_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_8_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_8_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			

位	名称	权限	复位值	描述
22	reg_gpio_8_int_mask	r/w	1	mask interrupt (1)
21	gpio_8_int_stat	r	0	interrupt status
20	reg_gpio_8_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_8_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_8_func_sel	r/w	5'hB	GPIO Function Select (Default : SW-GPIO)
7	RSVD			
6	reg_gpio_8_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
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	0	GPIO Input Enable

#### 4.8.10 gpio\_cfg9

地址: 0x200008e8

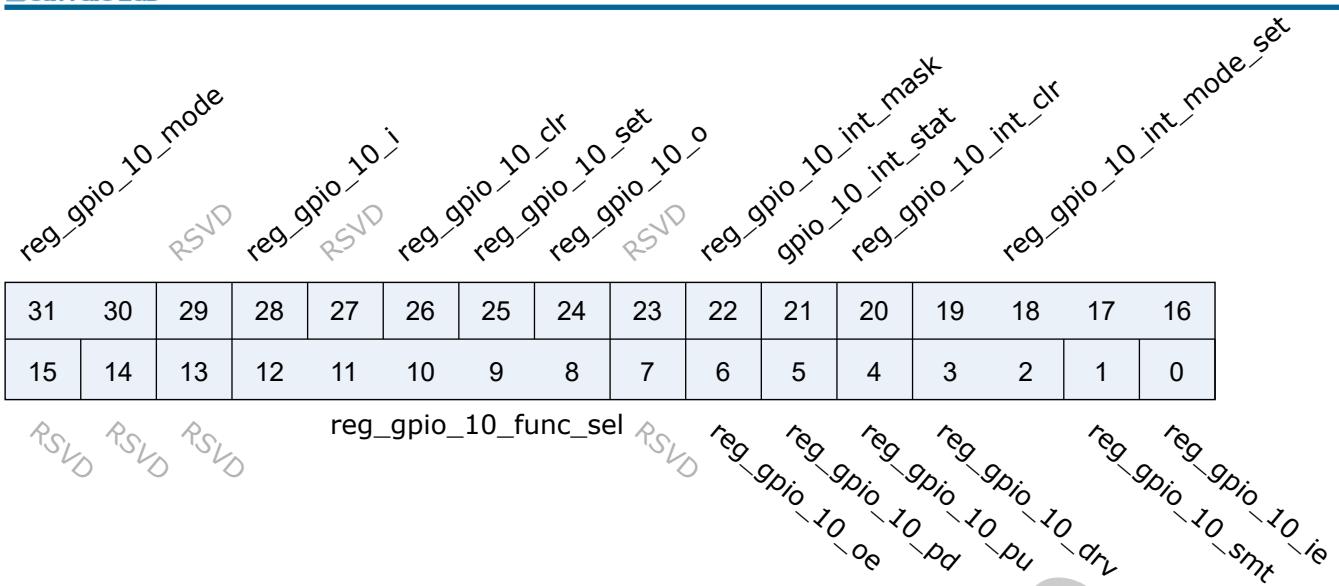


位	名称	权限	复位值	描述
31:30	reg_gpio_9_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_9_i	r	0	
27	RSVD			
26	reg_gpio_9_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_9_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_9_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_9_int_mask	r/w	1	mask interrupt (1)
21	gpio_9_int_stat	r	0	interrupt status

位	名称	权限	复位值	描述
20	reg_gpio_9_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_9_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_9_func_sel	r/w	5'hB	GPIO Function Select (Default : SW-GPIO)
7	RSVD			
6	reg_gpio_9_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
5	reg_gpio_9_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_9_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_9_drv	r/w	0	GPIO Driving Control
1	reg_gpio_9_smt	r/w	1	GPIO SMT Control
0	reg_gpio_9_ie	r/w	0	GPIO Input Enable

#### 4.8.11 gpio\_cfg10

地址: 0x200008ec

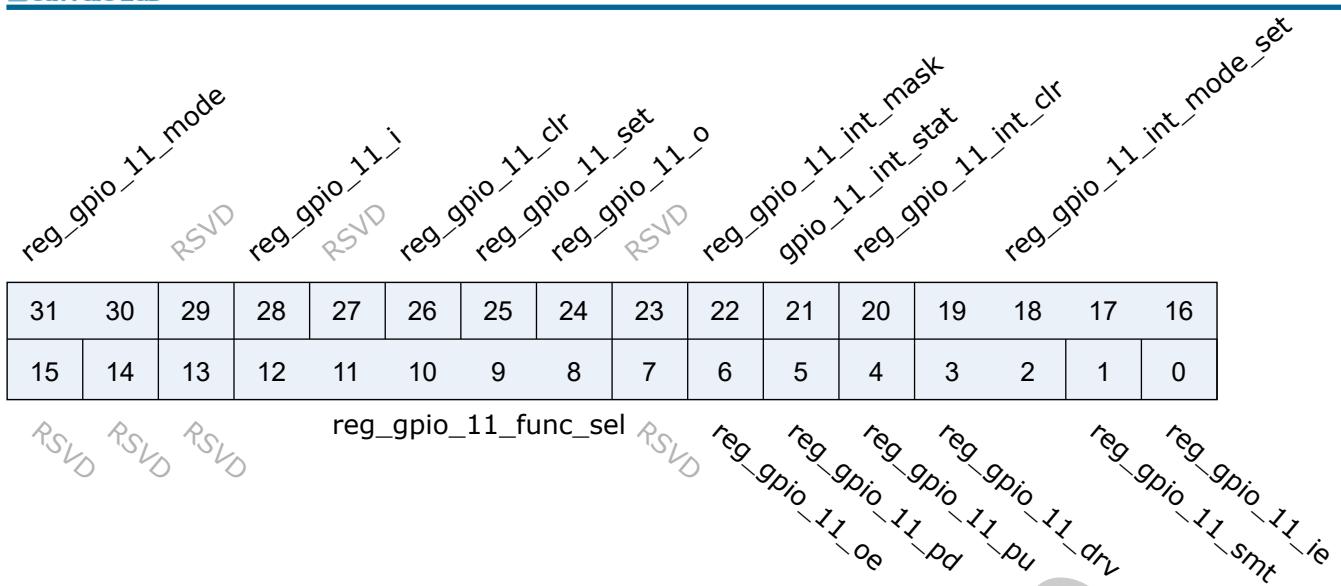


位	名称	权限	复位值	描述
31:30	reg_gpio_10_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_10_i	r	0	
27	RSVD			
26	reg_gpio_10_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_10_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_10_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_10_int_mask	r/w	1	mask interrupt (1)
21	gpio_10_int_stat	r	0	interrupt status

位	名称	权限	复位值	描述
20	reg_gpio_10_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_10_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_10_func_sel	r/w	5'hF	GPIO Function Select (Default : CCI)
7	RSVD			
6	reg_gpio_10_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
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
0	reg_gpio_10_ie	r/w	1	GPIO Input Enable

#### 4.8.12 gpio\_cfg11

地址: 0x200008f0

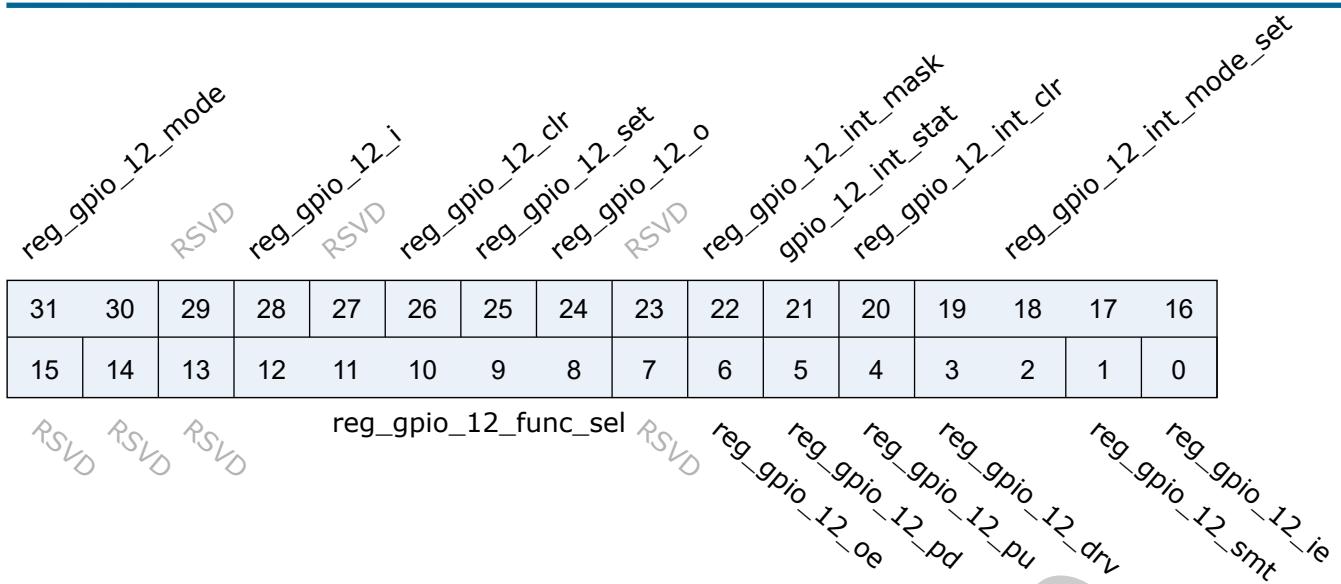


位	名称	权限	复位值	描述
31:30	reg_gpio_11_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_11_i	r	0	
27	RSVD			
26	reg_gpio_11_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_11_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_11_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_11_int_mask	r/w	1	mask interrupt (1)
21	gpio_11_int_stat	r	0	interrupt status

位	名称	权限	复位值	描述
20	reg_gpio_11_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_11_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_11_func_sel	r/w	5'hF	GPIO Function Select (Default : CCI)
7	RSVD			
6	reg_gpio_11_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
5	reg_gpio_11_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_11_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_11_drv	r/w	0	GPIO Driving Control
1	reg_gpio_11_smt	r/w	1	GPIO SMT Control
0	reg_gpio_11_ie	r/w	1	GPIO Input Enable

#### 4.8.13 gpio\_cfg12

地址: 0x200008f4

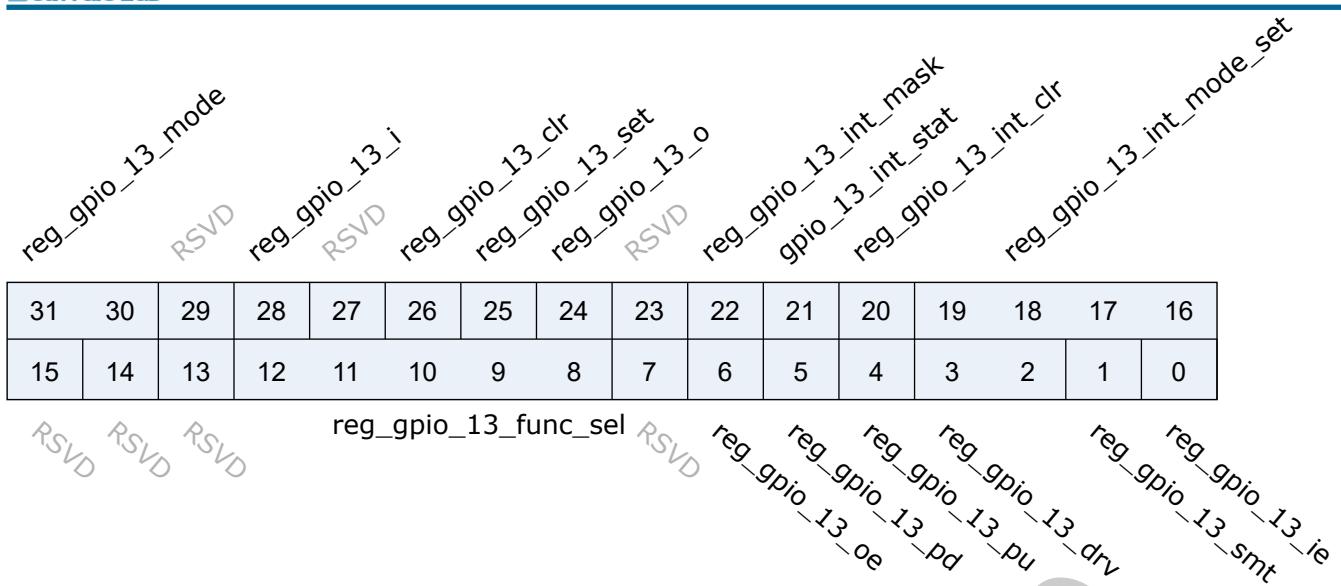


位	名称	权限	复位值	描述
31:30	reg_gpio_12_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_12_i	r	0	
27	RSVD			
26	reg_gpio_12_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_12_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_12_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_12_int_mask	r/w	1	mask interrupt (1)
21	gpio_12_int_stat	r	0	interrupt status

位	名称	权限	复位值	描述
20	reg_gpio_12_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_12_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_12_func_sel	r/w	5'hF	GPIO Function Select (Default : CCI)
7	RSVD			
6	reg_gpio_12_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
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
0	reg_gpio_12_ie	r/w	1	GPIO Input Enable

#### 4.8.14 gpio\_cfg13

地址: 0x200008f8

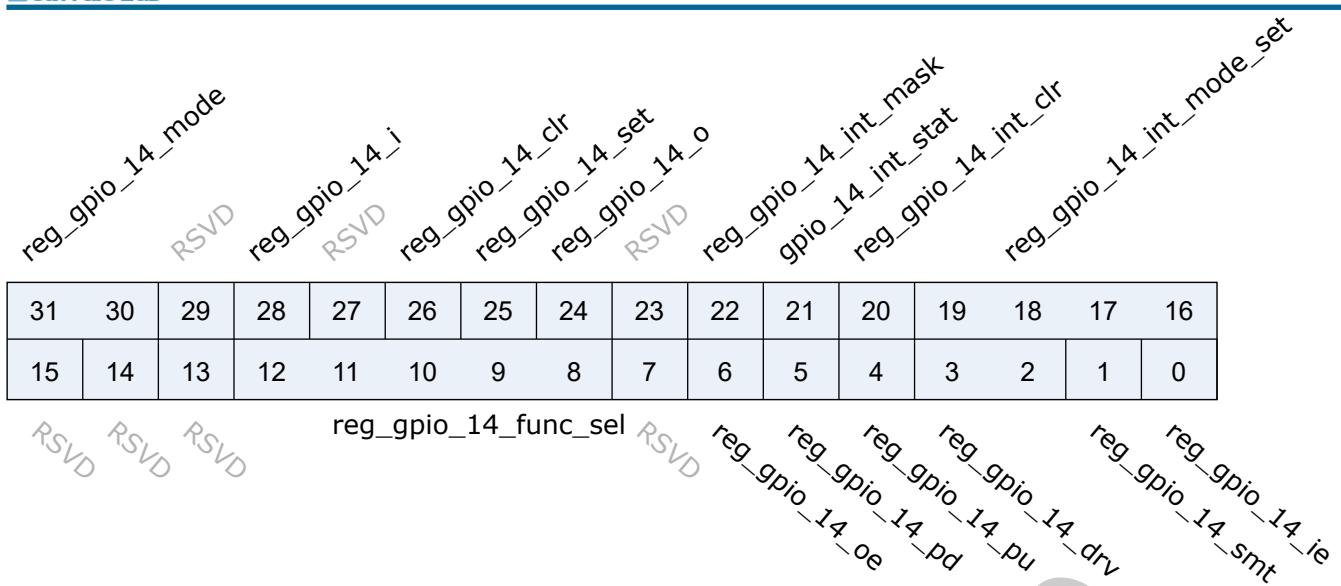


位	名称	权限	复位值	描述
31:30	reg_gpio_13_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_13_i	r	0	
27	RSVD			
26	reg_gpio_13_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_13_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_13_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_13_int_mask	r/w	1	mask interrupt (1)
21	gpio_13_int_stat	r	0	interrupt status

位	名称	权限	复位值	描述
20	reg_gpio_13_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_13_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_13_func_sel	r/w	5'hB	GPIO Function Select (Default : SW-GPIO)
7	RSVD			
6	reg_gpio_13_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
5	reg_gpio_13_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_13_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_13_drv	r/w	0	GPIO Driving Control
1	reg_gpio_13_smt	r/w	1	GPIO SMT Control
0	reg_gpio_13_ie	r/w	0	GPIO Input Enable

#### 4.8.15 gpio\_cfg14

地址: 0x200008fc

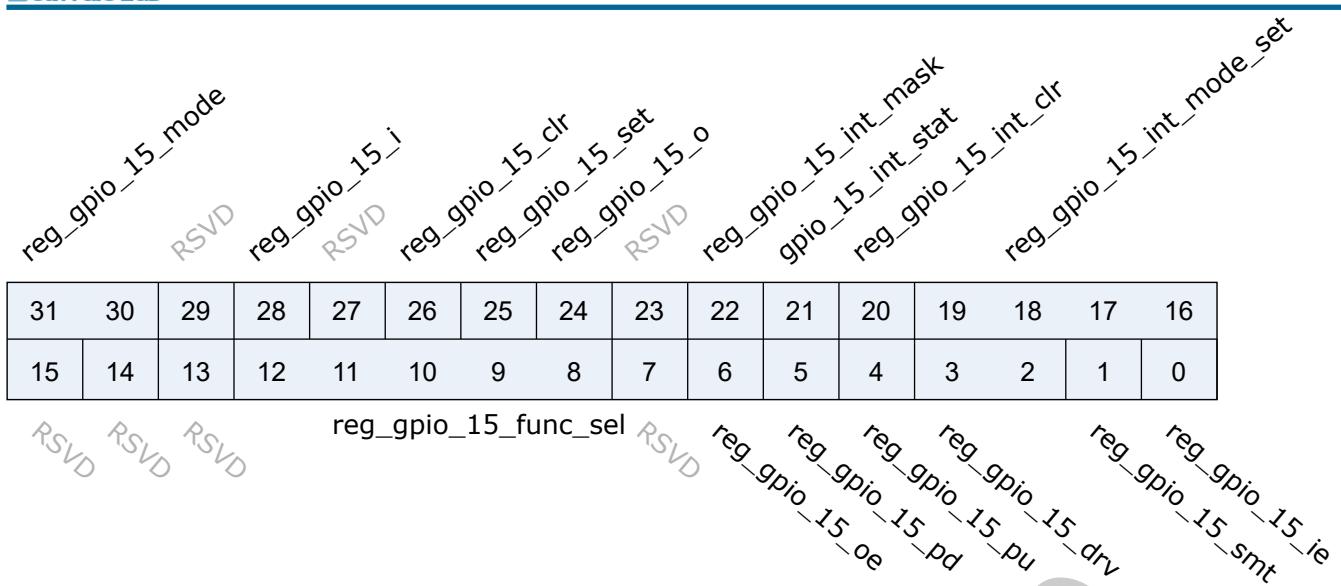


位	名称	权限	复位值	描述
31:30	reg_gpio_14_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_14_i	r	0	
27	RSVD			
26	reg_gpio_14_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_14_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_14_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_14_int_mask	r/w	1	mask interrupt (1)
21	gpio_14_int_stat	r	0	interrupt status

位	名称	权限	复位值	描述
20	reg_gpio_14_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_14_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_14_func_sel	r/w	5'hB	GPIO Function Select (Default : SW-GPIO)
7	RSVD			
6	reg_gpio_14_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
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	0	GPIO Input Enable

#### 4.8.16 gpio\_cfg15

地址: 0x20000900

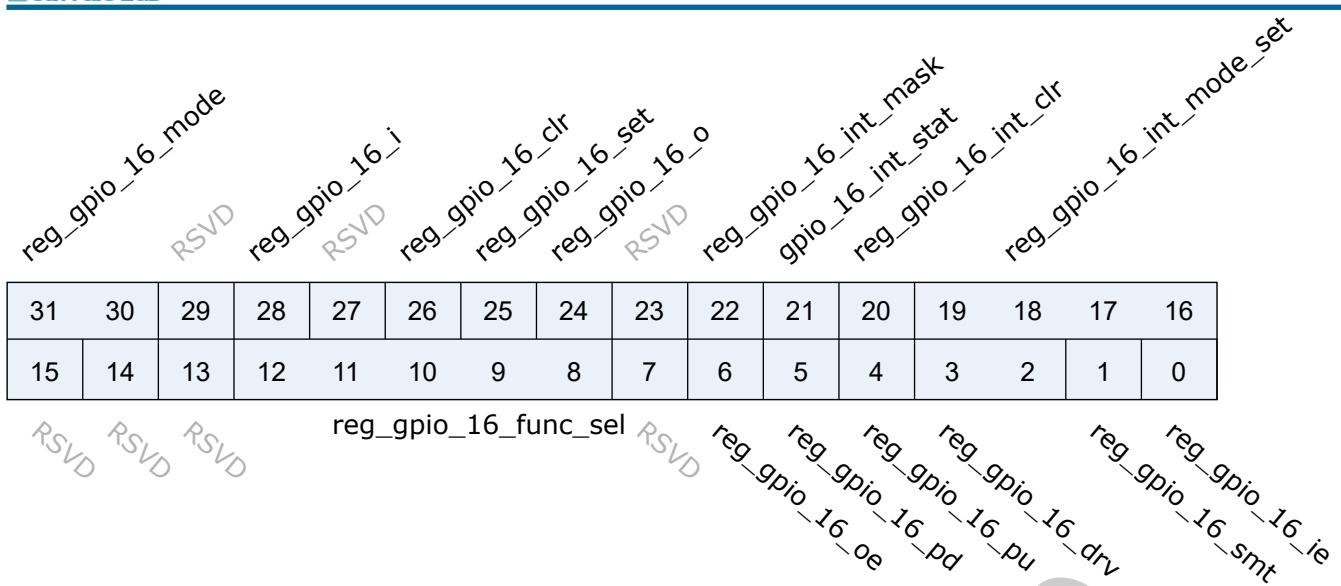


位	名称	权限	复位值	描述
31:30	reg_gpio_15_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_15_i	r	0	
27	RSVD			
26	reg_gpio_15_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_15_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_15_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_15_int_mask	r/w	1	mask interrupt (1)
21	gpio_15_int_stat	r	0	interrupt status

位	名称	权限	复位值	描述
20	reg_gpio_15_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_15_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_15_func_sel	r/w	5'hB	GPIO Function Select (Default : SW-GPIO)
7	RSVD			
6	reg_gpio_15_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
5	reg_gpio_15_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_15_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_15_drv	r/w	0	GPIO Driving Control
1	reg_gpio_15_smt	r/w	1	GPIO SMT Control
0	reg_gpio_15_ie	r/w	0	GPIO Input Enable

#### 4.8.17 gpio\_cfg16

地址: 0x20000904

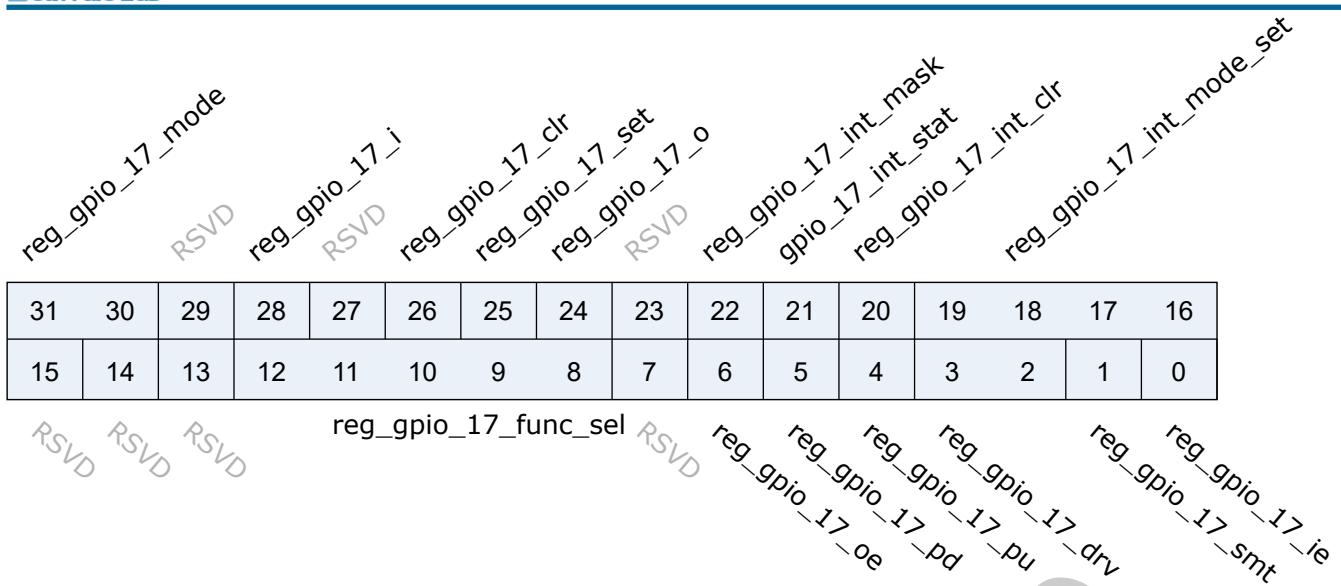


位	名称	权限	复位值	描述
31:30	reg_gpio_16_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_16_i	r	0	
27	RSVD			
26	reg_gpio_16_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_16_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_16_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_16_int_mask	r/w	1	mask interrupt (1)
21	gpio_16_int_stat	r	0	interrupt status

位	名称	权限	复位值	描述
20	reg_gpio_16_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_16_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_16_func_sel	r/w	5'hB	GPIO Function Select (Default : SW-GPIO)
7	RSVD			
6	reg_gpio_16_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
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	0	GPIO Input Enable

#### 4.8.18 gpio\_cfg17

地址: 0x20000908

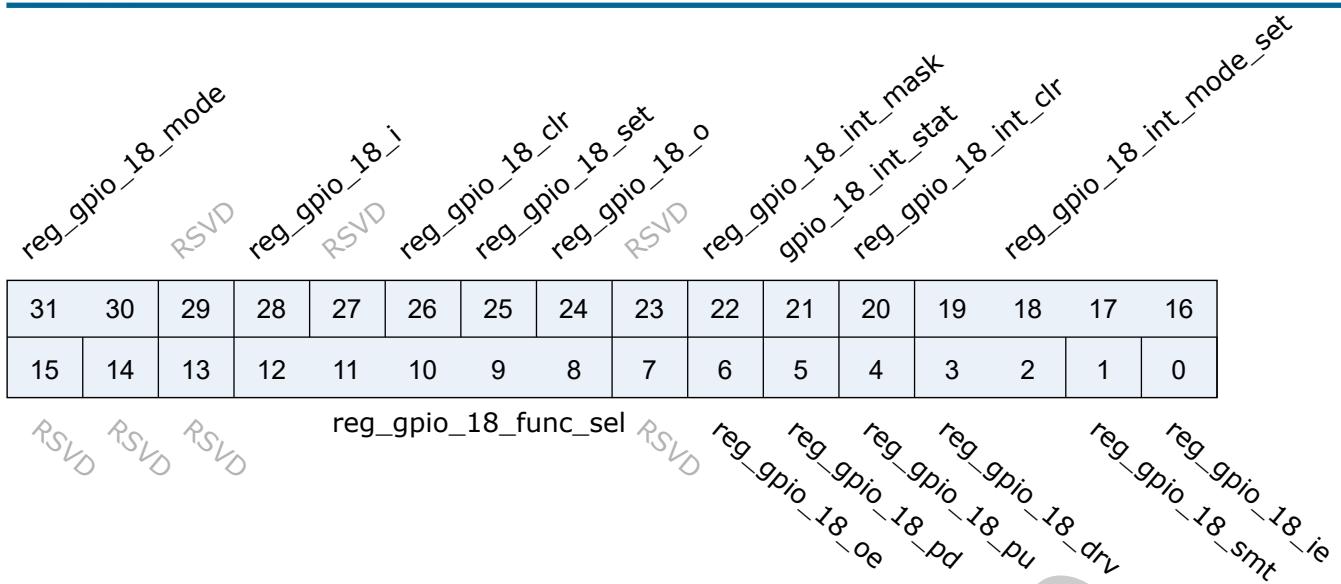


位	名称	权限	复位值	描述
31:30	reg_gpio_17_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_17_i	r	0	
27	RSVD			
26	reg_gpio_17_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_17_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_17_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_17_int_mask	r/w	1	mask interrupt (1)
21	gpio_17_int_stat	r	0	interrupt status

位	名称	权限	复位值	描述
20	reg_gpio_17_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_17_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_17_func_sel	r/w	5'hB	GPIO Function Select (Default : SW-GPIO)
7	RSVD			
6	reg_gpio_17_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
5	reg_gpio_17_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_17_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_17_drv	r/w	0	GPIO Driving Control
1	reg_gpio_17_smt	r/w	1	GPIO SMT Control
0	reg_gpio_17_ie	r/w	0	GPIO Input Enable

#### 4.8.19 gpio\_cfg18

地址: 0x2000090c

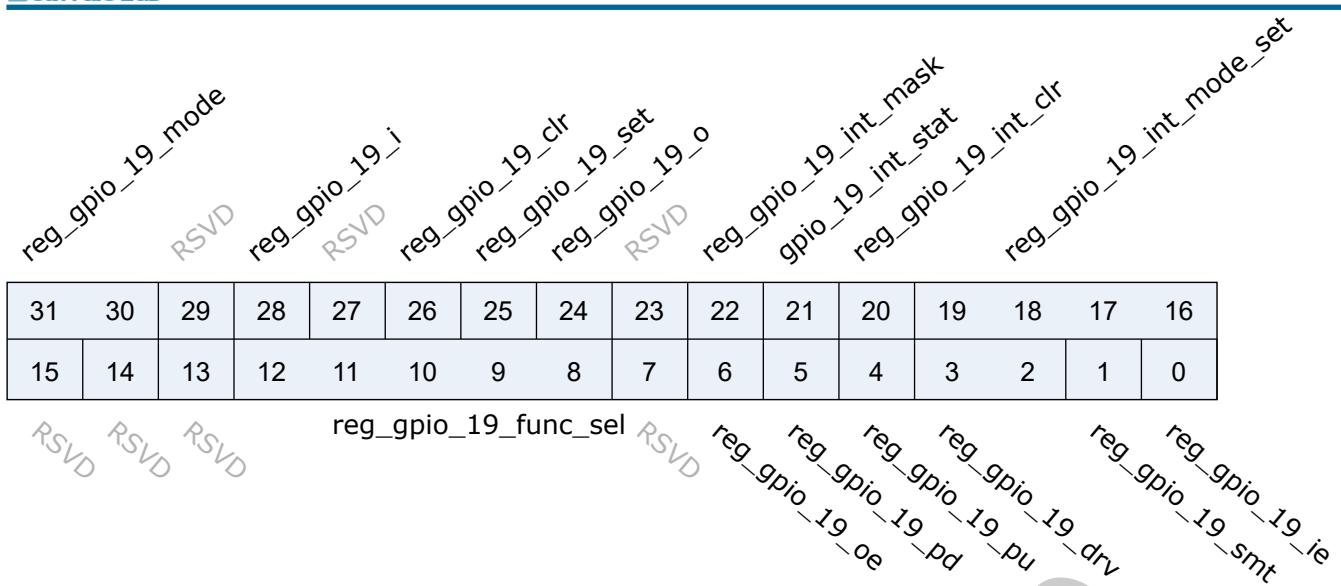


位	名称	权限	复位值	描述
31:30	reg_gpio_18_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_18_i	r	0	
27	RSVD			
26	reg_gpio_18_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_18_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_18_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_18_int_mask	r/w	1	mask interrupt (1)
21	gpio_18_int_stat	r	0	interrupt status

位	名称	权限	复位值	描述
20	reg_gpio_18_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_18_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_18_func_sel	r/w	5'hB	GPIO Function Select (Default : SW-GPIO)
7	RSVD			
6	reg_gpio_18_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
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	0	GPIO Input Enable

#### 4.8.20 gpio\_cfg19

地址: 0x20000910

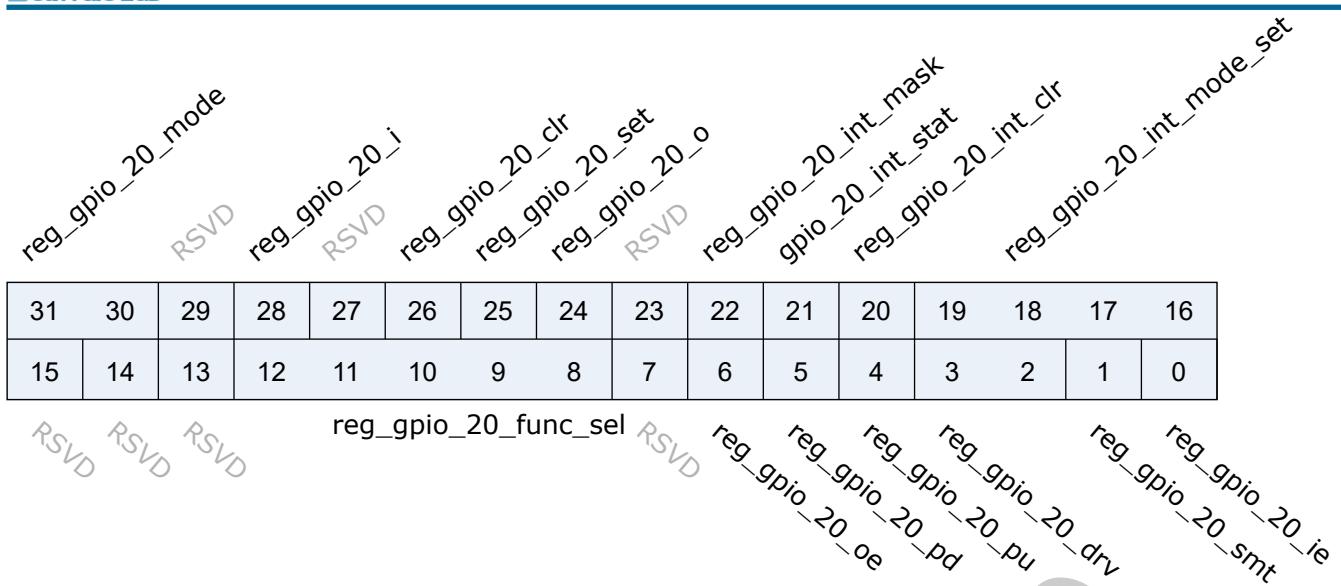


位	名称	权限	复位值	描述
31:30	reg_gpio_19_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_19_i	r	0	
27	RSVD			
26	reg_gpio_19_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_19_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_19_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_19_int_mask	r/w	1	mask interrupt (1)
21	gpio_19_int_stat	r	0	interrupt status

位	名称	权限	复位值	描述
20	reg_gpio_19_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_19_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_19_func_sel	r/w	5'hB	GPIO Function Select (Default : SW-GPIO)
7	RSVD			
6	reg_gpio_19_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
5	reg_gpio_19_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_19_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_19_drv	r/w	0	GPIO Driving Control
1	reg_gpio_19_smt	r/w	1	GPIO SMT Control
0	reg_gpio_19_ie	r/w	0	GPIO Input Enable

#### 4.8.21 gpio\_cfg20

地址: 0x20000914

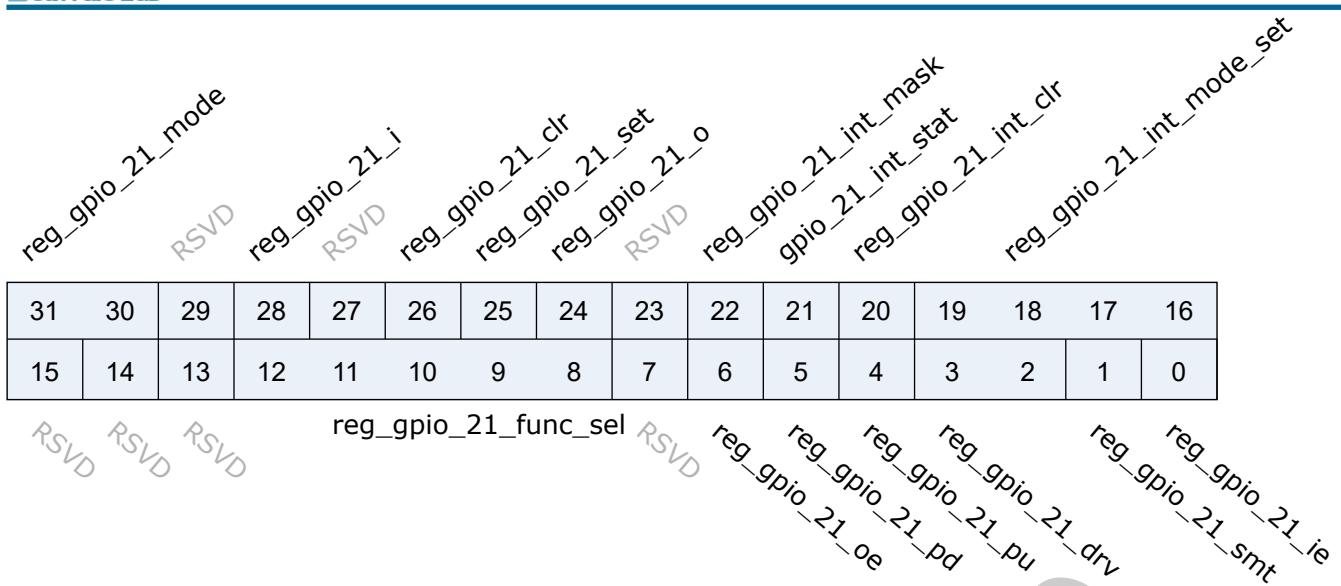


位	名称	权限	复位值	描述
31:30	reg_gpio_20_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_20_i	r	0	
27	RSVD			
26	reg_gpio_20_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_20_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_20_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_20_int_mask	r/w	1	mask interrupt (1)
21	gpio_20_int_stat	r	0	interrupt status

位	名称	权限	复位值	描述
20	reg_gpio_20_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_20_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_20_func_sel	r/w	5'hB	GPIO Function Select (Default : SW-GPIO)
7	RSVD			
6	reg_gpio_20_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
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	0	GPIO Input Enable

#### 4.8.22 gpio\_cfg21

地址: 0x20000918

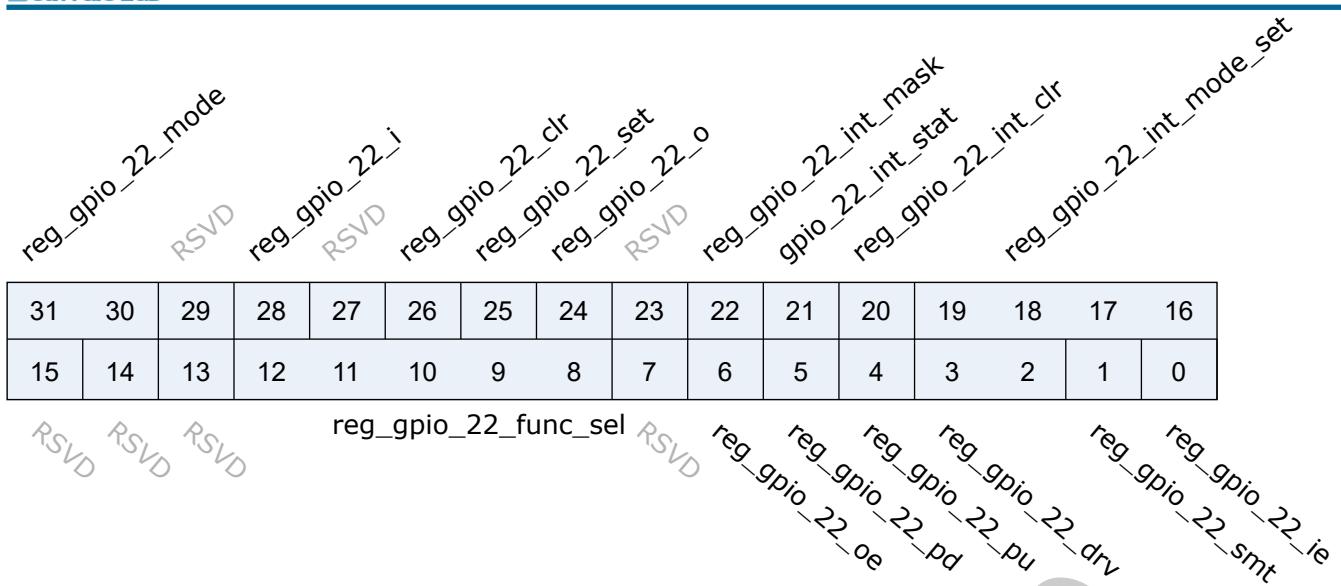


位	名称	权限	复位值	描述
31:30	reg_gpio_21_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_21_i	r	0	
27	RSVD			
26	reg_gpio_21_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_21_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_21_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_21_int_mask	r/w	1	mask interrupt (1)
21	gpio_21_int_stat	r	0	interrupt status

位	名称	权限	复位值	描述
20	reg_gpio_21_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_21_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_21_func_sel	r/w	5'hB	GPIO Function Select (Default : SW-GPIO)
7	RSVD			
6	reg_gpio_21_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
5	reg_gpio_21_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_21_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_21_drv	r/w	0	GPIO Driving Control
1	reg_gpio_21_smt	r/w	1	GPIO SMT Control
0	reg_gpio_21_ie	r/w	0	GPIO Input Enable

#### 4.8.23 gpio\_cfg22

地址: 0x2000091c

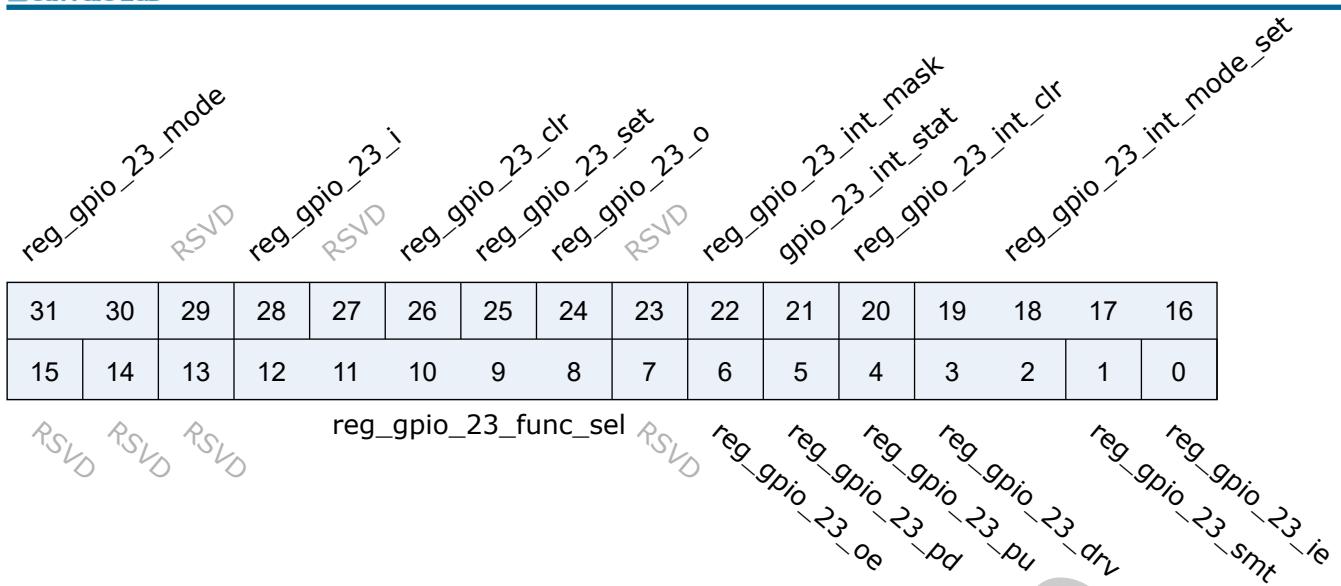


位	名称	权限	复位值	描述
31:30	reg_gpio_22_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_22_i	r	0	
27	RSVD			
26	reg_gpio_22_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_22_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_22_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_22_int_mask	r/w	1	mask interrupt (1)
21	gpio_22_int_stat	r	0	interrupt status

位	名称	权限	复位值	描述
20	reg_gpio_22_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_22_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_22_func_sel	r/w	5'hB	GPIO Function Select (Default : SW-GPIO)
7	RSVD			
6	reg_gpio_22_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
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	0	GPIO Input Enable

#### 4.8.24 gpio\_cfg23

地址: 0x20000920

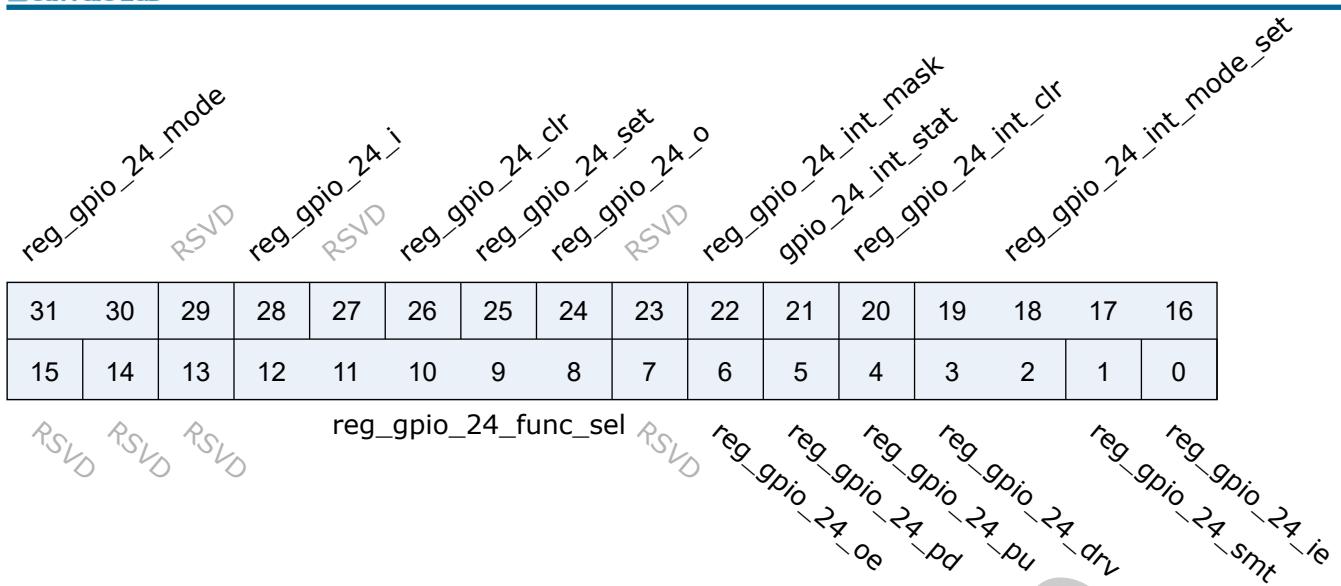


位	名称	权限	复位值	描述
31:30	reg_gpio_23_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_23_i	r	0	
27	RSVD			
26	reg_gpio_23_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_23_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_23_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_23_int_mask	r/w	1	mask interrupt (1)
21	gpio_23_int_stat	r	0	interrupt status

位	名称	权限	复位值	描述
20	reg_gpio_23_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_23_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_23_func_sel	r/w	5'hB	GPIO Function Select (Default : SW-GPIO)
7	RSVD			
6	reg_gpio_23_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
5	reg_gpio_23_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_23_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_23_drv	r/w	0	GPIO Driving Control
1	reg_gpio_23_smt	r/w	1	GPIO SMT Control
0	reg_gpio_23_ie	r/w	0	GPIO Input Enable

#### 4.8.25 gpio\_cfg24

地址: 0x20000924

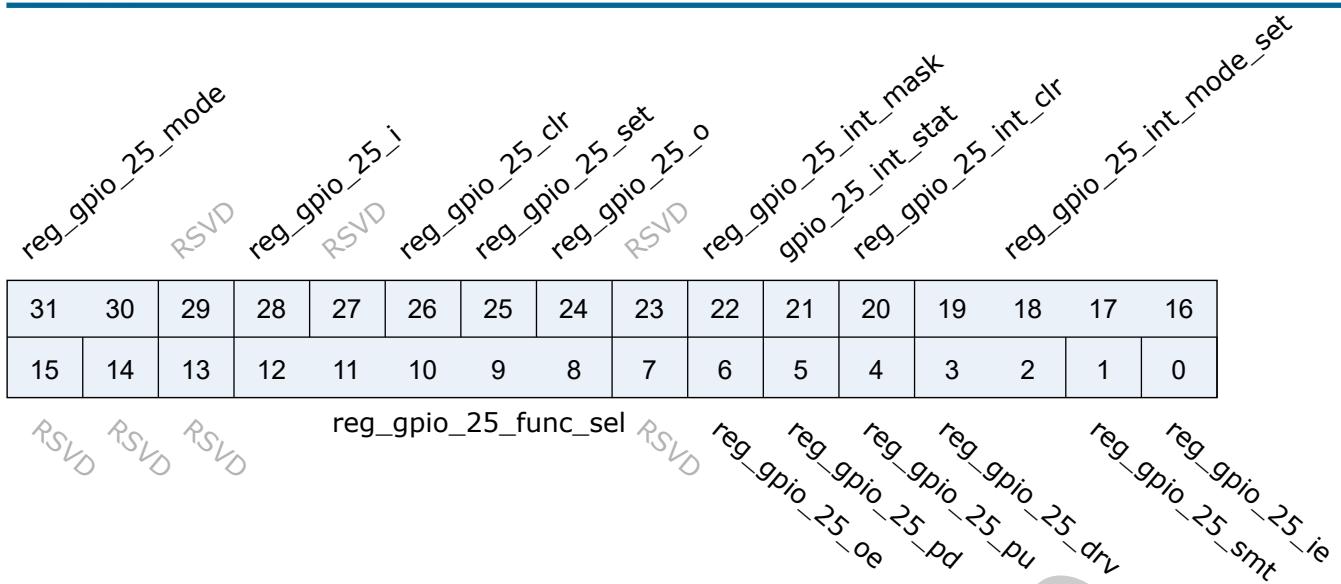


位	名称	权限	复位值	描述
31:30	reg_gpio_24_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_24_i	r	0	
27	RSVD			
26	reg_gpio_24_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_24_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_24_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_24_int_mask	r/w	1	mask interrupt (1)
21	gpio_24_int_stat	r	0	interrupt status

位	名称	权限	复位值	描述
20	reg_gpio_24_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_24_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_24_func_sel	r/w	5'hB	GPIO Function Select (Default : SW-GPIO)
7	RSVD			
6	reg_gpio_24_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
5	reg_gpio_24_pd	r/w	0	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	0	GPIO Input Enable

#### 4.8.26 gpio\_cfg25

地址: 0x20000928

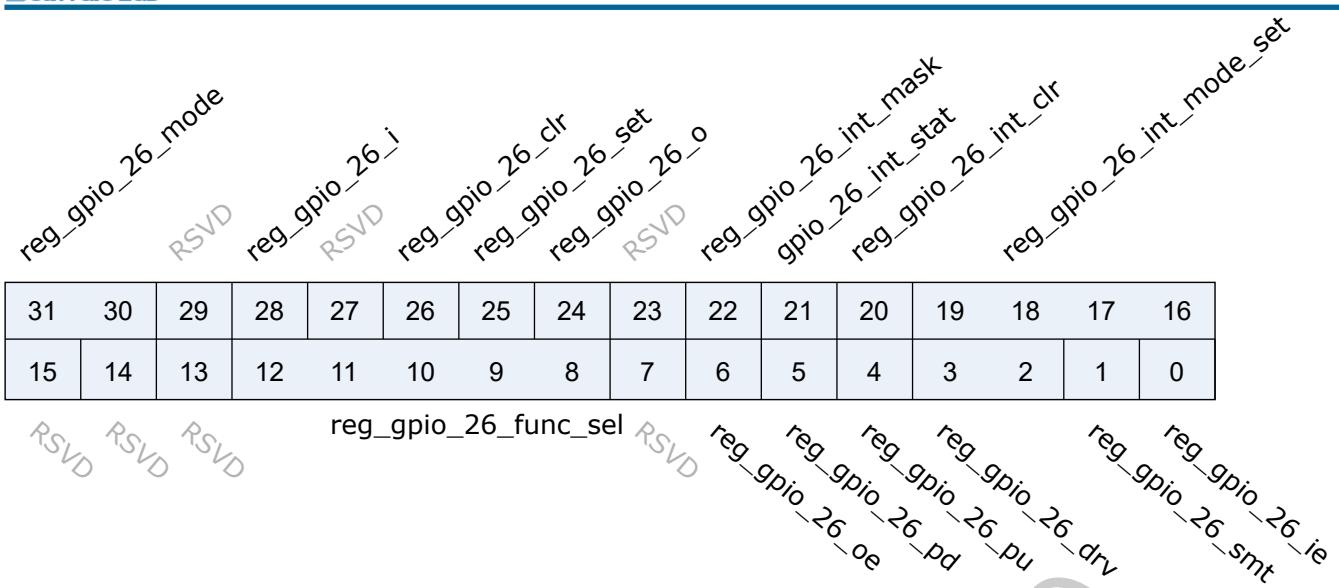


位	名称	权限	复位值	描述
31:30	reg_gpio_25_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_25_i	r	0	
27	RSVD			
26	reg_gpio_25_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_25_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_25_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_25_int_mask	r/w	1	mask interrupt (1)
21	gpio_25_int_stat	r	0	interrupt status

位	名称	权限	复位值	描述
20	reg_gpio_25_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_25_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_25_func_sel	r/w	5'hB	GPIO Function Select (Default : SW-GPIO)
7	RSVD			
6	reg_gpio_25_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
5	reg_gpio_25_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_25_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_25_drv	r/w	0	GPIO Driving Control
1	reg_gpio_25_smt	r/w	1	GPIO SMT Control
0	reg_gpio_25_ie	r/w	0	GPIO Input Enable

#### 4.8.27 gpio\_cfg26

地址: 0x2000092c

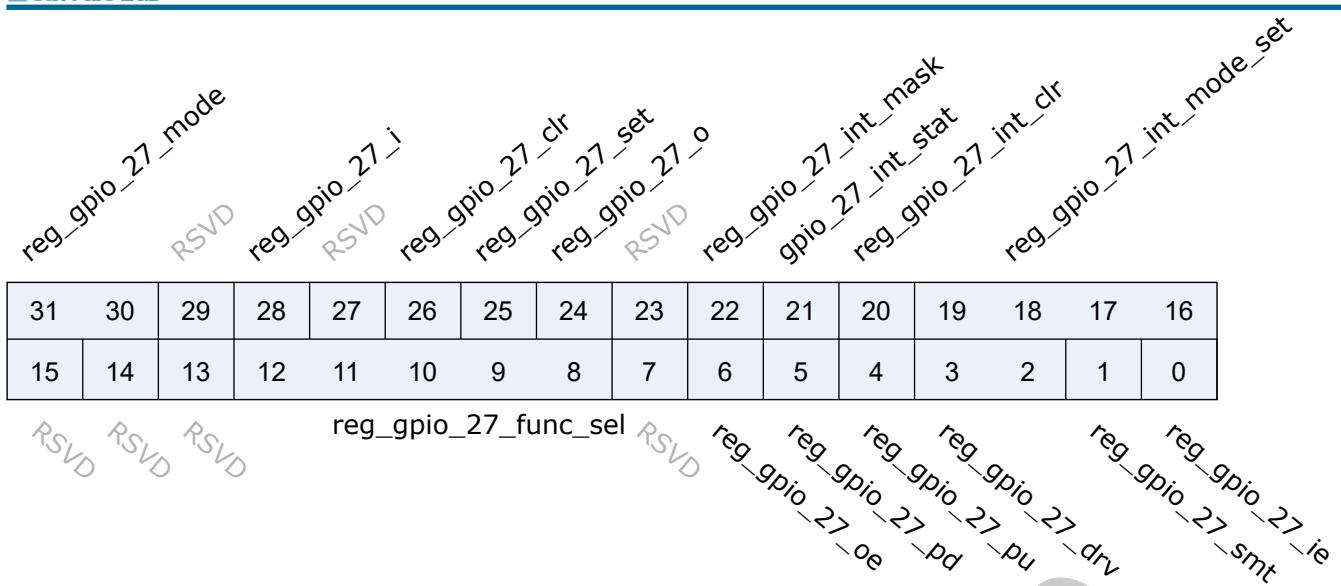


位	名称	权限	复位值	描述
31:30	reg_gpio_26_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_26_i	r	0	
27	RSVD			
26	reg_gpio_26_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_26_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_26_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_26_int_mask	r/w	1	mask interrupt (1)
21	gpio_26_int_stat	r	0	interrupt status

位	名称	权限	复位值	描述
20	reg_gpio_26_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_26_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_26_func_sel	r/w	5'hB	GPIO Function Select (Default : SW-GPIO)
7	RSVD			
6	reg_gpio_26_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
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	0	GPIO Input Enable

#### 4.8.28 gpio\_cfg27

地址: 0x20000930

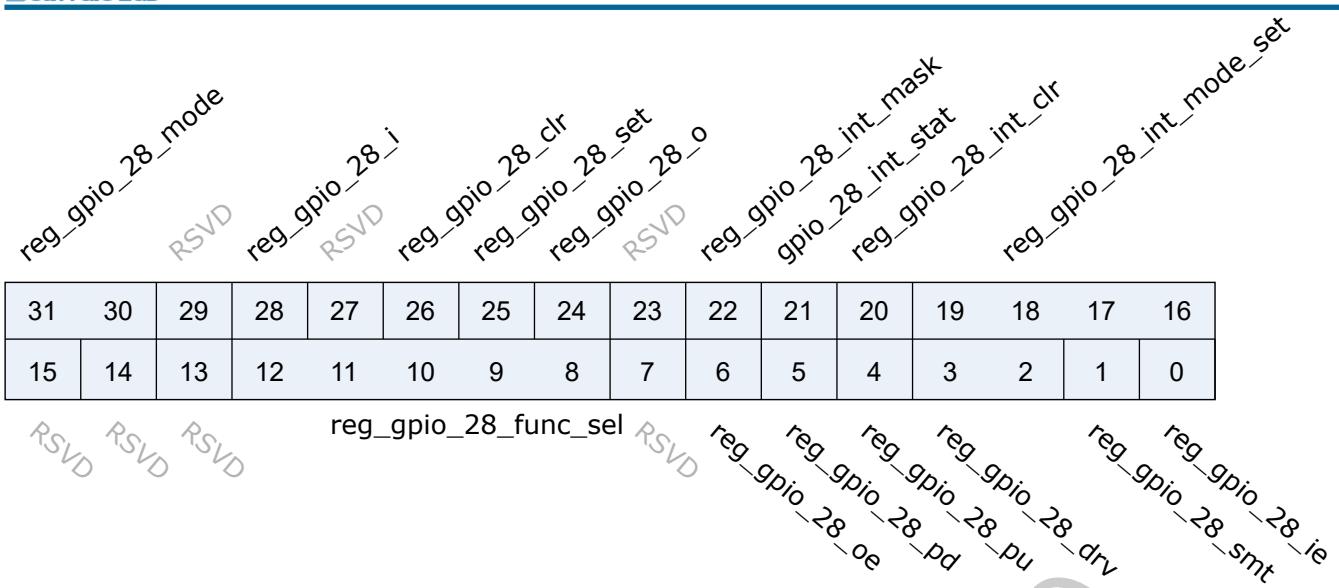


位	名称	权限	复位值	描述
31:30	reg_gpio_27_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_27_i	r	0	
27	RSVD			
26	reg_gpio_27_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_27_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_27_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_27_int_mask	r/w	1	mask interrupt (1)
21	gpio_27_int_stat	r	0	interrupt status

位	名称	权限	复位值	描述
20	reg_gpio_27_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_27_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_27_func_sel	r/w	5'hB	GPIO Function Select (Default : SW-GPIO)
7	RSVD			
6	reg_gpio_27_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
5	reg_gpio_27_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_27_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_27_drv	r/w	0	GPIO Driving Control
1	reg_gpio_27_smt	r/w	1	GPIO SMT Control
0	reg_gpio_27_ie	r/w	0	GPIO Input Enable

#### 4.8.29 gpio\_cfg28

地址: 0x20000934

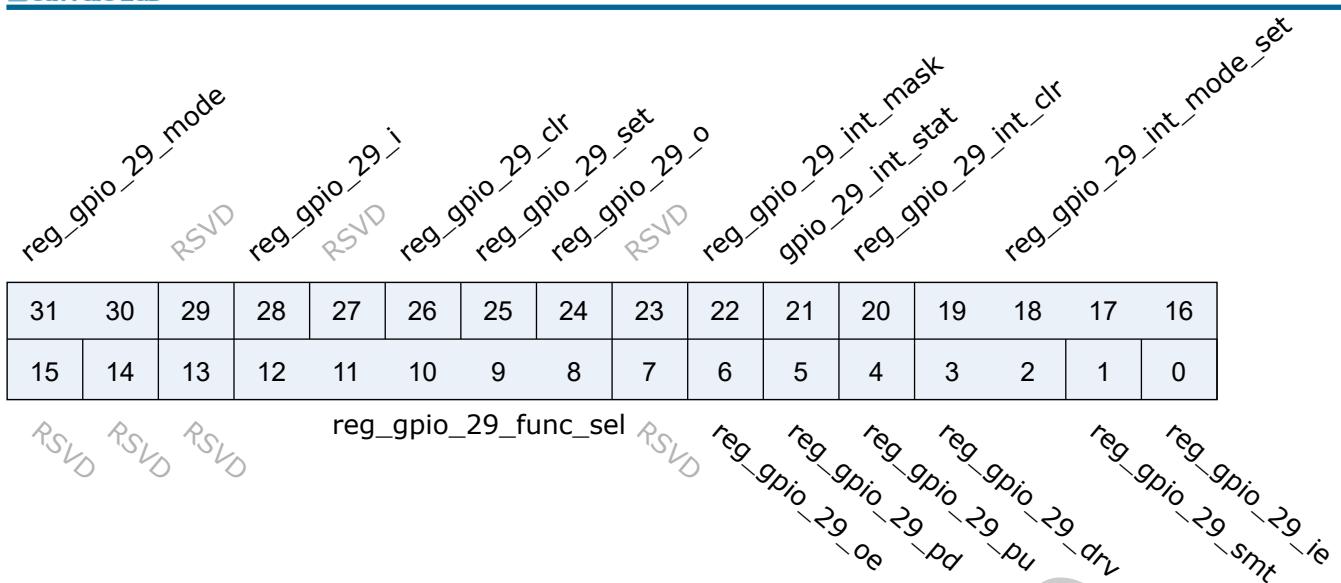


位	名称	权限	复位值	描述
31:30	reg_gpio_28_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_28_i	r	0	
27	RSVD			
26	reg_gpio_28_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_28_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_28_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_28_int_mask	r/w	1	mask interrupt (1)
21	gpio_28_int_stat	r	0	interrupt status

位	名称	权限	复位值	描述
20	reg_gpio_28_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_28_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_28_func_sel	r/w	5'hB	GPIO Function Select (Default : SW-GPIO)
7	RSVD			
6	reg_gpio_28_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
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	0	GPIO Input Enable

#### 4.8.30 gpio\_cfg29

地址: 0x20000938

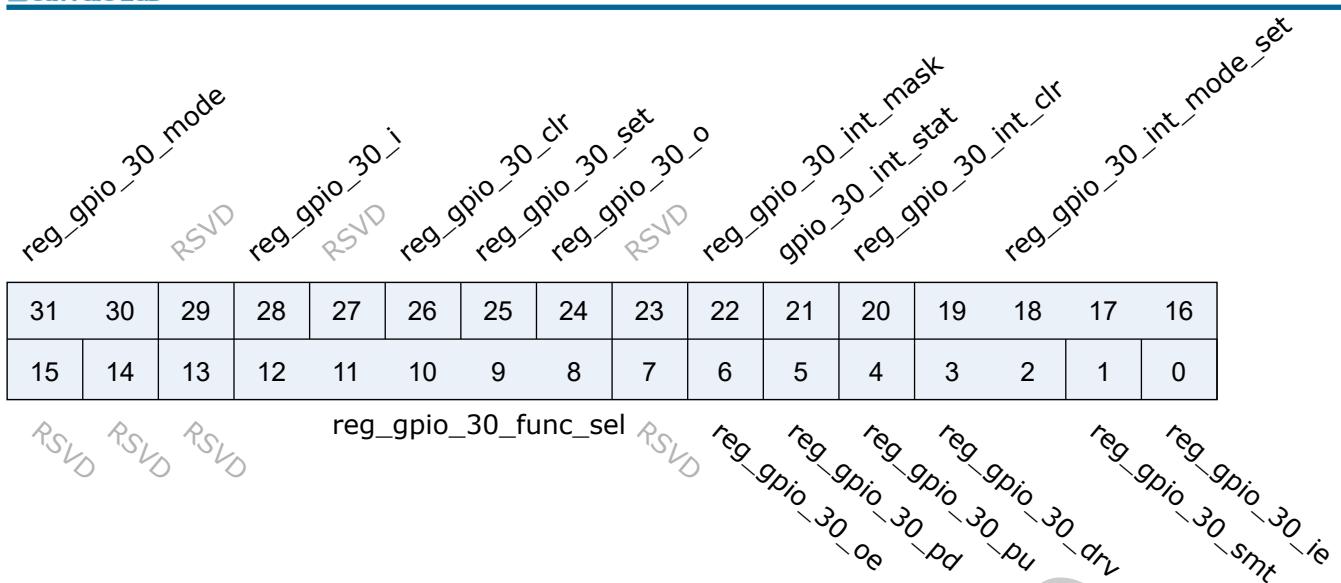


位	名称	权限	复位值	描述
31:30	reg_gpio_29_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_29_i	r	0	
27	RSVD			
26	reg_gpio_29_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_29_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_29_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_29_int_mask	r/w	1	mask interrupt (1)
21	gpio_29_int_stat	r	0	interrupt status

位	名称	权限	复位值	描述
20	reg_gpio_29_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_29_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_29_func_sel	r/w	5'hB	GPIO Function Select (Default : SW-GPIO)
7	RSVD			
6	reg_gpio_29_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
5	reg_gpio_29_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_29_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_29_drv	r/w	0	GPIO Driving Control
1	reg_gpio_29_smt	r/w	1	GPIO SMT Control
0	reg_gpio_29_ie	r/w	0	GPIO Input Enable

#### 4.8.31 gpio\_cfg30

地址: 0x2000093c

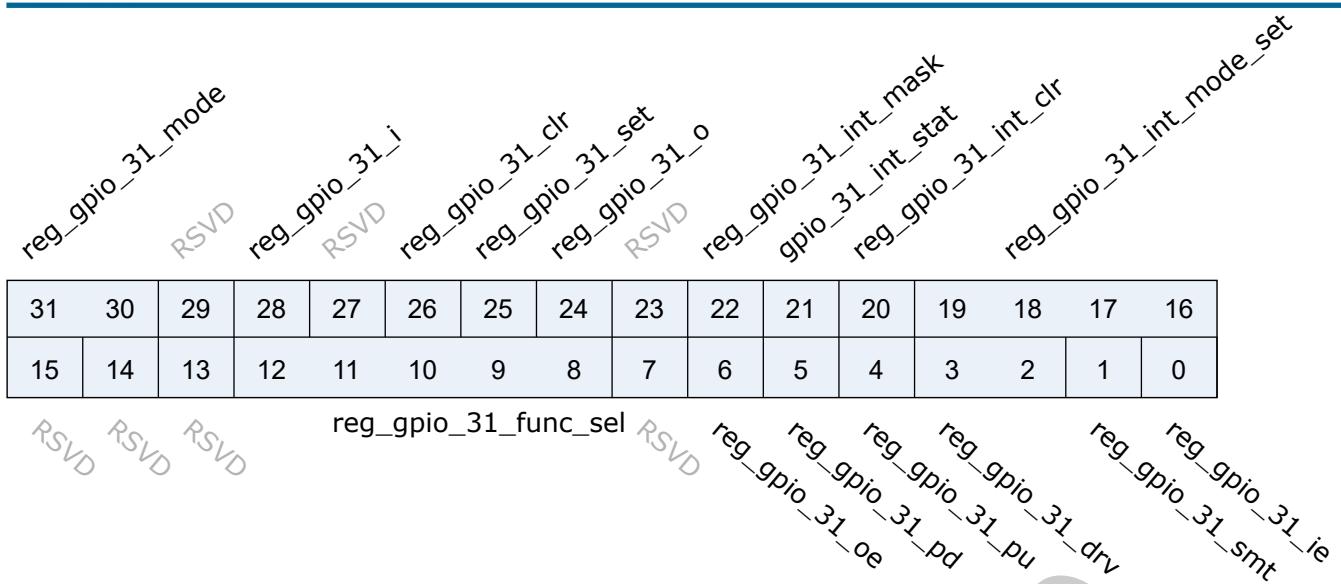


位	名称	权限	复位值	描述
31:30	reg_gpio_30_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_30_i	r	0	
27	RSVD			
26	reg_gpio_30_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_30_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_30_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_30_int_mask	r/w	1	mask interrupt (1)
21	gpio_30_int_stat	r	0	interrupt status

位	名称	权限	复位值	描述
20	reg_gpio_30_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_30_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_30_func_sel	r/w	5'hB	GPIO Function Select (Default : SW-GPIO)
7	RSVD			
6	reg_gpio_30_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
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	0	GPIO Input Enable

#### 4.8.32 gpio\_cfg31

地址: 0x20000940

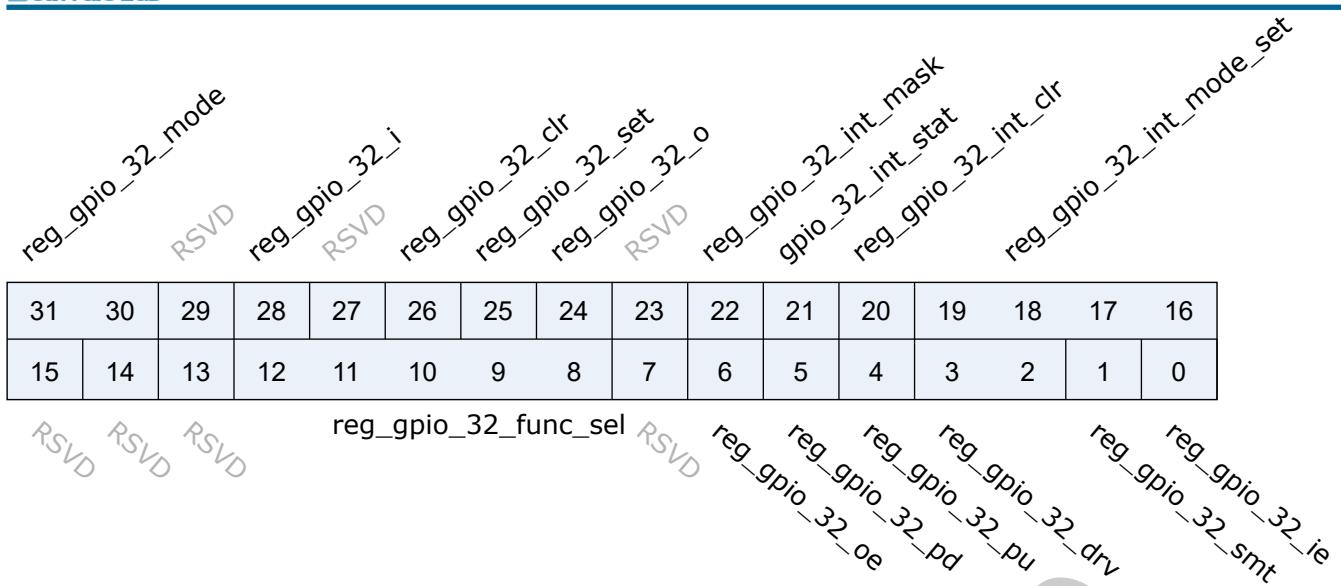


位	名称	权限	复位值	描述
31:30	reg_gpio_31_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_31_i	r	0	
27	RSVD			
26	reg_gpio_31_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_31_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_31_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_31_int_mask	r/w	1	mask interrupt (1)
21	gpio_31_int_stat	r	0	interrupt status

位	名称	权限	复位值	描述
20	reg_gpio_31_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_31_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_31_func_sel	r/w	5'hB	GPIO Function Select (Default : SW-GPIO)
7	RSVD			
6	reg_gpio_31_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
5	reg_gpio_31_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_31_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_31_drv	r/w	0	GPIO Driving Control
1	reg_gpio_31_smt	r/w	1	GPIO SMT Control
0	reg_gpio_31_ie	r/w	0	GPIO Input Enable

#### 4.8.33 gpio\_cfg32

地址: 0x20000944

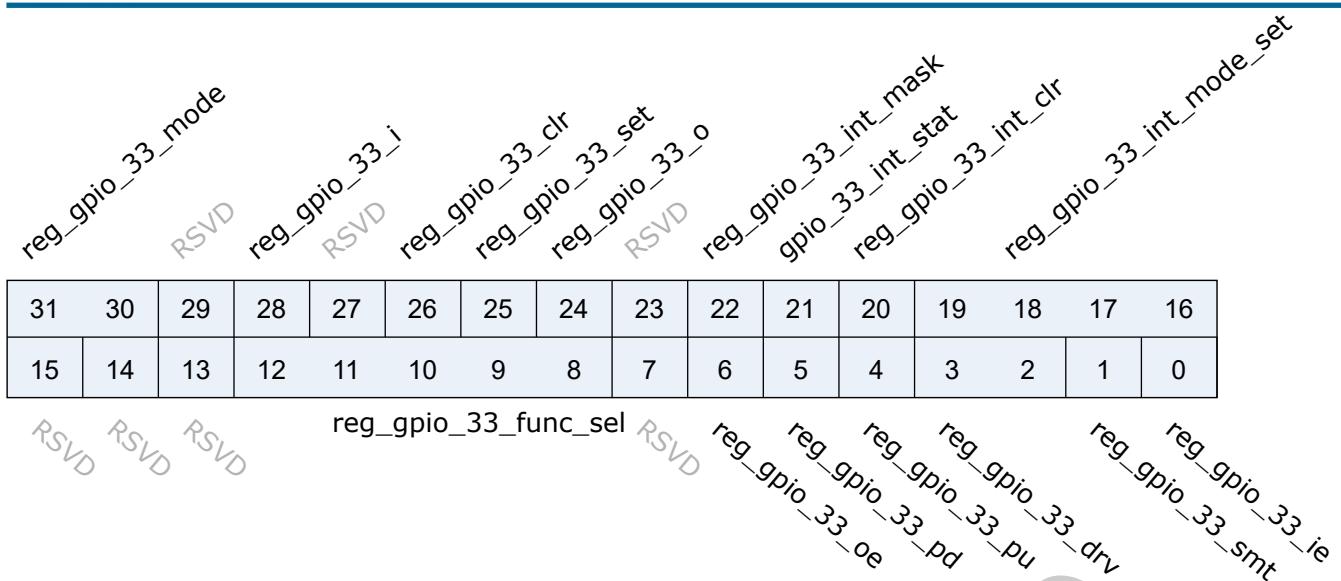


位	名称	权限	复位值	描述
31:30	reg_gpio_32_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_32_i	r	0	
27	RSVD			
26	reg_gpio_32_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_32_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_32_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_32_int_mask	r/w	1	mask interrupt (1)
21	gpio_32_int_stat	r	0	interrupt status

位	名称	权限	复位值	描述
20	reg_gpio_32_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_32_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_32_func_sel	r/w	5'hB	GPIO Function Select (Default : SW-GPIO)
7	RSVD			
6	reg_gpio_32_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
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	0	GPIO Input Enable

#### 4.8.34 gpio\_cfg33

地址: 0x20000948

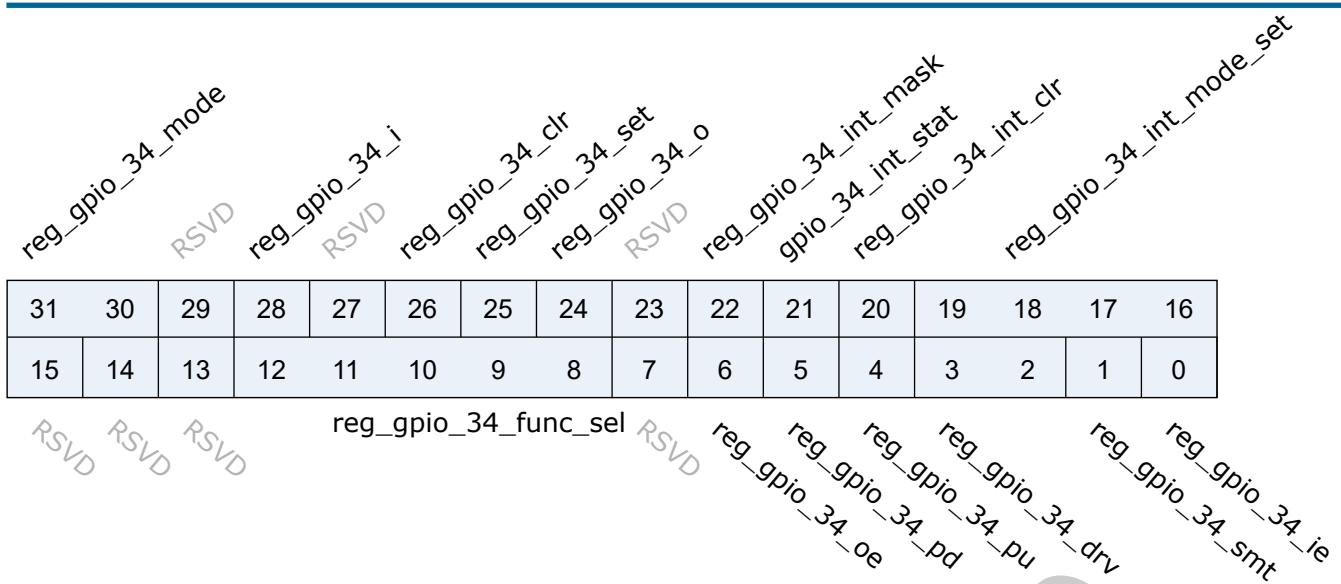


位	名称	权限	复位值	描述
31:30	<i>reg_gpio_33_mode</i>	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by <i>reg_gpio_x_o</i> Value 01 (Set/Clear Mode) :GPIO Output set by <i>reg_gpio_x_set</i> and clear by <i>reg_gpio_x_clr</i> 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by <i>gpio_dma_o</i> 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by <i>gpio_dma_set/gpio_dma_clr</i>
29	RSVD			
28	<i>reg_gpio_33_i</i>	r	0	
27	RSVD			
26	<i>reg_gpio_33_clr</i>	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	<i>reg_gpio_33_set</i>	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	<i>reg_gpio_33_o</i>	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by <i>clr_reg</i>
23	RSVD			
22	<i>reg_gpio_33_int_mask</i>	r/w	1	mask interrupt (1)
21	<i>gpio_33_int_stat</i>	r	0	interrupt status

位	名称	权限	复位值	描述
20	reg_gpio_33_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_33_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_33_func_sel	r/w	5'hB	GPIO Function Select (Default : SW-GPIO)
7	RSVD			
6	reg_gpio_33_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
5	reg_gpio_33_pd	r/w	0	GPIO Pull Down Control
4	reg_gpio_33_pu	r/w	0	GPIO Pull Up Control
3:2	reg_gpio_33_drv	r/w	0	GPIO Driving Control
1	reg_gpio_33_smt	r/w	1	GPIO SMT Control
0	reg_gpio_33_ie	r/w	0	GPIO Input Enable

#### 4.8.35 gpio\_cfg34

地址: 0x2000094c



位	名称	权限	复位值	描述
31:30	reg_gpio_34_mode	r/w	0	When GPIO Function Selected to SWGPIO 00 (Output Value Mode): GPIO Output by reg_gpio_x_o Value 01 (Set/Clear Mode) :GPIO Output set by reg_gpio_x_set and clear by reg_gpio_x_clr 10 : SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Output value by gpio_dma_o 11: SWGPIO Source comes from GPIO DMA (GPIO DMA Mode), GPIO Outout value by gpio_dma_set/gpio_dma_clr
29	RSVD			
28	reg_gpio_34_i	r	0	
27	RSVD			
26	reg_gpio_34_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg_gpio_34_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg_gpio_34_o	r/w	0	When SWGPIO @ Output Value Mode 00 : GPIO Value changes according to this value 01 : GPIO Value Set by this register and clr by clr_reg
23	RSVD			
22	reg_gpio_34_int_mask	r/w	1	mask interrupt (1)
21	gpio_34_int_stat	r	0	interrupt status

位	名称	权限	复位值	描述
20	reg_gpio_34_int_clr	r/w	0	clear interrupt
19:16	reg_gpio_34_int_mode_set	r/w	0	0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
15:13	RSVD			
12:8	reg_gpio_34_func_sel	r/w	5'hB	GPIO Function Select (Default : SW-GPIO)
7	RSVD			
6	reg_gpio_34_oe	r/w	0	Register Controlled GPIO Output Enable (Used when GPIO Function select to Register Control GPIO)
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	0	GPIO Input Enable

#### 4.8.36 gpio\_cfg128

地址: 0x20000ac4

reg2_gpio_31_i	reg2_gpio_30_i	reg2_gpio_29_i	reg2_gpio_28_i	reg2_gpio_27_i	reg2_gpio_26_i	reg2_gpio_25_i	reg2_gpio_24_i	reg2_gpio_23_i	reg2_gpio_22_i	reg2_gpio_21_i	reg2_gpio_20_i	reg2_gpio_19_i	reg2_gpio_18_i	reg2_gpio_17_i	reg2_gpio_16_i
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

reg2\_gpio\_15\_i reg2\_gpio\_14\_i reg2\_gpio\_13\_i reg2\_gpio\_12\_i reg2\_gpio\_11\_i reg2\_gpio\_10\_i reg2\_gpio\_9\_i reg2\_gpio\_8\_i reg2\_gpio\_7\_i reg2\_gpio\_6\_i reg2\_gpio\_5\_i reg2\_gpio\_4\_i reg2\_gpio\_3\_i reg2\_gpio\_2\_i reg2\_gpio\_1\_i reg2\_gpio\_0\_i

位	名称	权限	复位值	描述
31	reg2_gpio_31_i	r	0	Register Controlled GPIO Input value
30	reg2_gpio_30_i	r	0	Register Controlled GPIO Input value
29	reg2_gpio_29_i	r	0	Register Controlled GPIO Input value
28	reg2_gpio_28_i	r	0	Register Controlled GPIO Input value
27	reg2_gpio_27_i	r	0	Register Controlled GPIO Input value
26	reg2_gpio_26_i	r	0	Register Controlled GPIO Input value
25	reg2_gpio_25_i	r	0	Register Controlled GPIO Input value
24	reg2_gpio_24_i	r	0	Register Controlled GPIO Input value
23	reg2_gpio_23_i	r	0	Register Controlled GPIO Input value
22	reg2_gpio_22_i	r	0	Register Controlled GPIO Input value
21	reg2_gpio_21_i	r	0	Register Controlled GPIO Input value
20	reg2_gpio_20_i	r	0	Register Controlled GPIO Input value
19	reg2_gpio_19_i	r	0	Register Controlled GPIO Input value
18	reg2_gpio_18_i	r	0	Register Controlled GPIO Input value
17	reg2_gpio_17_i	r	0	Register Controlled GPIO Input value
16	reg2_gpio_16_i	r	0	Register Controlled GPIO Input value
15	reg2_gpio_15_i	r	0	Register Controlled GPIO Input value
14	reg2_gpio_14_i	r	0	Register Controlled GPIO Input value
13	reg2_gpio_13_i	r	0	Register Controlled GPIO Input value
12	reg2_gpio_12_i	r	0	Register Controlled GPIO Input value
11	reg2_gpio_11_i	r	0	Register Controlled GPIO Input value
10	reg2_gpio_10_i	r	0	Register Controlled GPIO Input value
9	reg2_gpio_9_i	r	0	Register Controlled GPIO Input value
8	reg2_gpio_8_i	r	0	Register Controlled GPIO Input value
7	reg2_gpio_7_i	r	0	Register Controlled GPIO Input value
6	reg2_gpio_6_i	r	0	Register Controlled GPIO Input value
5	reg2_gpio_5_i	r	0	Register Controlled GPIO Input value
4	reg2_gpio_4_i	r	0	Register Controlled GPIO Input value
3	reg2_gpio_3_i	r	0	Register Controlled GPIO Input value
2	reg2_gpio_2_i	r	0	Register Controlled GPIO Input value
1	reg2_gpio_1_i	r	0	Register Controlled GPIO Input value

位	名称	权限	复位值	描述
0	reg2_gpio_0_i	r	0	Register Controlled GPIO Input value

#### 4.8.37 gpio\_cfg129

地址: 0x20000ac8

位	名称	权限	复位值	描述
31	RSVD			
30	RSVD			
29	RSVD			
28	RSVD			
27	RSVD			
26	RSVD			
25	RSVD			
24	RSVD			
23	RSVD			
22	RSVD			
21	RSVD			
20	RSVD			
19	RSVD			
18	RSVD			
17	RSVD			
16	RSVD			
15	14	r	0	Register Controlled GPIO Input value
14	13	r	0	Register Controlled GPIO Input value
13	12	r	0	Register Controlled GPIO Input value
12	11	r	0	Register Controlled GPIO Input value
11	10	r	0	Register Controlled GPIO Input value
10	9	r	0	Register Controlled GPIO Input value
9	8	r	0	Register Controlled GPIO Input value
8	7	r	0	Register Controlled GPIO Input value
7	6	r	0	Register Controlled GPIO Input value
6	5	r	0	Register Controlled GPIO Input value
5	4	r	0	Register Controlled GPIO Input value
4	3	r	0	Register Controlled GPIO Input value
3	2	r	0	Register Controlled GPIO Input value
2	1	r	0	Register Controlled GPIO Input value
1	0	r	0	Register Controlled GPIO Input value
0	reg2_gpio_32_i	r	0	Register Controlled GPIO Input value
	reg2_gpio_33_i	r	0	Register Controlled GPIO Input value
	reg2_gpio_34_i	r	0	Register Controlled GPIO Input value

#### 4.8.38 gpio\_cfg136

地址: 0x20000ae4

位	名称	权限	复位值	描述
31	reg2_gpio_31_o	r	0	Register Controlled GPIO Output value
30	reg2_gpio_30_o	r	0	Register Controlled GPIO Output value
29	reg2_gpio_29_o	r	0	Register Controlled GPIO Output value
28	reg2_gpio_28_o	r	0	Register Controlled GPIO Output value
27	reg2_gpio_27_o	r	0	Register Controlled GPIO Output value
26	reg2_gpio_26_o	r	0	Register Controlled GPIO Output value
25	reg2_gpio_25_o	r	0	Register Controlled GPIO Output value
24	reg2_gpio_24_o	r	0	Register Controlled GPIO Output value
23	reg2_gpio_23_o	r	0	Register Controlled GPIO Output value
22	reg2_gpio_22_o	r	0	Register Controlled GPIO Output value
21	reg2_gpio_21_o	r	0	Register Controlled GPIO Output value
20	reg2_gpio_20_o	r	0	Register Controlled GPIO Output value
19	reg2_gpio_19_o	r	0	Register Controlled GPIO Output value
18	reg2_gpio_18_o	r	0	Register Controlled GPIO Output value
17	reg2_gpio_17_o	r	0	Register Controlled GPIO Output value
16	reg2_gpio_16_o	r	0	Register Controlled GPIO Output value
15	reg2_gpio_15_o	r	0	Register Controlled GPIO Output value
14	reg2_gpio_14_o	r	0	Register Controlled GPIO Output value
13	reg2_gpio_13_o	r	0	Register Controlled GPIO Output value
12	reg2_gpio_12_o	r	0	Register Controlled GPIO Output value
11	reg2_gpio_11_o	r	0	Register Controlled GPIO Output value
10	reg2_gpio_10_o	r	0	Register Controlled GPIO Output value
9	reg2_gpio_9_o	r	0	Register Controlled GPIO Output value
8	reg2_gpio_8_o	r	0	Register Controlled GPIO Output value
7	reg2_gpio_7_o	r	0	Register Controlled GPIO Output value
6	reg2_gpio_6_o	r	0	Register Controlled GPIO Output value
5	reg2_gpio_5_o	r	0	Register Controlled GPIO Output value
4	reg2_gpio_4_o	r	0	Register Controlled GPIO Output value
3	reg2_gpio_3_o	r	0	Register Controlled GPIO Output value
2	reg2_gpio_2_o	r	0	Register Controlled GPIO Output value
1	reg2_gpio_1_o	r	0	Register Controlled GPIO Output value
0	reg2_gpio_0_o	r	0	Register Controlled GPIO Output value

位	名称	权限	复位值	描述
31	reg2_gpio_31_o	r/w	0	Register Controlled GPIO Output Value
30	reg2_gpio_30_o	r/w	0	Register Controlled GPIO Output Value
29	reg2_gpio_29_o	r/w	0	Register Controlled GPIO Output Value
28	reg2_gpio_28_o	r/w	0	Register Controlled GPIO Output Value
27	reg2_gpio_27_o	r/w	0	Register Controlled GPIO Output Value
26	reg2_gpio_26_o	r/w	0	Register Controlled GPIO Output Value
25	reg2_gpio_25_o	r/w	0	Register Controlled GPIO Output Value
24	reg2_gpio_24_o	r/w	0	Register Controlled GPIO Output Value
23	reg2_gpio_23_o	r/w	0	Register Controlled GPIO Output Value
22	reg2_gpio_22_o	r/w	0	Register Controlled GPIO Output Value
21	reg2_gpio_21_o	r/w	0	Register Controlled GPIO Output Value
20	reg2_gpio_20_o	r/w	0	Register Controlled GPIO Output Value
19	reg2_gpio_19_o	r/w	0	Register Controlled GPIO Output Value
18	reg2_gpio_18_o	r/w	0	Register Controlled GPIO Output Value
17	reg2_gpio_17_o	r/w	0	Register Controlled GPIO Output Value
16	reg2_gpio_16_o	r/w	0	Register Controlled GPIO Output Value
15	reg2_gpio_15_o	r/w	0	Register Controlled GPIO Output Value
14	reg2_gpio_14_o	r/w	0	Register Controlled GPIO Output Value
13	reg2_gpio_13_o	r/w	0	Register Controlled GPIO Output Value
12	reg2_gpio_12_o	r/w	0	Register Controlled GPIO Output Value
11	reg2_gpio_11_o	r/w	0	Register Controlled GPIO Output Value
10	reg2_gpio_10_o	r/w	0	Register Controlled GPIO Output Value
9	reg2_gpio_9_o	r/w	0	Register Controlled GPIO Output Value
8	reg2_gpio_8_o	r/w	0	Register Controlled GPIO Output Value
7	reg2_gpio_7_o	r/w	0	Register Controlled GPIO Output Value
6	reg2_gpio_6_o	r/w	0	Register Controlled GPIO Output Value
5	reg2_gpio_5_o	r/w	0	Register Controlled GPIO Output Value
4	reg2_gpio_4_o	r/w	0	Register Controlled GPIO Output Value
3	reg2_gpio_3_o	r/w	0	Register Controlled GPIO Output Value
2	reg2_gpio_2_o	r/w	0	Register Controlled GPIO Output Value
1	reg2_gpio_1_o	r/w	0	Register Controlled GPIO Output Value

位	名称	权限	复位值	描述
0	reg2_gpio_0_o	r/w	0	Register Controlled GPIO Output Value

#### 4.8.39 gpio\_cfg137

地址: 0x20000ae8

位	名称	权限	复位值	描述
31	RSVD			
30	RSVD			
29	RSVD			
28	RSVD			
27	RSVD			
26	RSVD			
25	RSVD			
24	RSVD			
23	RSVD			
22	RSVD			
21	RSVD			
20	RSVD			
19	RSVD			
18	RSVD			
17	RSVD			
16	RSVD			
15	RSVD			
14	RSVD			
13	RSVD			
12	RSVD			
11	RSVD			
10	RSVD			
9	RSVD			
8	RSVD			
7	RSVD			
6	RSVD			
5	RSVD			
4	RSVD			
3	RSVD			
2	RSVD			
1	RSVD			
0	RSVD			
	reg2_gpio_32_o			
	reg2_gpio_33_o			
	reg2_gpio_34_o			

#### 4.8.40 gpio\_cfg138

地址: 0x20000aec

位	名称	权限	复位值	描述
31	reg2_gpio_31_set			
30	reg2_gpio_30_set			
29	reg2_gpio_29_set			
28	reg2_gpio_28_set			
27	reg2_gpio_27_set			
26	reg2_gpio_26_set			
25	reg2_gpio_25_set			
24	reg2_gpio_24_set			
23	reg2_gpio_23_set			
22	reg2_gpio_22_set			
21	reg2_gpio_21_set			
20	reg2_gpio_20_set			
19	reg2_gpio_19_set			
18	reg2_gpio_18_set			
17	reg2_gpio_17_set			
16	reg2_gpio_16_set			
15	reg2_gpio_15_set			
14	reg2_gpio_14_set			
13	reg2_gpio_13_set			
12	reg2_gpio_12_set			
11	reg2_gpio_11_set			
10	reg2_gpio_10_set			
9	reg2_gpio_9_set			
8	reg2_gpio_8_set			
7	reg2_gpio_7_set			
6	reg2_gpio_6_set			
5	reg2_gpio_5_set			
4	reg2_gpio_4_set			
3	reg2_gpio_3_set			
2	reg2_gpio_2_set			
1	reg2_gpio_1_set			
0	reg2_gpio_0_set			

位	名称	权限	复位值	描述
31	reg2_gpio_31_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
30	reg2_gpio_30_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
29	reg2_gpio_29_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
28	reg2_gpio_28_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
27	reg2_gpio_27_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
26	reg2_gpio_26_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
25	reg2_gpio_25_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
24	reg2_gpio_24_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
23	reg2_gpio_23_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
22	reg2_gpio_22_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
21	reg2_gpio_21_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
20	reg2_gpio_20_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect

位	名称	权限	复位值	描述
19	reg2_gpio_19_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
18	reg2_gpio_18_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
17	reg2_gpio_17_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
16	reg2_gpio_16_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
15	reg2_gpio_15_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
14	reg2_gpio_14_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
13	reg2_gpio_13_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
12	reg2_gpio_12_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
11	reg2_gpio_11_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
10	reg2_gpio_10_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
9	reg2_gpio_9_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
8	reg2_gpio_8_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect

位	名称	权限	复位值	描述
7	reg2_gpio_7_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
6	reg2_gpio_6_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
5	reg2_gpio_5_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
4	reg2_gpio_4_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
3	reg2_gpio_3_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
2	reg2_gpio_2_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
1	reg2_gpio_1_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
0	reg2_gpio_0_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect

#### 4.8.41 gpio\_cfg139

地址: 0x20000af0

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	reg2_gpio_34_set	reg2_gpio_33_set	reg2_gpio_32_set

位	名称	权限	复位值	描述
31:3	RSVD			
2	reg2_gpio_34_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
1	reg2_gpio_33_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
0	reg2_gpio_32_set	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will set GPIO output value to 1,when set/clr at the same time, only set take effect
-1:32	RSVD			
31	reg2_gpio_31_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
30	reg2_gpio_30_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
29	reg2_gpio_29_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
28	reg2_gpio_28_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
27	reg2_gpio_27_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
26	reg2_gpio_26_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
25	reg2_gpio_25_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
24	reg2_gpio_24_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
23	reg2_gpio_23_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect

位	名称	权限	复位值	描述
22	reg2_gpio_22_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
21	reg2_gpio_21_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
20	reg2_gpio_20_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
19	reg2_gpio_19_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
18	reg2_gpio_18_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
17	reg2_gpio_17_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
16	reg2_gpio_16_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
15	reg2_gpio_15_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
14	reg2_gpio_14_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
13	reg2_gpio_13_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
12	reg2_gpio_12_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
11	reg2_gpio_11_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect

位	名称	权限	复位值	描述
10	reg2_gpio_10_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
9	reg2_gpio_9_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
8	reg2_gpio_8_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
7	reg2_gpio_7_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
6	reg2_gpio_6_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
5	reg2_gpio_5_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
4	reg2_gpio_4_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
3	reg2_gpio_3_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
2	reg2_gpio_2_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
1	reg2_gpio_1_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
0	reg2_gpio_0_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect

#### 4.8.42 gpio\_cfg141

地址: 0x20000af8

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

reg2\_gpio\_34\_clr reg2\_gpio\_33\_clr reg2\_gpio\_32\_clr

位	名称	权限	复位值	描述
31:3	RSVD			
2	reg2_gpio_34_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
1	reg2_gpio_33_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect
0	reg2_gpio_32_clr	w1p	0	When SWGPIO @ Set/Clear Mode Set this bit will clear GPIO output value to 0,when set/clr at the same time, only set take effect

#### 4.8.43 gpio\_cfg142

地址: 0x20000afc

cr_code1_high_time								cr_code0_high_time							
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_code_total_time								RSVD	RSVD	RSVD	RSVD	cr_invert_code0_high	cr_gpio_tx_en	cr_invert_code1_high	
RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD								

位	名称	权限	复位值	描述
31:24	cr_code1_high_time	r/w	8'd200	Used to generate Code 1 Duty Cycle Waveform (in units of xclk (XTAL or RC32M) clock cycle) waveform keep high during cr_code1_high_time waveform keep low during cr_code1_low_time (cr_code_total_time - cr_code1_high_time )
23:16	cr_code0_high_time	r/w	8'd200	Used to generate Code 0 Duty Cycle Waveform (in units of xclk (XTAL or RC32M) clock cycle) waveform keep high during cr_code0_high_time waveform keep low during cr_code0_low_time (cr_code_total_time - cr_code0_high_time)
15:7	cr_code_total_time	r/w	9'd400	Used to define Code0/Code1 total waveform time
6:3	RSVD			
2	cr_invert_code1_high	r/w	1'b0	1: cr_code1_high_time -> cr_code1_low_time
1	cr_invert_code0_high	r/w	1'b0	1: cr_code0_high_time -> cr_code0_low_time
0	cr_gpio_tx_en	r/w	1'b0	Enable GPIO DMA OUT/GPIO DMA Latch

#### 4.8.44 gpio\_cfg143

地址: 0x20000b00

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_tx_fifo_th															
cr_gpio_tx_fer_en	cr_gpio_tx_fifo_en	cr_gpio_tx_end_en	r_gpio_tx_fer_int	r_gpio_tx_fifo_int	r_gpio_tx_end_int	cr_gpio_tx_fer_mask	cr_gpio_tx_fifo_mask	cr_gpio_tx_end_mask							
gpio_tx_fifo_cnt	RSVD		cr_gpio_dma_park_value	cr_gpio_tx_end_clr	cr_gpio_tx_fifo_overflow	cr_gpio_tx_fifo_underflow	cr_gpio_dma_tx_en	cr_gpio_dma_out_sel_latch							

位	名称	权限	复位值	描述
31	cr_gpio_tx_fer_en	r/w	1'b1	Interrupt enable of gpio_tx_fer_int (GPIO DMA FIFO Underflow or Overflow)
30	cr_gpio_tx_fifo_en	r/w	1'b1	Interrupt enable of gpio_tx_fifo_int
29	cr_gpio_tx_end_en	r/w	1'b1	Interrupt enable of gpio_tx_end_int
28	r_gpio_tx_fer_int	r	1'b0	GPIO TX FIFO error interrupt, auto-cleared when FIFO overflow/underflow error flag is cleared
27	r_gpio_tx_fifo_int	r	1'b0	GPIO TX FIFO ready (tx_fifo_cnt > tx_fifo_th) interrupt, auto-cleared when data is pushed
26	r_gpio_tx_end_int	r	1'b0	GPIO TX END Interrupt (GPIO DMA FIFO Empty)
25	cr_gpio_tx_fer_mask	r/w	1'b1	Interrupt mask of gpio_tx_fer_int
24	cr_gpio_tx_fifo_mask	r/w	1'b1	Interrupt mask of gpio_tx_fifo_int
23	cr_gpio_tx_end_mask	r/w	1'b1	Interrupt mask of gpio_tx_end_int
22:16	cr_gpio_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	gpio_tx_fifo_cnt	r	8'd128	TX FIFO available count
7	cr_gpio_dma_park_value	r/w	1'b0	1: park at high level when TX FIFO empty 0: park at low level when TX FIFO empty
6	RSVD			
5	gpio_tx_fifo_underflow	r	1'b0	Underflow flag of TX FIFO, can be cleared by tx_fifo_clr
4	gpio_tx_fifo_overflow	r	1'b0	Overflow flag of TX FIFO, can be cleared by tx_fifo_clr
3	gpio_tx_end_clr	w1c	1'b0	Interrupt clear of gpio_tx_end_int
2	gpio_tx_fifo_clr	w1c	1'b0	Clear signal of TX FIFO
1	cr_gpio_dma_out_sel_latch	r/w	1'b0	1 : select latch format (8bit set/8bit clr) , 0 : select 16bit output value
0	cr_gpio_dma_tx_en	r/w	1'b0	Enable signal of dma_tx_req/ack interface

#### 4.8.45 gpio\_cfg144

地址: 0x20000b04

| 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    |      |      |

gpio\_tx\_data\_to\_fifo

位	名称	权限	复位值	描述
31:16	RSVD			
15:0	gpio_tx_data_to_fifo	w	x	

## 5.1 简介

芯片内置一个 12bits 的逐次逼近式模拟数字转换器 (ADC)，支持 12 路外部模拟输入和若干内部模拟信号选择。ADC 可以工作在单次转换和多通道扫描两种模式下，转换结果为 12/14/16bits 左对齐模式。ADC 拥有深度为 32 的 FIFO，支持多种中断，支持 DMA 操作。ADC 除了用于普通模拟信号测量外，还可以用于测量供电电压，此外 ADC 还可以通过测量内/外部二极管电压用于温度检测。

## 5.2 主要特点

- 高性能
  - 可以选择 12-bit, 14-bit, 16-bit 转换结果输出
  - ADC 转换时间最快 0.5us (12-bit 转换结果)
  - 支持 2.0V, 3.3V 可选参考电压
  - 支持 DMA 将转换结果搬运到内存
  - 支持单通道转换和多通道扫描两种模式
  - 支持单端与差分两种输入模式
  - 支持抖动补偿
  - 支持用户自行设定转换结果偏移值
  - 扫描模式时钟最大支持 1M，非扫描模式支持 2M
- 模拟通道数
  - 12 路外部模拟通道
  - 2 路 DAC 内部通道

- 1 路 VBAT/2 通道
- 1 路 TSEN 通道

### 5.3 功能描述

ADC 模块基本框图如图所示。

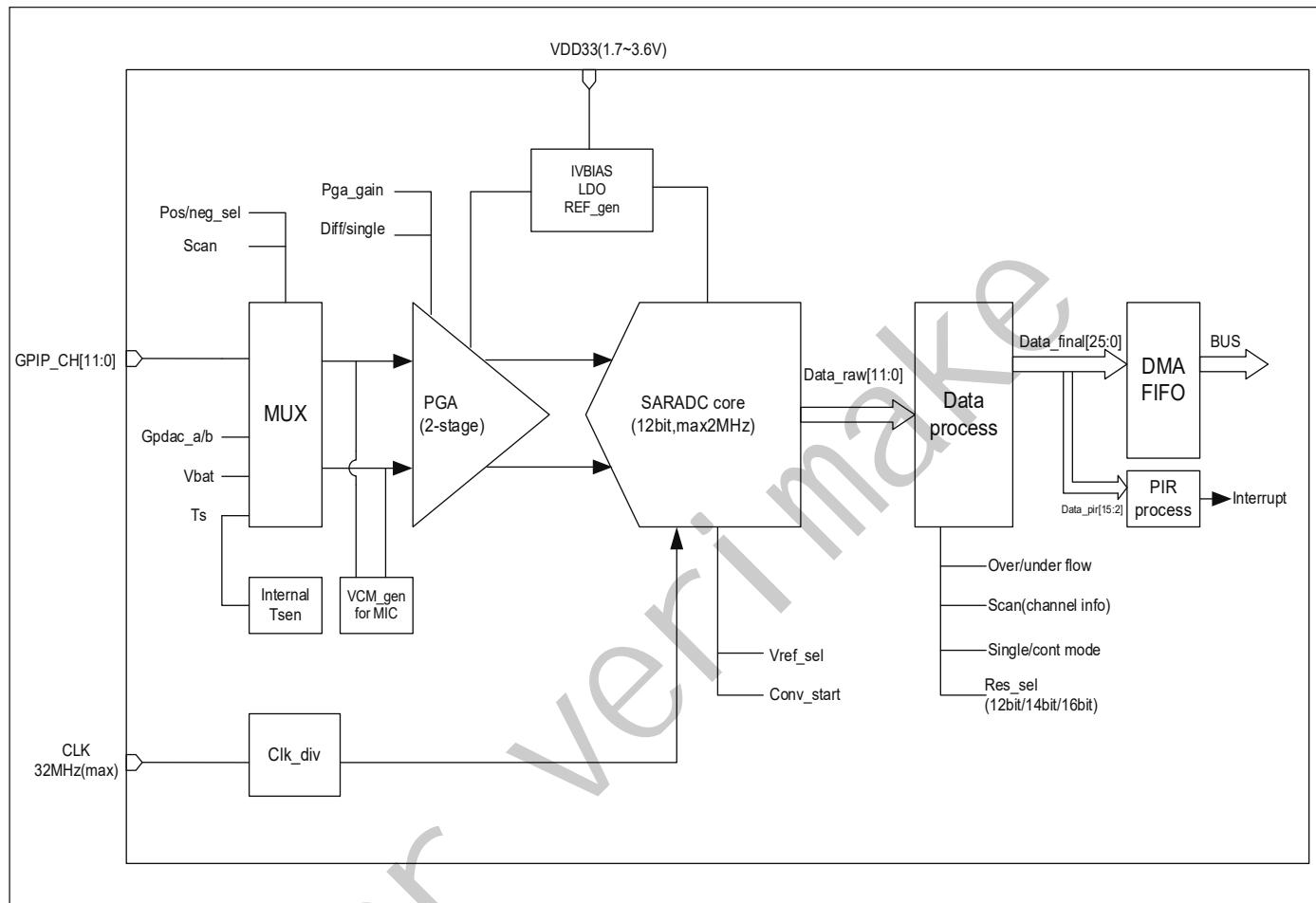


图 5.1: ADC 基本框图

ADC 模块包含五大部分，分别为前端输入通道选择器，程控放大器，ADC 采样模块，数据处理模块以及 FIFO。输入通道选择器用于选择需要采样的通道，既包含外部模拟信号，也包含内部模拟信号，程控放大器用于对输入信号做进一步处理，可以根据输入信号的特点，比如直流，交流，进行设定，以便得到更准确的转换值。ADC 采样模块是最主要的功能模块，实现通过逐次比较的方式，得到模拟信号到数字信号的转换。转换的结果为 12bit，数据处理模块负责将转换的结果进一步处理，包括添加通道信息等。最后得到的数据会推送到最后端的 FIFO 中。

### 5.3.1 ADC 引脚和内部信号

表 5.1: ADC 内部信号

内部信号	信号类型	信号描述
VBAT/2	Input	从电源引脚分压过来的电压信号
TSEN	Input	内部温度传感器输出电压
VREF	Input	内部模拟模块参考电压
DACOUTA	Input	DAC 模块输出
DACOUTB	Input	DAC 模块输出

表 5.2: ADC 外部引脚

外部引脚	信号类型	信号描述
VDDA	Input	模拟模块供电电压正极
VSSA	Input	模拟模块供电地
ADC_CHX	Input	模拟输入引脚, 总共 12 路

### 5.3.2 ADC 通道

ADC 采样的可以选择的通道包括外部模拟引脚的输入信号和芯片内部可选信号，具体包括：

- ADC CH0
- ADC CH1
- ADC CH2
- ADC CH3
- ADC CH4
- ADC CH5
- ADC CH6
- ADC CH7
- ADC CH8
- ADC CH9
- ADC CH10

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

需要注意的是，如果选择 VBAT/2 或 TSEN 作为输入待采信号，需要把 `gpadc_vbat_en` 或 `gpadc_ts_en` 置为 1。ADC 模块可以支持单端输入或者差分输入，如果是单端输入模式，负极输入通道需要选择 GND。

### 5.3.3 ADC 时钟

ADC 模块的工作时钟来源如下图所示。

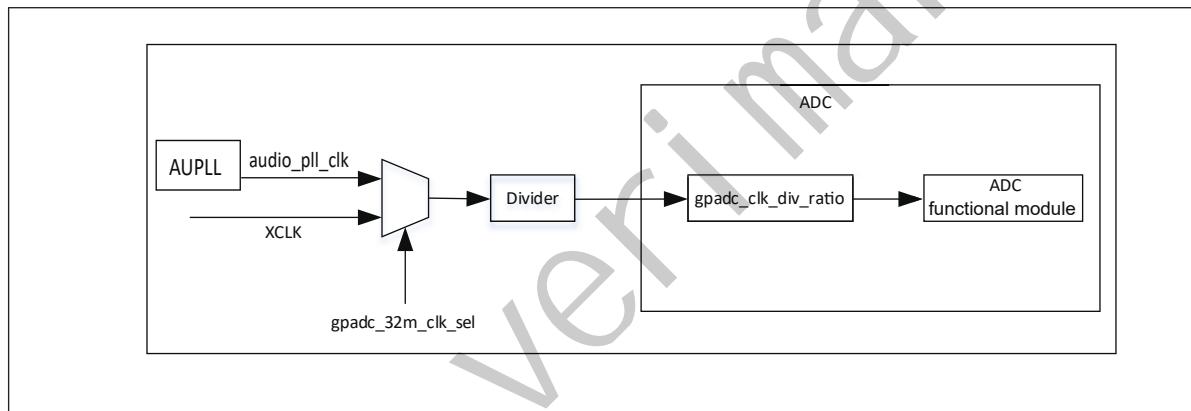


图 5.2: ADC 时钟

ADC 的时钟源可以选择来自 AUPLL 分频的 `audio_pll_clk`, XTAL 或者内部 RC32M, 时钟源的选择在 GLB 模块中设定, 同时 GLB 模块也提供了时钟分频。例如: ADC 的时钟源选择 XTAL, 时钟分频是 0, 到达 ADC 模块的时钟是 40M。在 ADC 模块内部, 提供了一个时钟分频, 如果选择 20 分频, 则 ADC 模块内部的时钟则是 2M。用户可以根据实际采样需求, 自行调整 ADC 的时钟源和各个分频系数。`gpadc_32m_clk_div` 分频寄存器宽度为 6bits, 最大分频为 64, 分频公式为  $f_{out} = f_{source}/(gpadc\_32m\_clk\_div + 1)$ 。`gpadc_clk_div_ratio` 分频寄存器位于 ADC 模块内部, 宽度为 3bits, 其分频值定义如下:

- 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

### 5.3.4 ADC 转换模式

ADC 支持单通道转换和扫描转换两种模式，在单通道转换模式下，用户需要通过 `gpadc_pos_sel` 选择正极输入通道，通过 `gpadc_neg_sel` 选择负极输入通道，同时把 `gpadc_cont_conv_en` 控制位设置为 0，表示单通道转换，然后设置 `gpadc_conv_start` 控制位启动转换即可。

在扫描转换模式下，`gpadc_cont_conv_en` 控制位需要设置为 1，ADC 根据 `gpadc_scan_length` 控制位设定的转换通道个数，依次按照 `gpadc_reg_scn_posX(X=1, 2)` 和 `gpadc_reg_scn_negX(X=1, 2)` 寄存器组所设定的通道顺序，逐个进行转换，转换的结果会自动推入 ADC 的 FIFO。`gpadc_reg_scn_posX(X=1, 2)` 和 `gpadc_reg_scn_negX(X=1, 2)` 寄存器组所设定的通道可以相同，这也就意味着用户可以实现对一个通道进行多次采样转换。

ADC 的转换结果一般都是放入 FIFO 中。用户需要根据实际转换通道数，设定 FIFO 接收数据阈值中断，通过 FIFO 的阈值中断，作为 ADC 转换完成中断。

### 5.3.5 ADC 结果

`gpadc_raw_data` 寄存器存放了 ADC 的原始结果，在单端模式下，数据有效位是 12bits，无符号位，在差分模式下，最高位为符号位，剩下 11bits 代表转换的结果。

`gpadc_data_out` 寄存器存放了 ADC 的结果，这个结果里包含了 ADC 结果，符号位和通道信息，数据格式如下：

表 5.3: ADC 转换结果含义

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
含义	正极通道号				负极通道号				转换结果																	

转换结果的 bit21-bit25 是正极通道号，bit16-bit20 是负极通道号，bit0-bit15 是转换的数值。

`gpadc_res_sel` 控制位可以设定转换结果的位数为 12 位，14 位，和 16 位，其中 14 位和 16 位是多次采样提高精度得到的结果，其可以设置的值如下：

- 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

ADC 转换结果为左对齐模式，当选择 12 位时，转换结果的 bit15-bit4 有效，当选择 14 位时，转换结果的 bit15-bit2 有效，当选择 16 位时，转换结果的 bit15-bit0 有效。同样，在差分模式下，最高位是符号位，也就是，当选择 14 位时，bit15 是符号位，bit14-bit2 是转换结果，bit14 是 MSB，在单端模式下，没有符号位，也就是，当选择 12 位时，bit15-bit4 是转换结果，bit15 是 MSB。

在实际使用中，ADC 的结果一般都是放入 FIFO，这在多通道扫描模式下尤为重要，所以用户一般都是从 ADC FIFO 获取转换结果，ADC FIFO 的数据格式 `gpadc_data_out` 寄存器中数据格式相同。

### 5.3.6 ADC 中断

ADC 模块在正极采样超量程和负极采样超量程时可以产生中断，可以通过 `gpadc_pos_satur_mask`,`gpadc_neg_satur_mask` 屏蔽各自中断，当中断产生时，可以通过 `gpadc_pos_satur`，和 `gpadc_neg_satur` 寄存器查询中断状态，同时可以通过 `gpadc_pos_satur_clr` 和 `gpadc_neg_satur_clr` 清除中断。该功能可以用来判断输入电压是否异常。

### 5.3.7 ADC FIFO

ADC 模块拥有深度为 32 的 FIFO，数据宽度为 26bits，当 ADC 完成转换后，会自动将结果推入到 FIFO。ADC 的 FIFO 有如下状态和中断管理功能：

- FIFO 满状态
- FIFO 非空状态
- FIFO Overrun 中断
- FIFO Underrun 中断

当中断产生时，可以通过对应的 `clear` 位将中断标志清除掉。

利用 ADC 的 FIFO 用户可以实现三种模式获取数据：查询模式，中断模式，DMA 模式

查询模式

CPU 轮询 `gpadc_rdy` 位，当该控制位置位时，说明 FIFO 中存在有效数据，CPU 可以根据 `gpadc_fifo_data_count` 获知 FIFO 数据个数并从 FIFO 读出这些数据。

中断模式

CPU 设置 `gpadc_rdy_mask` 为 0，ADC 就会在 FIFO 有数据推入的时候产生中断，用户可在中断函数中，根据 `gpadc_fifo_data_count` 获知 FIFO 数据个数并从 FIFO 读出这些数据，然后设置 `gpadc_rdy_clr` 清除中断。

DMA 模式

用户设定 `gpadc_dma_en` 控制位，可以配合 DMA 完成转换数据到内存的搬运，在使用 DMA 模式时，通过 `gpadc_fifo_thl` 设置 ADC FIFO 发送 DMA 请求的数据个数阈值，DMA 在收到请求时，会自动根据用户设定的参数，从 FIFO 搬运指定个数的结果到对应的内存。

### 5.3.8 ADC 设置流程

设置 **ADC** 时钟

根据 ADC 转换速度需求，确定 ADC 的工作时钟，设定 GLB 模块的 ADC 时钟源和分频，结合 `gpadc_clk_div_ratio`，确定最终 ADC 模块的工作时钟频率。

根据使用的通道设置 **GPIO**

根据使用的模拟引脚，确定使用的通道号，初始化对应的 GPIO 为模拟功能，需要注意的是，在设定 GPIO 为模拟输入的时候，不要设置 GPIO 的上拉或者下拉，需要设置为浮空输入。

设定要转换的通道

根据使用的模拟通道和转换模式，设定对应的通道寄存器，对于单通道转换，在 `gpadc_pos_sel` 和 `gpadc_neg_sel` 寄存器中设置转换的通道信息。对于多通道扫描模式，根据要扫描通道数目和扫描顺序，设定 `gpadc_scan_length`,`gpadc_reg_scn_posX` 和 `gpadc_reg_scn_negX`。

设定数据读取方式

根据 ADC FIFO 介绍的读取数据方式，选择使用的模式，设置对应的寄存器。如果使用 DMA，同样需要配置 DMA 的一个通道，配合 ADC FIFO 完成数据的搬运。

启动转换

最后设置 `gpadc_res_sel` 选择数据转换结果的精度，最后设置 `gpadc_global_en=1`, `gpadc_conv_start=1` 就可以启动 ADC 开始转换。当转换完成，需要再次转换时，需要将 `gpadc_conv_start` 设置为 0，再设置为 1，以便再次触发转换。

### 5.3.9 VBAT 测量

这里的 VBAT/2 测量的是芯片 VDD33 的电压，而不是外部的比如锂电池的电压，如果需要测量锂电池等供电源头的电压，可以将电压分压，然后输入 ADC 的 GPIO 模拟通道，测量 VDD33 的电压可以减少 GPIO 的使用。

ADC 模块测量的 VBAT/2 电压是经过分压的，实际输入到 ADC 模块的电压是 VDD33 的一半，即  $VBAT/2=VDD33/2$ 。由于电压经过分压，为了得到较高的精确度，建议 ADC 的参考电压选择 2.0V，采用单端模式，正极输入电压选择 VBAT/2，负极输入电压选择 GND，同时将 `gpadc_vbat_en` 设置为 1，启动转换后，将对应的转换结果乘以 2 就可以得到 VDD33 电压。

### 5.3.10 TSEN 测量

ADC 可以测量内部二极管或者外部二极管电压值，而二极管的压差和温度有关，所以通过测量二极管的电压，可以计算得到环境温度，我们称之为 Temperature Sensor，简称 TSEN。

TSEN 的测试原理是通过一个二极管上面测量两次不同大小的电流产生的电压差  $\Delta V$  随着温度的变化拟合的曲线，无论外部或者内部二极管的测量，最终输出的值和温度有关，都可以表示成  $\Delta(ADC_{out})=7.753T+X$ ，当我们知道了电压值，也就知道了温度 T。这里的 X 是一个偏移值，可以作为标准值，在实际使用前，我们需要确定 X。芯片厂商会在芯片出厂前，在标准温度下，例如室温 25 度，测量  $\Delta(ADC_{out})$ ，从而得到 X。在用户使用的时候，只要根据公式  $T=[\Delta(ADC_{out})-X]/7.753$ ，就可以得到温度 T。

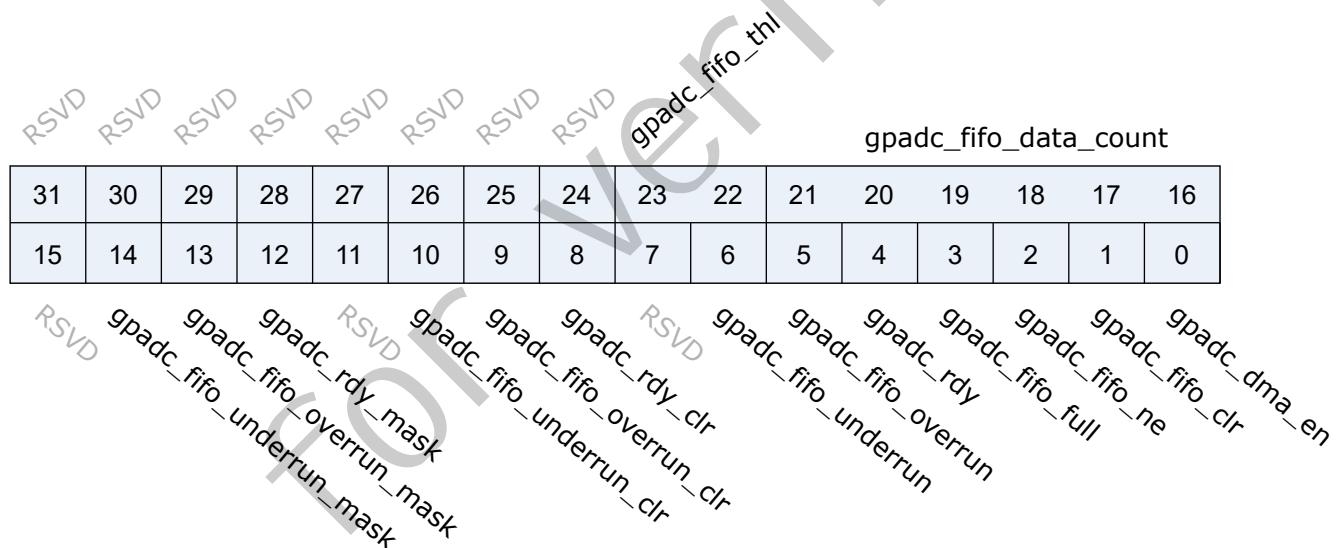
在使用 TSEN 时，建议把 ADC 设置成 16bits 模式，通过多次采样以减少误差，参考电压选择 1.8V 以提高精度，设置 `gpadc_ts_en` 为 1 以便启动 TSEN 功能，如果选择内部二极管，`gpadc_tsext_sel=0`，如果选择外部二极管，`gpadc_tsext_sel=1`，根据实际情况选择正向输入通道，如果是内部二极管，选择 TSEN 通道，如果是外部，选择对应的模拟 GPIO 通道，负极输入端选择 GND。在上述设定完毕后，设置 `gpadc_tsvbe_low=0`，启动测量，得到测量结果 V0，再设置 `gpadc_tsvbe_low=1`，启动测量，得到测量结果 V1,  $\Delta(\text{ADC\_out}) = V1 - V0$ ，根据公式  $T = [\Delta(\text{ADC\_out}) - X] / 7.753$ ，得到温度 T。

## 5.4 寄存器描述

名称	描述
<code>gpadc_config</code>	
<code>gpadc_dma_rdata</code>	

### 5.4.1 `gpadc_config`

地址: 0x20002000



位	名称	权限	复位值	描述
31:24	RSVD			
23:22	<code>gpadc_fifo_thl</code>	r/w	2'd0	fifo threshold 2'b00: 1 data 2'b01: 4 data 2'b10: 8 data 2'b11: 16 data

位	名称	权限	复位值	描述
21:16	gpadc_fifo_data_count	r	6'd0	fifo data number
15	RSVD			
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	r/w	1'b0	Write 1 to clear flag
9	gpadc_fifo_overrun_clr	r/w	1'b0	Write 1 to clear flag
8	gpadc_rdy_clr	r/w	1'b0	Write 1 to clear flag
7	RSVD			
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
0	gpadc_dma_en	r/w	1'b0	GPADC DMA enable

#### 5.4.2 gpadc\_dma\_rdata

地址: 0x20002004

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															

位	名称	权限	复位值	描述
31:26	RSVD			
25:0	gpadc_dma_rdata	r	26'd0	GPADC final conversion result stored in the FIFO

## 6.1 简介

DAC 模块是 12 位电压输出数模转换器，可与 DMA 控制器配合使用。DAC 有两个输出通道，每个通道各有一个转换器，每个通道都可以单独进行转换。可用于音频播放，变送器电压调制等应用。

## 6.2 主要特点

- DAC 调制精度为 12-bits
- DAC 的输入时钟可选为 32MHz 或者 xclk
- 每个通道都具有 DMA 功能，支持 10 种数据传输格式
- DAC 双通道单独或同时转换
- 支持双声道播放 DMA 搬运模式
- DAC 的输出引脚固定为 ChannelA 为 GPIO3, ChannelB 为 GPIO2
- 支持内部和外部输入参考电压

## 6.3 功能描述

DAC 模块基本框图如图所示。

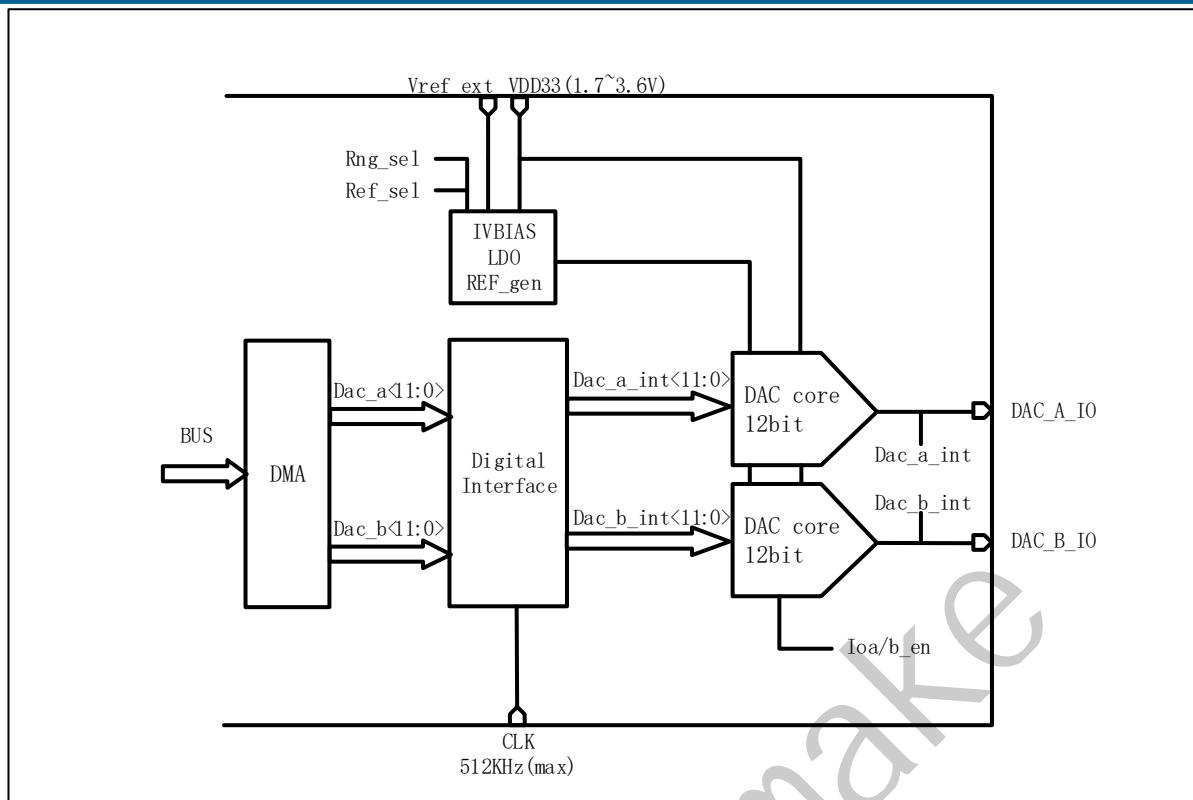


图 6.1: DAC 基本框图

DAC 模块包含两路 DAC 调制电路，以及调制模拟信号相关的电源电路，用户可以通过 Ref\_Sel 来选择 DAC 的参考电压是外部/内部，Rng\_Sel 来选择输出电压范围。DAC 的调制数据可以由 CPU 直接写入 DAC 调制寄存器（寄存器 dac\_cfg3 中的 gpdac\_a\_data、gpdac\_b\_data），也可以由 DMA 搬运至 gpdac\_dma\_wdata 寄存器。

### 6.3.1 DAC 通道使能

以使能 A 通道为例，配置流程为：

1. 将寄存器 gpdac\_config 中对应的 gpdac\_en 位置 1，使能 DAC
2. 将寄存器 dac\_cfg1 中对应的 gpdac\_a\_en 位置 1，使能 DAC A 通道转换
3. 将寄存器 dac\_cfg1 中对应的 gpdac\_ioa\_en 位置 1，使能通道 A 转换结果到 GPIO 口

### 6.3.2 DAC 数据格式

使用 DMA 方式转换数据时，共有 10 种数据传输格式，通过设置寄存器 gpdac\_dma\_config 中对应的 gpdac\_dma\_format 位数值，可以配置 gpdac\_dma\_wdata 寄存器（宽度为 32-bit）中存储的数据传输格式，其对应关系如下：

表 6.1: 数据传输格式

gpdac_dma_format 位对应的数值	gpdac_dma_wdata ( 对应 A 和 B 通道的数据传输格式 )
0	[11:0]: {A0}, {A1}, {A2} ...
1	[27:16][11:0]: {B0,A0}, {B1,A1}, {B2,A2} ...
2	[27:16][11:0]: {A1,A0}, {A3,A2}, {A5,A4} ...
4	[15:4]: {A0}, {A1}, {A2} ...
5	[31:20][15:4]: {B0,A0}, {B1,A1}, {B2,A2} ...
6	[31:20][15:4]: {A1,A0}, {A3,A2}, {A5,A4} ...
8	[31:24][23:16][15:8][7:0]: {A3,A2,A1,A0}, {A7,A6,A5,A4} ...
9	[31:24][23:16][15:8][7:0]: {B1,B0,A1,A0}, {B3,B2,A3,A2} ...
10	[31:24][23:16][15:8][7:0]: {B1,A1,B0,A0}, {B3,A3,B2,A2} ...
11	[15:8][7:0]: {B0,A0}, {B1,A1}, {B2,A2} ...

### 6.3.3 DAC 输出电压

用户可以通过设置 `dac_cfg0` 寄存器中对应的 `gpdac_ref_sel` 位数值来选择使用外部参考电压还是内部参考电压。

如果选择内部参考电压，配置以及输出电压如下表所示。如果选择外部参考电压，请将外部电压连入固定的 GPIO28。

表 6.2: 内部参考电压输出电压

gpdac_a_rng	gpdac_ref_sel	输出范围 ( V )
00	0	0.2-1
01/10	0	0.225-1.425
11	0	0.2-1.8

### 6.3.4 DAC 转换

#### 6.3.4.1 CPU 方式

使用 CPU 的方式进行数据转换，以 A 通道和 B 通道同时转换为例，配置过程如下：

1. 设置 DAC 时钟：将寄存器 `dig_clk_cfg0` 中对应的 `dig_clk_src_sel` 位数值写 1，即选择 `xclk` 作为 DAC 时钟源
2. 设置时钟分频系数：用户根据需求设置寄存器 `dig_clk_cfg0` 对应的 `dig_512k_div` 位和寄存器 `gpdac_config` 对应的 `gpdac_mode` 位的数值
3. 初始化 A 和 B 通道的 GPIO 引脚

4. 初始化并使能 DAC A 通道和 B 通道
5. 将需要转换的数据写入寄存器 `dac_cfg3` 中对应的 `gpdac_a_data` 和 `gpdac_b_data` 位，完成数据转换

#### 6.3.4.2 DMA 方式

每个 DAC 通道都具有 DMA 功能。以 A 通道使用 DMA 的方式进行数据转换为例，配置过程如下：

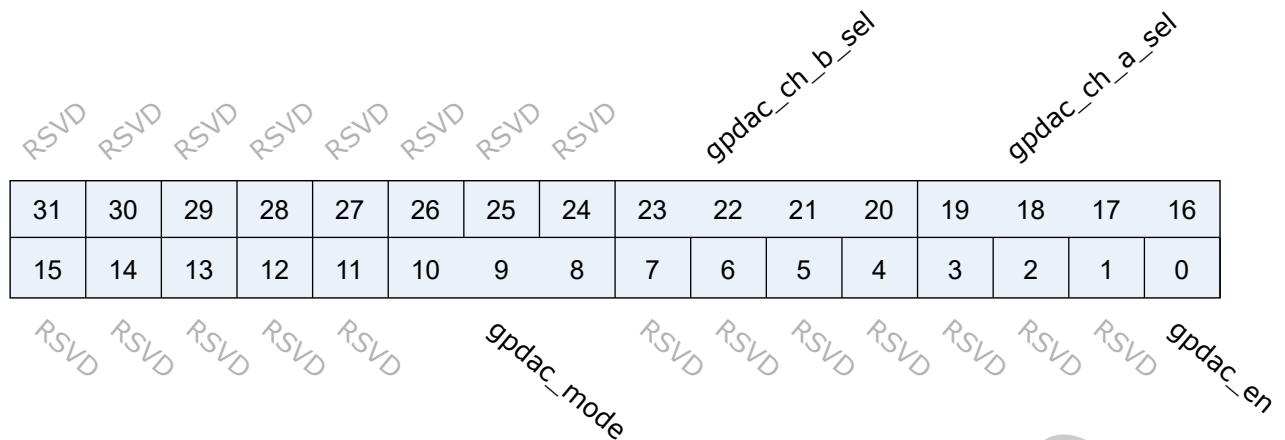
1. 设置 DAC 时钟：将寄存器 `dig_clk_cfg0` 中对应的 `dig_clk_src_sel` 位数值写 1，即选择 `xclk` 作为 DAC 时钟源
2. 设置时钟分频系数：用户根据需求设置寄存器 `dig_clk_cfg0` 对应的 `dig_512k_div` 位和寄存器 `gpdac_config` 对应的 `gpdac_mode` 位的数值
3. 初始化 A 通道的 GPIO 引脚
4. 初始化并使能 DAC A 通道
5. 初始化并使能 DMA 通道：设置 DMA 的传输数据宽度、来源地址、目标地址和数据传输长度等
6. 使能 DAC DMA 模式：将寄存器 `gpdac_dma_config` 中对应的 `gpdac_dma_tx_en` 位数值写 1
7. 将需要转换的数据写入寄存器 `gpdac_dma_wdata` 中，根据不同的数据格式作用于 A 或 B 通道，完成数据转换

### 6.4 寄存器描述

名称	描述
<code>gpdac_config</code>	
<code>gpdac_dma_config</code>	
<code>gpdac_dma_wdata</code>	

### 6.4.1 gpdac\_config

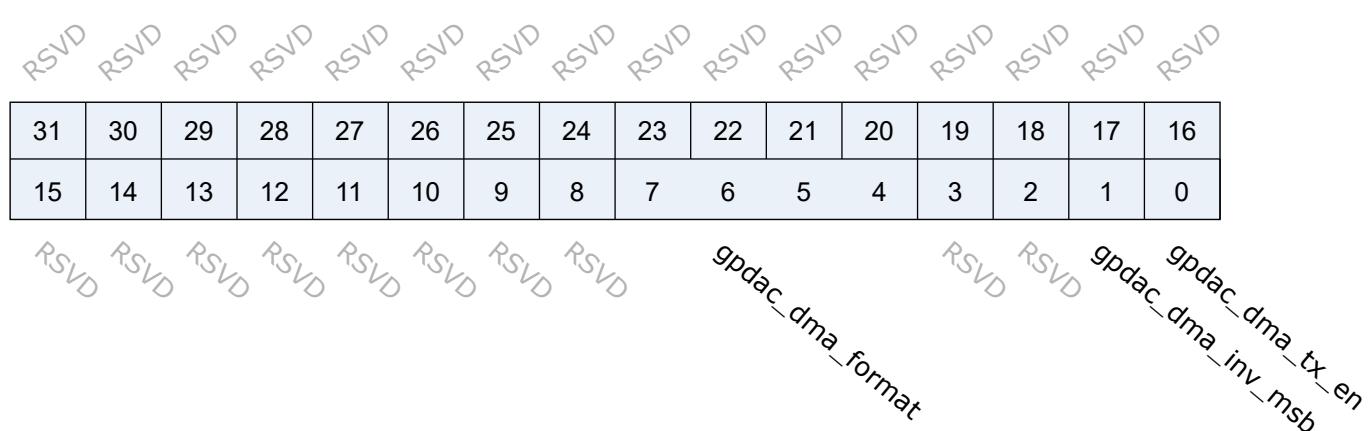
地址: 0x20002040



位	名称	权限	复位值	描述
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)
7:1	RSVD			
0	gpdac_en	r/w	0	GPDAC enable

### 6.4.2 gpdac\_dma\_config

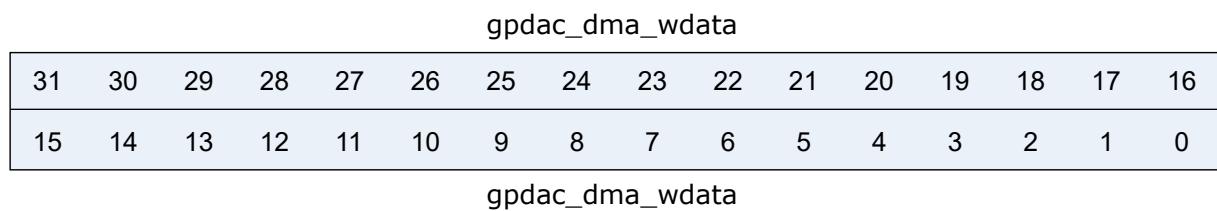
地址: 0x20002044



位	名称	权限	复位值	描述
31:8	RSVD			
7:4	gpdac_dma_format	r/w	0	DMA TX format (Data 12-bit) 0: ([11:0]) A0, A1, A2... 1: ([27:16][11:0]) B0,A0, B1,A1, B2,A2... 2: ([27:16][11:0]) A1,A0, A3,A2, A5,A4... 4: ([15:4]) A0, A1, A2... 5: ([31:20][15:4]) B0,A0, B1,A1, B2,A2... 6: ([31:20][15:4]) A1,A0, A3,A2, A5,A4... 8: ([31:24][23:16][15:8][7:0]) A3,A2,A1,A0, A7,A6,A5,A4... 9: ([31:24][23:16][15:8][7:0]) B1,B0,A1,A0, B3,B2,A3,A2... 10: ([31:24][23:16][15:8][7:0]) B1,A1,B0,A0, B3,A3,B2,A2... ... 11: ([15:8][7:0]) B0,A0, B1,A1, B2,A2...
3:2	RSVD			
1	gpdac_dma_inv_msb	r/w	0	GPDAC DMA Data Inverse MSB
0	gpdac_dma_tx_en	r/w	0	GPDAC DMA TX enable

### 6.4.3 gpdac\_dma\_wdata

地址: 0x20002048



位	名称	权限	复位值	描述
31:0	gpdac_dma_wdata	w	x	GPDAC DMA TX data

for VeriMake

## 7.1 简介

DMA(Direct Memory Access) 是一种内存存取技术，可以独立地直接读写系统内存，而无需处理器介入处理。在同等程度的处理器负担下，DMA 是一种快速的数据传送方式。1 个独立的 DMA 控制器，有 4 组独立专用通道，管理外围设备和内存之间的数据传输以提高总线效率。主要有三种类型传输包括内存至内存、内存至外设、外设至内存。并支持 LLI 链接列表功能。使用上由软件配置传输数据大小、数据源地址和目标地址。

## 7.2 主要特征

- 1 个 DMA 控制器，包含 4 组独立专用通道
- 独立控制来源与目标存取宽度 (单字节、双字节、四字节)
- 每个通道独立作为读写缓存
- 每个通道可被独立的外设硬件触发或是软件触发
- DMA 支持外设包括 UART、I2C、SPI、AUDAC、AUADC、GPIO、I2S、DBI、GPADC、GPDAC
- 八种流程控制
  - DMA 流程控制，来源内存、目标内存
  - DMA 流程控制，来源内存、目标外设
  - DMA 流程控制，来源外设、目标内存
  - DMA 流程控制，来源外设、目标外设
  - 目标外设流程控制，来源外设、目标外设
  - 目标外设流程控制，来源内存、目标外设
  - 来源外设流程控制，来源外设、目标内存
  - 来源外设流程控制，来源外设、目标外设

- 支持 LLI 链表功能，提高 DMA 效率

## 7.3 功能描述

### 7.3.1 工作原理

当一个设备试图通过总线直接向另一个设备传输数据时，它会先向 CPU 发送 DMA 请求信号。外设通过 DMA 向 CPU 提出接管总线控制权的总线请求，CPU 收到该信号后，在当前的总线周期结束后，会按 DMA 信号的优先级和提出 DMA 请求的先后顺序响应 DMA 信号。CPU 对某个设备接口响应 DMA 请求时，会让出总线控制权。于是在 DMA 控制器的管理下，外设和存储器直接进行数据交换，而无需 CPU 干预。数据传送完毕后，设备会向 CPU 发送 DMA 结束信号，交还总线控制权。

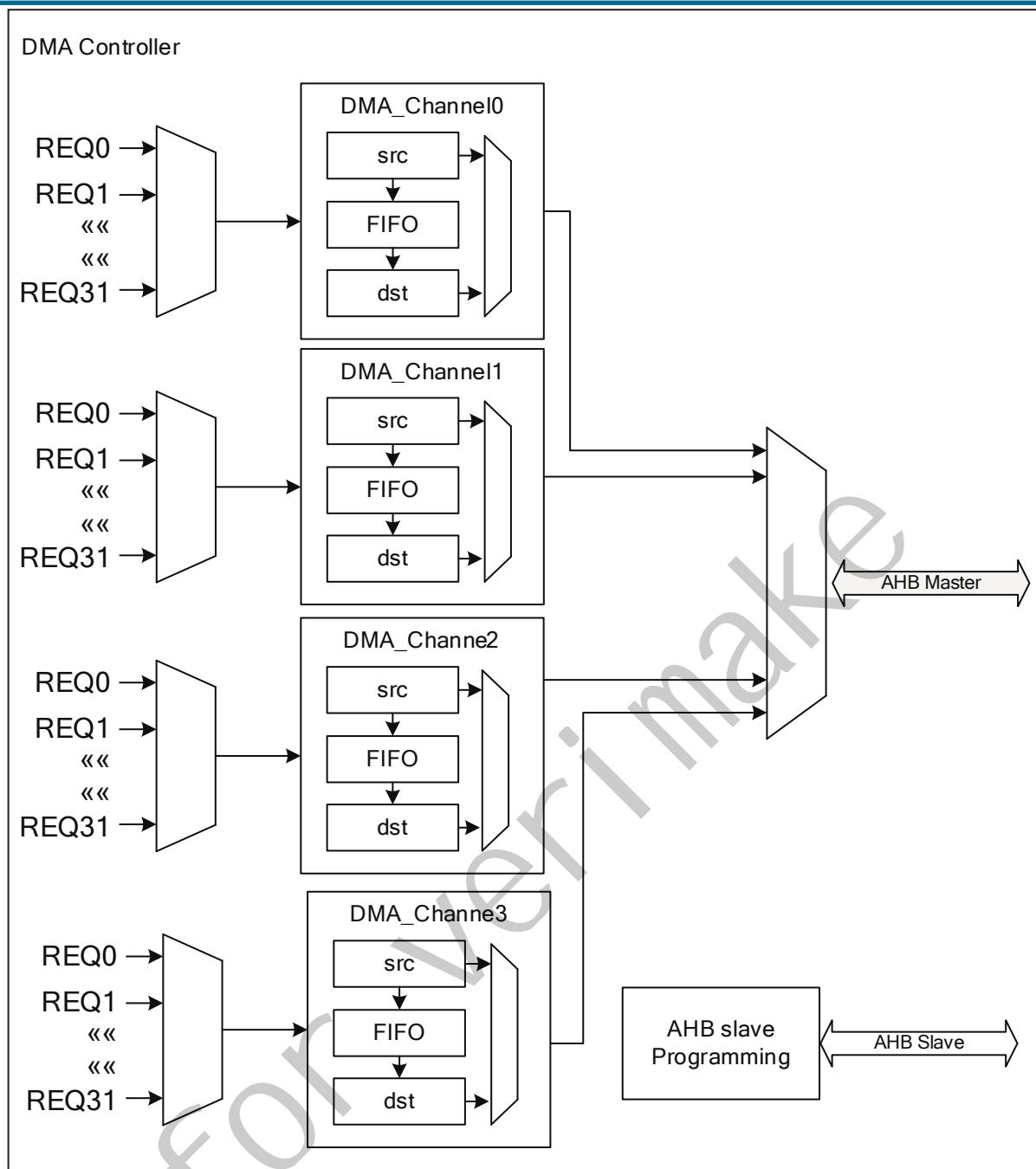


图 7.1: DMA 框图

DMA 包含一组 AHB Master 接口和一组 AHB Slave 接口。AHB Master 接口根据当前配置需求通过系统总线主动存取内存或是外设，作为数据搬移的端口。AHB Slave 接口作为配置 DMA 的接口，只支持 32-bit 存取。

### 7.3.2 DMA 通道配置

DMA 支持 4 路通道，各通道之间互不干涉，可以同时运行，下面是 DMA 通道 x 的配置过程：

1. 在 DMA\_C0SrcAddr 寄存器中设置 32-bit 来源地址
2. 在 DMA\_C0DstAddr 寄存器中设置 32-bit 目标地址
3. 地址自动累加，可通过配置 DMA\_C0Control 寄存器中的 SI(来源)、DI(目标) 设定是否开启地址自动累加模式，设置为 1 时，开启地址自动累加模式
4. 设置传输数据宽度，可通过配置 DMA\_C0Control 寄存器中的 SWidth(来源)、DWidth(目标) 位，宽度选项有单字节、双字节、四字节
5. Burst 型态，可通过配置 DMA\_C0Control 寄存器中的 SBSIZE(来源)、DBSIZE(目标) 位来设置，配置选项有 INCR1、INCR4、INCR8、INCR16
6. 需要特别注意的是所配置的组合，DMA 单笔 burst 不能超过 16 字节
7. 设置数据传输长度的范围为：0-4095

### 7.3.3 外设支持

可通过配置 SrcPeripheral(来源) 和 DstPeripheral(目标) 来决定当前 DMA 配合的外设，外设类型与配置值的对应关系如下表所示：

Value	DMA0
0	UART0_RX
1	UART0_TX
2	UART1_RX
3	UART1_TX
6	I2C0_RX
7	I2C0_TX
9	GPIO_TX
10	SPI0_RX
11	SPI0_TX
13	AUDIO_TX
14	I2C1_RX
15	I2C1_TX
16	I2S_RX
17	I2S_TX
20	DBI_TX
21	SOLO_RX
22	GPADC_RX
23	GPDAC_TX
24	FIO0_RX
25	FIO1_RX
26	FIO2_RX
27	FIO3_RX
28	FIO0_TX
29	FIO1_TX
30	FIO2_TX
31	FIO3_TX

图 7.2: 外设类型选择

以下是部分外设配置示例：

### UART 使用 DMA 传输数据

UART 发送数据包，使用 DMA 方式能大量减轻 CPU 处理的时间，使其 CPU 资源不被大量浪费，尤其在 UART 收发大量数据包（如高频率收发指令）时具有明显优势。

以 UART0 传输为例，配置过程如下：

1. 将寄存器 DMA\_C0Config 中的 SrcPeripheral 位的值设置为 1，即将 Source peripheral 设置为 UART0\_TX
2. 将寄存器 DMA\_C0Config 中的 DstPeripheral 位的值设置为 0，即将 Destination peripheral 设置为 UART0\_RX

### I2C 使用 DMA 传输数据

配置如下：

1. 将寄存器 DMA\_C0Config 中的 SrcPeripheral 位的值设置为 7，即将 Source peripheral 设置为 I2C0\_TX

2. 将寄存器 DMA\_C0Config 中的 DstPeripheral 位的值设置为 6，即将 Destination peripheral 设置为 I2C0\_RX

#### SPI 使用 DMA 传输数据

配置如下：

1. 将寄存器 DMA\_C0Config 中的 SrcPeripheral 位的值设置为 11，即将 Source peripheral 设置为 SPI0\_TX
2. 将寄存器 DMA\_C0Config 中的 DstPeripheral 位的值设置为 10，即将 Destination peripheral 设置为 SPI0\_RX

#### ADC 使用 DMA 传输数据

配置如下：

1. 将寄存器 DMA\_C0Config 中的 SrcPeripheral 位的值设置为 22，即将 Source peripheral 设置为 GPADC

### 7.3.4 链表模式

DMA 支持链表工作模式。在进行一次 DMA 读或写操作时，可以向下一条链表中填写数据，当完成当前链表的数据传输后，通过读取 DMA\_C0LLI 寄存器的数值获取下一条链表的起始地址，直接传输下一条链表中的数据。保证 DMA 传输过程中连续不间断的工作，提高 CPU 和 DMA 的效率。

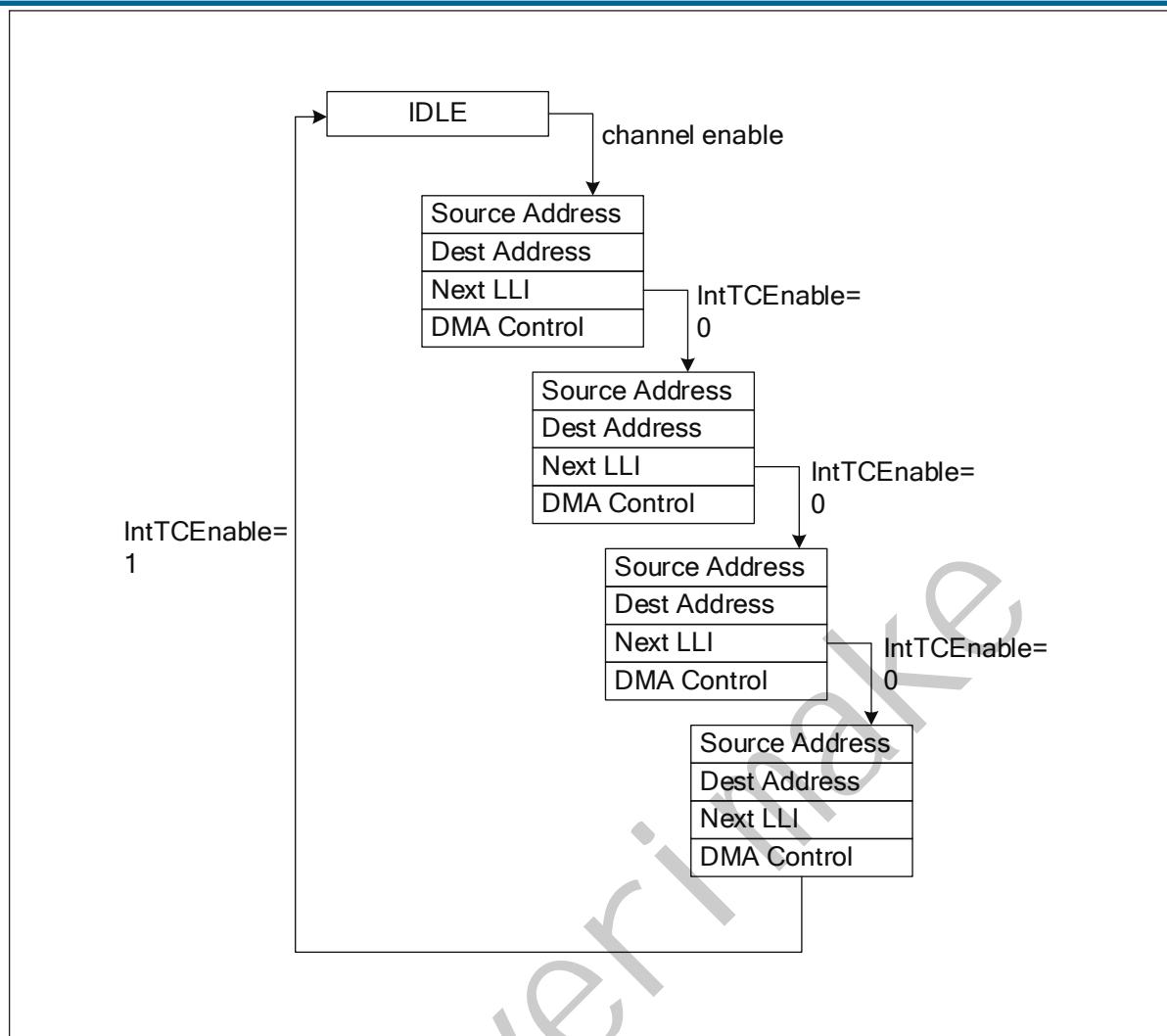


图 7.3: LLI 框架

### 7.3.5 DMA 中断

- DMA\_INT\_TCOMPLETED
  - 数据传输完成中断，当一次数据传输完毕后，会进入此中断
- DMA\_INT\_ERR
  - 数据传输出错中断，当数据传输过程中出现错误时，会进入此中断

## 7.4 传输模式

### 7.4.1 内存到内存

这个模式启动后，DMA 会根据设定好的搬移数量 (TransferSize)，将数据从来源地址搬到目标地址，传输完毕后 DMA 控制器会自动回到空闲状态，等待下一次的搬运。

具体配置流程如下：

1. 将寄存器 DMA\_C0SrcAddr 的值设置为来源的内存地址
2. 将寄存器 DMA\_C0DstAddr 的值设置为目标的内存地址
3. 选择传输模式，将寄存器 DMA\_C0Config 中的 FlowCntrl 位的值设置为 0，即选择 memory-to-memory 模式
4. 设置 DMA\_C0Control 寄存器中对应的位的数值：DI、SI 位设置为 1，开启地址自动累加模式，DWidth、SWidth 位分别设置来源和目标的传输宽度，DBSize、SBSize 位分别设置来源和目标的 burst 型态
5. 选择合适的通道，使能 DMA，完成数据传输

### 7.4.2 内存到外设

在这种工作模式下，DMA 会根据设定好的搬移数量 (TransferSize)，把数据从来源端搬至内部缓存，当缓存空间不够时自动暂停，待有足够的缓存空间时继续，直到达到设定的搬移数量。另外一方面，当目标外设请求触发会将目标配置 burst 到目标地址，直到达到设定的搬移数量，完成后自动回到空闲状态，等待下一次启动。

具体配置流程如下：

1. 将寄存器 DMA\_C0SrcAddr 的值设置为来源的内存地址
2. 将寄存器 DMA\_C0DstAddr 的值设置为目标的外设地址
3. 选择传输模式，将寄存器 DMA\_C0Config 中的 FlowCntrl 位的值设置为 1，即选择 Memory-to-peripheral 模式
4. 设置 DMA\_C0Control 寄存器中对应的位的数值：SI 位设置为 1，DI 位设置为 0，开启源地址自动累加模式，DWidth、SWidth 位分别设置来源和目标的传输宽度，DBSize、SBSize 位分别设置来源和目标的 burst 型态
5. 选择合适的通道，使能 DMA，完成数据传输

### 7.4.3 外设到内存

在这种工作模式下，当来源外设请求触发时将来源配置 burst 到缓存，直到设定的搬移数量达到停止。另外一方面，当内部缓存足够一次目标 burst 数量时，DMA 会自动将缓存的内容搬到目标地址直到达到设定的搬移数量，完成后自动回到空闲状态，等待下一次启动。

具体配置流程如下：

1. 将寄存器 DMA\_C0SrcAddr 的值设置为来源的外设地址
2. 将寄存器 DMA\_C0DstAddr 的值设置为目标的内存地址

3. 选择传输模式，将寄存器 DMA\_C0Config 中的 FlowCntrl 位的值设置为 2，即选择 Peripheral-to-memory 模式
4. 设置 DMA\_C0Control 寄存器中对应的位的数值:DI 位设置为 1, SI 位设置为 0, 开启目的地址自动累加模式,DWidth、SWidth 位分别设置来源和目标的传输宽度，DBSSize、SBSSize 位分别设置来源和目标的 burst 型态
5. 选择合适的通道，使能 DMA，完成数据传输

## 7.5 寄存器描述

名称	描述
DMA_IntStatus	
DMA_IntTCStatus	
DMA_IntTCClear	
DMA_IntErrorStatus	
DMA_IntErrClr	
DMA_RawIntTCStatus	
DMA_RawIntErrorStatus	
DMA_EnbldChns	
DMA_SoftBReq	
DMA_SoftSReq	
DMA_SoftLBReq	
DMA_SoftLSReq	
DMA_Config	
DMA_Sync	
DMA_C0SrcAddr	
DMA_C0DstAddr	
DMA_C0LLI	
DMA_C0Control	
DMA_C0Config	
DMA_C0RSVD	
DMA_C1SrcAddr	
DMA_C1DstAddr	
DMA_C1LLI	

名称	描述
DMA_C1Control	
DMA_C1Config	
DMA_C1RSVD	
DMA_C2SrcAddr	
DMA_C2DstAddr	
DMA_C2LLI	
DMA_C2Control	
DMA_C2Config	
DMA_C2RSVD	
DMA_C3SrcAddr	
DMA_C3DstAddr	
DMA_C3LLI	
DMA_C3Control	
DMA_C3Config	
DMA_C3RSVD	

### 7.5.1 DMA\_IntStatus

地址: 0x2000c000

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

位	名称	权限	复位值	描述
31:8	RSVD			
7:0	IntStatus	r	0	Status of the DMA interrupts after masking

### 7.5.2 DMA\_IntTCStatus

地址: 0x2000c004

| 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

位	名称	权限	复位值	描述
31:8	RSVD			
7:0	IntTCStatus	r	0	Interrupt terminal count request status

### 7.5.3 DMA\_IntTCClear

地址: 0x2000c008

| 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

位	名称	权限	复位值	描述
31:8	RSVD			
7:0	IntTCClear	w	0	Terminal count request clear

### 7.5.4 DMA\_IntErrorStatus

地址: 0x2000c00c

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 IntErrorStatus

位	名称	权限	复位值	描述
31:8	RSVD			
7:0	IntErrorStatus	r	0	Interrupt error status

### 7.5.5 DMA\_IntErrClr

地址: 0x2000c010

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

位	名称	权限	复位值	描述
31:8	RSVD			
7:0	IntErrClr	w	0	Interrupt error clear

### 7.5.6 DMA\_RawIntTCStatus

地址: 0x2000c014

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 RawIntTCStatus

位	名称	权限	复位值	描述
31:8	RSVD			
7:0	RawIntTCStatus	r	0	Status of the terminal count interrupt prior to masking

### 7.5.7 DMA\_RawIntErrorStatus

地址: 0x2000c018

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

RawIntErrorHandler

位	名称	权限	复位值	描述
31:8	RSVD			
7:0	RawIntErrorHandler	r	0	Status of the error interrupt prior to masking

### 7.5.8 DMA\_EnbldChns

地址: 0x2000c01c

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

EnabledChannels

位	名称	权限	复位值	描述
31:8	RSVD			
7:0	EnabledChannels	r	0	Channel enable status

### 7.5.9 DMA\_SoftBReq

地址: 0x2000c020

SoftBReq

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

SoftBReq

位	名称	权限	复位值	描述
31:0	SoftBReq	r/w	0	Software burst request

### 7.5.10 DMA\_SoftSReq

地址: 0x2000c024

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

位	名称	权限	复位值	描述
31:0	SoftSReq	r/w	0	Software single request

### 7.5.11 DMA\_SoftLBReq

地址: 0x2000c028

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

位	名称	权限	复位值	描述
31:0	SoftLBReq	r/w	0	Software last burst request

### 7.5.12 DMA\_SoftLSReq

地址: 0x2000c02c

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

位	名称	权限	复位值	描述
31:0	SoftLSReq	r/w	0	Software last single request

### 7.5.13 DMA\_Config

地址: 0x2000c030

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

位	名称	权限	复位值	描述
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.

### 7.5.14 DMA\_Sync

地址: 0x2000c034

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

位	名称	权限	复位值	描述
31:0	DMA_Sync	r/w	0	DMA synchronization logic for DMA request signals: 0 = enable, 1 = disable

### 7.5.15 DMA\_C0SrcAddr

地址: 0x2000c100

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

位	名称	权限	复位值	描述
31:0	SrcAddr	r/w	0	DMA source address

### 7.5.16 DMA\_C0DstAddr

地址: 0x2000c104

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

位	名称	权限	复位值	描述
31:0	DstAddr	r/w	0	DMA Destination address

### 7.5.17 DMA\_C0LLI

地址: 0x2000c108

LLI

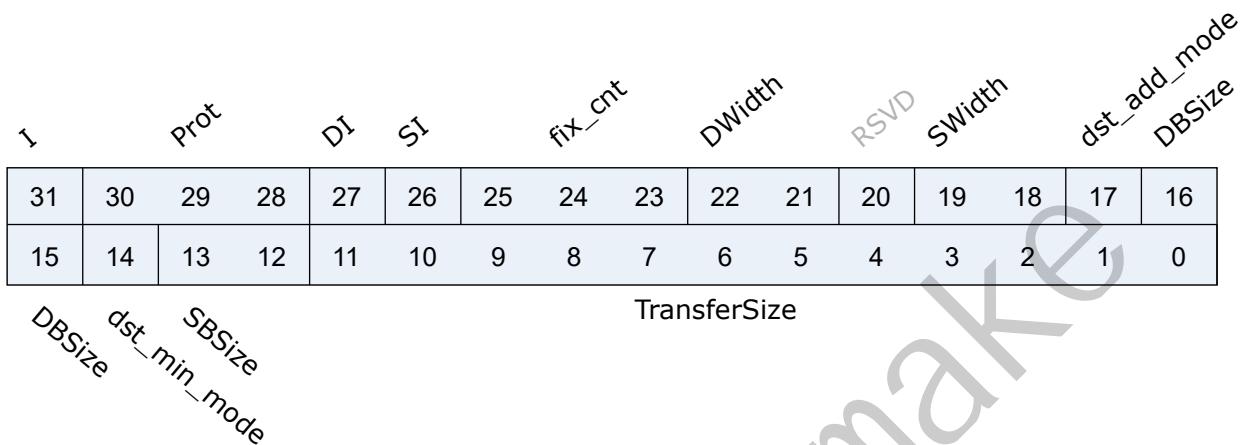
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

LLI

位	名称	权限	复位值	描述
31:0	LLI	r/w	0	First linked list item. Bits [1:0] must be 0.

### 7.5.18 DMA\_C0Control

地址: 0x2000c10c

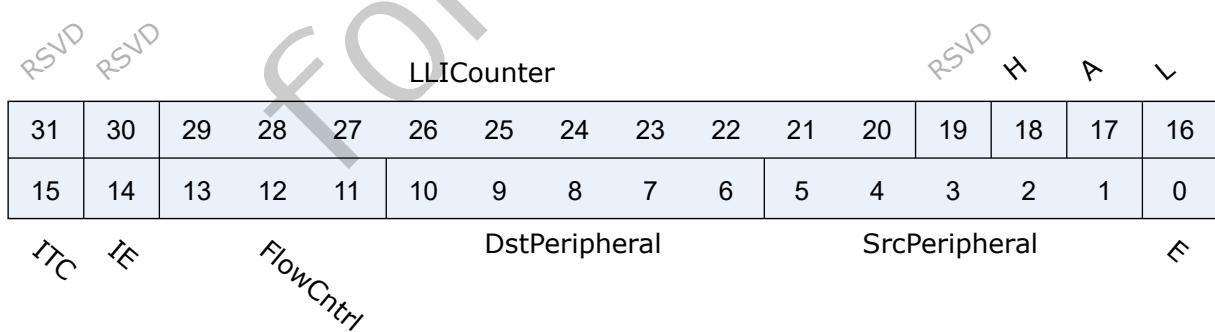


位	名称	权限	复位值	描述
31	I	r/w	0	Terminal count interrupt enable bit. It controls whether the current LLI is expected to trigger the terminal count interrupt.
30:28	Prot	r/w	0	No use for currently
27	DI	r/w	1	Destination increment. When set, the Destination address is incremented after each transfer.
26	SI	r/w	1	Source increment. When set, the source address is incremented after each transfer.
25:23	fix_cnt	r/w	3'd0	Only effect when dst_min_mode = 1 Destination transfer cnt = (total src byte cnt - (fix_cant << DWidth)) << DWidth
22:21	DWidth	r/w	2'b10	Destination transfer width: 2'b00 : byte 2'b01 : half-word 2'b10 : word 2'b11 : double-word
20	RSVD			

位	名称	权限	复位值	描述
19:18	SWidth	r/w	2'b10	Source transfer width 2'b00 : byte 2'b01 : half-word 2'b10 : word 2'b11 : double-word
17	dst_add_mode	r/w	1'b0	Add mode : issue remain destination traffic
16:15	DBSize	r/w	2'b01	Destination burst size 2'b00 : INCR1 2'b01 : INCR4 2'b10 : INCR8 2'b11 : INCR16 Note : SBSIZE*Swidth should <= CH FIFO Size
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: 2'b00 : INCR1 2'b01 : INCR4 2'b10 : INCR8 2'b11 : INCR16 Note : SBSIZE*Swidth should <= CH FIFO Size
11:0	TransferSize	r/w	0	Transfer size: 0~4095. Number of data transfers left to complete when the SMDMA is the flow controller.

### 7.5.19 DMA\_C0Config

地址: 0x2000c110



位	名称	权限	复位值	描述
31:30	RSVD			



位	名称	权限	复位值	描述
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)
10:6	DstPeripheral	r/w	0	Destination peripheral.
5:1	SrcPeripheral	r/w	0	Source peripheral.
0	E	r/w	0	Channel enable.

## 7.5.20 DMA C0RSVD

地址: 0x2000c11c

| 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    |      |      |

位	名称	权限	复位值	描述
31:5	RSVD			
4	SrcRemnSgle	r/w	0	Source remain single issue mode
3	DstRemnSgle	r/w	0	Destination remain single issue mode
2:0	RSVD			

### 7.5.21 DMA\_C1SrcAddr

地址: 0x2000c200

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

位	名称	权限	复位值	描述
31:0	SrcAddr	r/w	0	

### 7.5.22 DMA\_C1DstAddr

地址: 0x2000c204

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

位	名称	权限	复位值	描述
31:0	DstAddr	r/w	0	

### 7.5.23 DMA\_C1LLI

地址: 0x2000c208

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

位	名称	权限	复位值	描述
31:0	LLI	r/w	0	

### 7.5.24 DMA\_C1Control

地址: 0x2000c20c

1	Prot	DI	SI	fix_cnt	DWidth	RSVD	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
									DBSize

位	名称	权限	复位值	描述
31	T	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	No use for currently
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.
25:23	fix_cnt	r/w	3'd0	Only effect when dst_min_mode = 1 Destination transfer cnt = (total src byte cnt - (fix_cnt«DWidth))«DWidth

位	名称	权限	复位值	描述
22:21	DWidth	r/w	2'b10	Destination transfer width: 2'b00 : byte 2'b01 : half-word 2'b10 : word 2'b11 : double-word
20	RSVD			
19:18	SWidth	r/w	2'b10	Source transfer width 2'b00 : byte 2'b01 : half-word 2'b10 : word 2'b11 : double-word
17	dst_add_mode	r/w	1'b0	Add mode : issue remain destination traffic
16:15	DBSize	r/w	2'b01	Destination burst size 2'b00 : INCR1 2'b01 : INCR4 2'b10 : INCR8 2'b11 : INCR16 Note : SBSIZE*Swidth should <= CH FIFO Size
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: 2'b00 : INCR1 2'b01 : INCR4 2'b10 : INCR8 2'b11 : INCR16 Note : SBSIZE*Swidth should <= CH FIFO Size
11:0	TransferSize	r/w	0	Transfer size: 0~4095. Number of data transfers left to complete when the SMDMA is the flow controller.

### 7.5.25 DMA\_C1Config

地址: 0x2000c210

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

DstPeripheral      SrcPeripheral

ITC    IF      FlowCntrl      E

位	名称	权限	复位值	描述
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	

### 7.5.26 DMA\_C1RSVD

地址: 0x2000c21c

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	



位	名称	权限	复位值	描述
31:5	RSVD			
4	SrcRemnSgle	r/w	0	Source remain single issue mode
3	DstRemnSgle	r/w	0	Destination remain single issue mode
2:0	RSVD			

### 7.5.27 DMA\_C2SrcAddr

地址: 0x2000c300

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

位	名称	权限	复位值	描述
31:0	SrcAddr	r/w	0	

### 7.5.28 DMA\_C2DstAddr

地址: 0x2000c304

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

位	名称	权限	复位值	描述
31:0	DstAddr	r/w	0	

### 7.5.29 DMA\_C2LLI

地址: 0x2000c308

LLI

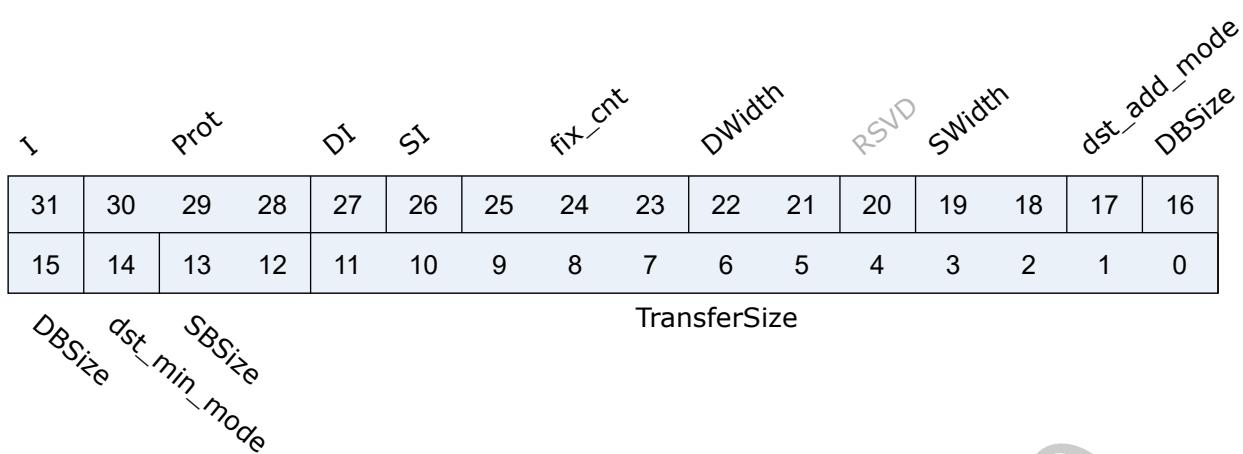
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

LLI

位	名称	权限	复位值	描述
31:0	LLI	r/w	0	

### 7.5.30 DMA\_C2Control

地址: 0x2000c30c

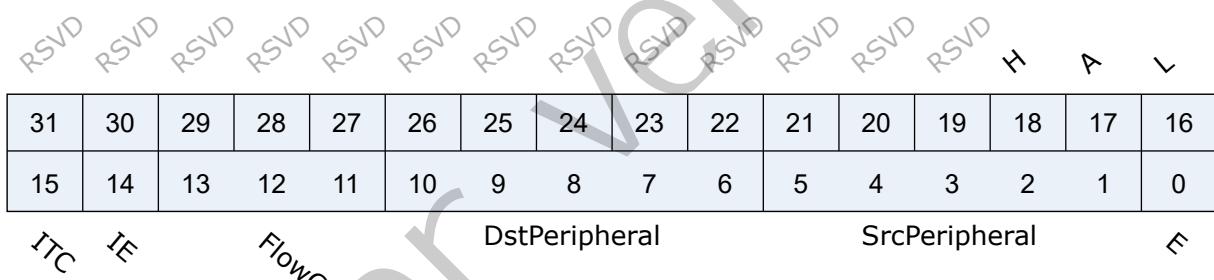


位	名称	权限	复位值	描述
31	I	r/w	0	Terminal count interrupt enable bit. It controls whether the current LLI is expected to trigger the terminal count interrupt.
30:28	Prot	r/w	0	No use for currently
27	DI	r/w	1	Destination increment. When set, the Destination address is incremented after each transfer.
26	SI	r/w	1	Source increment. When set, the source address is incremented after each transfer.
25:23	fix_cnt	r/w	3'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: 2'b00 : byte 2'b01 : half-word 2'b10 : word 2'b11 : double-word
20	RSVD			
19:18	SWidth	r/w	2'b10	Source transfer width 2'b00 : byte 2'b01 : half-word 2'b10 : word 2'b11 : double-word
17	dst_add_mode	r/w	1'b0	Add mode : issue remain destination traffic

位	名称	权限	复位值	描述
16:15	DBSize	r/w	2'b01	Destination burst size 2'b00 : INCR1 2'b01 : INCR4 2'b10 : INCR8 2'b11 : INCR16 Note : SBSIZE*Swidth should <= CH FIFO Size
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: 2'b00 : INCR1 2'b01 : INCR4 2'b10 : INCR8 2'b11 : INCR16 Note : SBSIZE*Swidth should <= CH FIFO Size
11:0	TransferSize	r/w	0	Transfer size: 0~4095. Number of data transfers left to complete when the SMDMA is the flow controller.

### 7.5.31 DMA\_C2Config

地址: 0x2000c310



位	名称	权限	复位值	描述
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	

位	名称	权限	复位值	描述
10:6	DstPeripheral	r/w	0	
5:1	SrcPeripheral	r/w	0	
0	E	r/w	0	

### 7.5.32 DMA\_C2RSVD

地址: 0x2000c31c

位	名称	权限	复位值	描述
31	RSVD			
30	RSVD			
29	RSVD			
28	RSVD			
27	RSVD			
26	RSVD			
25	RSVD			
24	RSVD			
23	RSVD			
22	RSVD			
21	RSVD			
20	RSVD			
19	RSVD			
18	RSVD			
17	RSVD			
16	RSVD			
15	RSVD			
14	RSVD			
13	RSVD			
12	RSVD			
11	RSVD			
10	RSVD			
9	RSVD			
8	RSVD			
7	RSVD			
6	RSVD			
5	RSVD			
4	SrcRemnSgle	r/w	0	Source remain single issue mode
3	DstRemnSgle	r/w	0	Destination remain single issue mode
2:0	RSVD			

### 7.5.33 DMA\_C3SrcAddr

地址: 0x2000c400

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

### SrcAddr

位	名称	权限	复位值	描述
31:0	SrcAddr	r/w	0	

### 7.5.34 DMA\_C3DstAddr

地址: 0x2000c404

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

位	名称	权限	复位值	描述
31:0	DstAddr	r/w	0	

### 7.5.35 DMA\_C3LLI

地址: 0x2000c408

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

位	名称	权限	复位值	描述
31:0	LLI	r/w	0	

### 7.5.36 DMA\_C3Control

地址: 0x2000c40c

1	Prot	DI	SI	fix_cnt	DWidth	RSVD	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
									5
								4	4
								3	3
								2	2
								1	1
								0	0

TransferSize

DBSize      dst\_min\_mode      SBSIZE

位	名称	权限	复位值	描述
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	No use for currently
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.
25:23	fix_cnt	r/w	3'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: 2'b00 : byte 2'b01 : half-word 2'b10 : word 2'b11 : double-word
20	RSVD			
19:18	SWidth	r/w	2'b10	Source transfer width 2'b00 : byte 2'b01 : half-word 2'b10 : word 2'b11 : double-word
17	dst_add_mode	r/w	1'b0	Add mode : issue remain destination traffic
16:15	DBSize	r/w	2'b01	Destination burst size 2'b00 : INCR1 2'b01 : INCR4 2'b10 : INCR8 2'b11 : INCR16 Note : SBSIZE*SWidth should <= CH FIFO Size
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: 2'b00 : INCR1 2'b01 : INCR4 2'b10 : INCR8 2'b11 : INCR16 Note : SBSIZE*SWidth should <= CH FIFO Size
11:0	TransferSize	r/w	0	Transfer size: 0 4095. Number of data transfers left to complete when the SMDMA is the flow controller.

### 7.5.37 DMA\_C3Config

地址: 0x2000c410

RSVD	H	A	V												
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

ITC IE      FlowCntrl      DstPeripheral      SrcPeripheral      E

位	名称	权限	复位值	描述
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	

### 7.5.38 DMA\_C3RSVD

地址: 0x2000c41c

| 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

DstRemnSgle SrcRemnSgle

位	名称	权限	复位值	描述
31:5	RSVD			
4	SrcRemnSgle	r/w	0	Source remain single issue mode
3	DstRemnSgle	r/w	0	Destination remain single issue mode
2:0	RSVD			

for Verimake

## 8.1 简介

红外遥控（Infrared remote，简称 IR）是一种无线、非接触式控制技术，具有抗干扰能力强、信息传输可靠、功耗低、成本低等优点。红外遥控的发射电路是采用红外发光二极管来发出经过调制的红外光波；接收电路由红外接收二极管、三极管或硅光电池组成，它们将红外发射器发射的红外光转换为相应的电信号，再送至后置放大器。

## 8.2 主要特征

- 支持以固定协议 NEC、RC-5 接收数据
- 支持以脉冲宽度计数方式接收任意格式数据
- 接收最多支持 64-bit 数据位
- 64\*2 字节的接收 FIFO
- 接收结束中断

## 8.3 功能描述

### 8.3.1 固定协议接收

IR 接收支持两种固定的协议，分别为 NEC 协议和 RC-5 协议。

- NEC 协议

NEC 协议的逻辑 1 与逻辑 0 波形如下图所示：

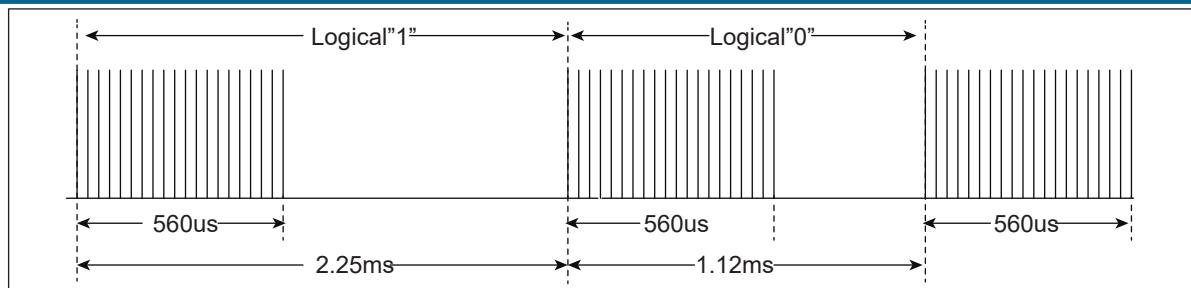


图 8.1: NEC 逻辑波形

逻辑1周期为 2.25ms，脉冲时间 560us；逻辑0周期为 1.12ms，脉冲时间 560us。NEC 协议的具体格式如下图所示：



图 8.2: NEC 协议波形

头脉冲是 9ms 的高电平脉冲和 4.5ms 的低电平，之后是 8-bit 的地址码及其反码，然后是 8-bit 的命令码及其反码，尾脉冲是 560us 高电平与 560us 低电平。

- RC-5 协议

RC-5 协议的逻辑1与逻辑0波形如下图所示：

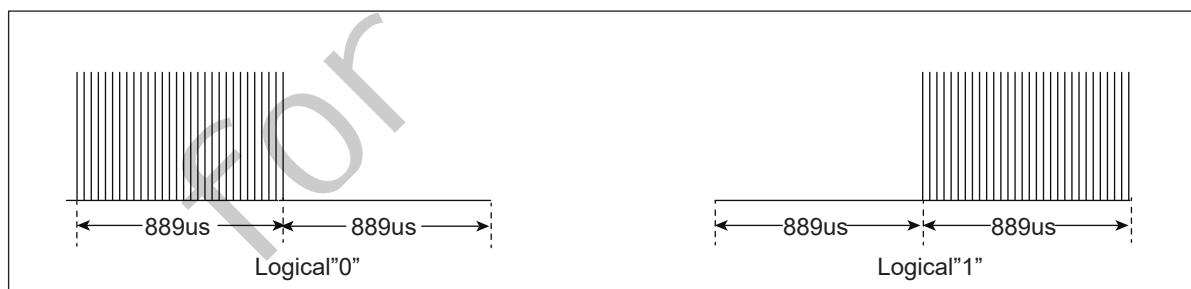


图 8.3: RC5 逻辑波形

逻辑1周期为 1.778ms，先是 889us 的低电平后是 889us 的高电平；逻辑0与逻辑1波形相反。RC-5 协议的具体格式如下图所示：

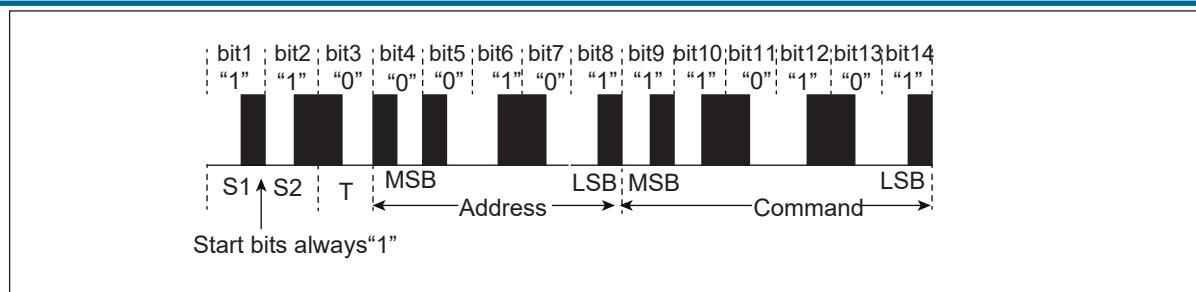


图 8.4: RC5 协议波形

前两位为开始位，固定为逻辑 1，第三位是翻转位，当一个键值发出然后再按下时该位会取反。之后 5 位是地址码与 6 位命令码。

需要注意的是，常见的红外一体接收头为了提高接收灵敏度，接收到高电平后输出的是低电平，所以在使用 IR 接收功能时要将接收翻转功能打开。

### 8.3.2 脉冲宽度接收

对于 NEC、RC-5 协议以外的其他任意格式的数据，IR 会以其内部工作时钟去依次计数每个高或者低电平的持续时间，然后将数据存入到深度为 64，宽度为 2 字节的接收 FIFO 之中。

### 8.3.3 IR 中断

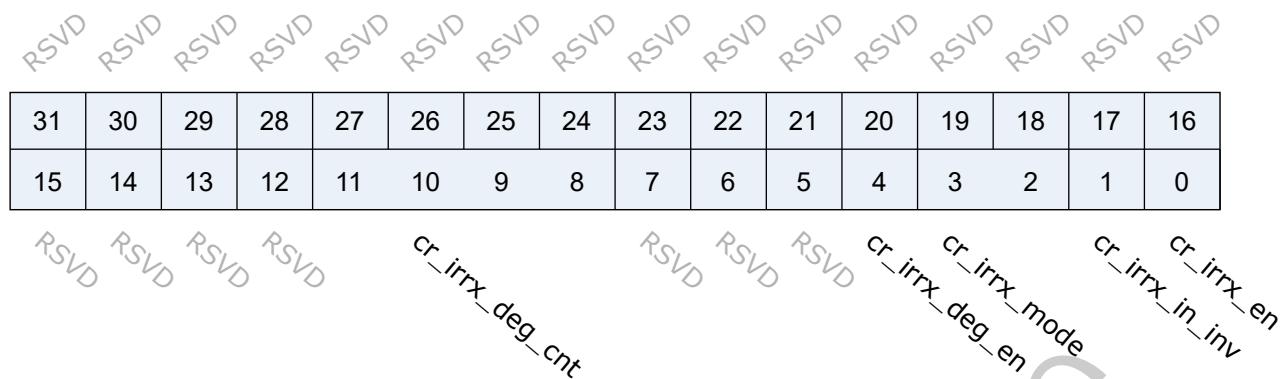
IR 有单独的接收结束中断，在接收过程中，它会用内部工作时钟去计数每段电平维持的时间，一旦这个计数达到设定的结束阈值，就会产生接收结束中断。可以通过寄存器 **IRRX\_INT\_STS** 查询接收中断状态和清除中断。

## 8.4 寄存器描述

名称	描述
irrx_config	
irrx_int_sts	
irrx_pw_config	
irrx_data_count	
irrx_data_word0	
irrx_data_word1	
ir_fifo_config_0	
ir_fifo_config_1	
ir_fifo_rdata	

### 8.4.1 irrx\_config

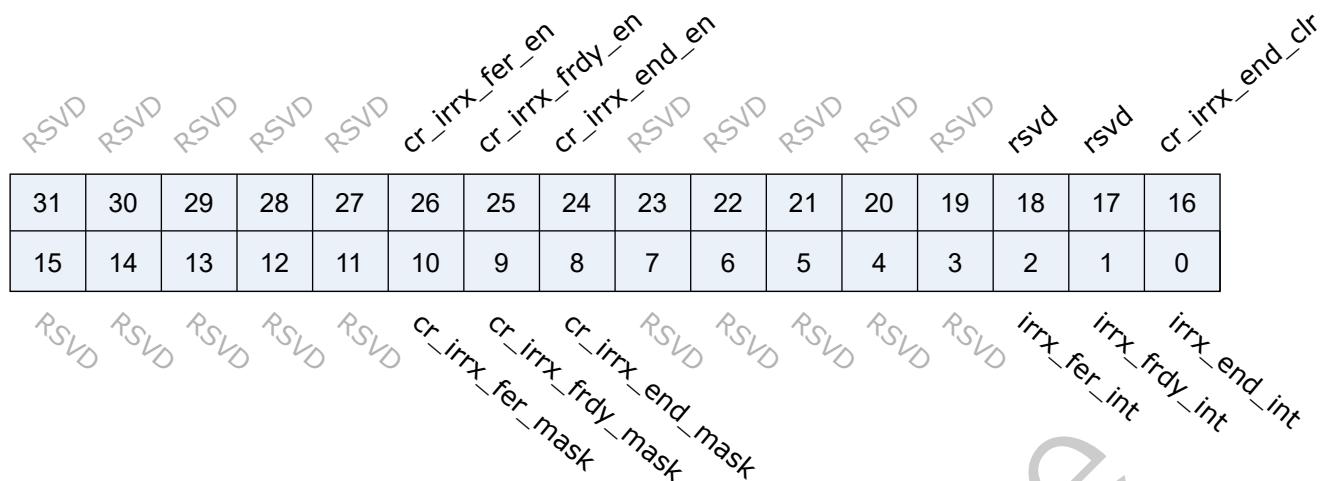
地址: 0x40010240



位	名称	权限	复位值	描述
31:12	RSVD			
11:8	cr_irrx_deg_cnt	r/w	4'd0	De-glitch function cycle count
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

### 8.4.2 irrx\_int\_sts

地址: 0x40010244



位	名称	权限	复位值	描述
31:27	RSVD			
26	cr_irrx_fer_en	r/w	1'b1	Interrupt enable of irrx_fer_int
25	cr_irrx_frdy_en	r/w	1'b1	Interrupt enable of irrx_frdy_int
24	cr_irrx_end_en	r/w	1'b1	Interrupt enable of irrx_end_int
23:19	RSVD			
18	rsvd	rsvd	1'b0	
17	rsvd	rsvd	1'b0	
16	cr_irrx_end_clr	w1c	1'b0	Interrupt clear of irrx_end_int
15:11	RSVD			
10	cr_irrx_fer_mask	r/w	1'b1	Interrupt mask of irrx_fer_int
9	cr_irrx_frdy_mask	r/w	1'b1	Interrupt mask of irrx_frdy_int
8	cr_irrx_end_mask	r/w	1'b1	Interrupt mask of irrx_end_int
7:3	RSVD			
2	irrx_fer_int	r	1'b0	IRRX FIFO error interrupt, auto-cleared when FIFO overflow/underflow error flag is cleared
1	irrx_frdy_int	r	1'b0	IRRX FIFO ready ( $rx\_fifo\_cnt > rx\_fifo\_th$ ) interrupt, auto-cleared when data is popped
0	irrx_end_int	r	1'b0	IRRX transfer end interrupt

### 8.4.3 irrx\_pw\_config

地址: 0x40010248

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

位	名称	权限	复位值	描述
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)

### 8.4.4 irrx\_data\_count

地址: 0x40010250

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

位	名称	权限	复位值	描述
31:7	RSVD			
6:0	sts_irrx_data_cnt	r	7'd0	RX data bit count (pulse-width count for SWM)

### 8.4.5 irrx\_data\_word0

地址: 0x40010254

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

位	名称	权限	复位值	描述
31:0	sts_irrx_data_word0	r	32'h0	RX data word 0

#### 8.4.6 irrx\_data\_word1

地址: 0x40010258

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

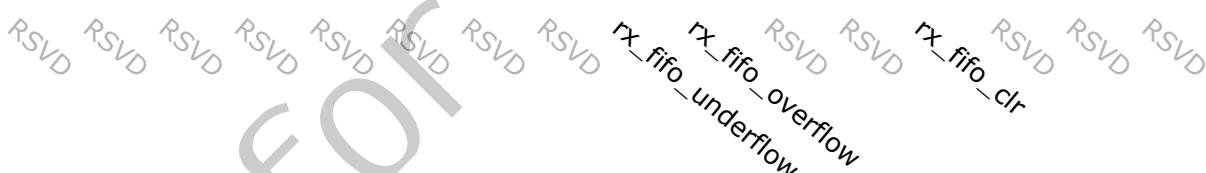
sts\_irrx\_data\_word1

位	名称	权限	复位值	描述
31:0	sts_irrx_data_word1	r	32'h0	RX data word 1

#### 8.4.7 ir\_fifo\_config\_0

Address: 0x40010280

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	



位	名称	权限	复位值	描述
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
5:4	RSVD			
3	rx_fifo_clr	w1c	1'b0	Clear signal of RX FIFO
2:0	RSVD			

### 8.4.8 ir\_fifo\_config\_1

Address: 0x40010284

rx_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
rx_fifo_cnt															
RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD

位	名称	权限	复位值	描述
31:30	RSVD			
29:24	rx_fifo_th	r/w	6'd0	RX FIFO threshold, irrx_frdy_int will not be asserted if rx_fifo_cnt is less than this value
23:15	RSVD			
14:8	rx_fifo_cnt	r	7'd0	RX FIFO available count
7:0	RSVD			

### 8.4.9 ir\_fifo\_rdata

地址: 0x4001028c

rx_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
RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD

位	名称	权限	复位值	描述
31:16	RSVD			
15:0	rx_fifo_rdata	r	16'h0	IRRX FIFO pulse width data for Software Mode

## 9.1 简介

串行外设接口（Serial Peripheral Interface Bus, SPI）是一种用于短程通信的同步串行通信接口规范，装置之间使用全双工模式通信，是一个主机和一个或多个从机的主从模式。SPI 使用 4 根线完成全双工的通信，这 4 根信号线分别是：CS（片选）、SCLK（时钟）、MOSI(主机输出从机输入)、MISO(主机输入从机输出)。

## 9.2 主要特征

- 既可作为 SPI 主设备，也可作为 SPI 从设备
- 主从设备都支持 4 种时钟格式（CPOL, CPHA）
- 主从设备都支持 1/2/3/4 字节传输模式
- 发送和接收通道各有深度为 32 个字节的 FIFO
  - 当 Frame 为 32Bits 时，FIFO 的深度为 8
  - 当 Frame 为 24Bits 时，FIFO 的深度为 8
  - 当 Frame 为 16Bits 时，FIFO 的深度为 16
  - 当 Frame 为 8Bits 时，FIFO 的深度为 32
- 可调整字节传输顺序
- 灵活的时钟配置，最高可支持 80M 时钟
- 可配置 MSB/LSB 优先传输
- 接收忽略功能，可以设定忽略对指定位置数据的接收
- 支持从设备模式下的超时机制

- 支持 DMA 传输模式

## 9.3 功能描述

### 9.3.1 时钟控制

依照不同的时钟相位以及极性设定，SPI 时钟共有四种模式，可以通过寄存器 `spi_config` 的 `cr_spi_sclk_pol` (CPOL) 和 `cr_spi_sclk_ph` (CPHA) 进行设置。CPOL 用来决定 SCK 时钟信号空闲时的电平，`CPOL=0` 则空闲电平为低电平，`CPOL=1` 则空闲电平为高电平。CPHA 用来决定采样时刻，`CPHA=0` 则在每个周期的第一个时钟沿采样，`CPHA=1` 则在每个周期的第二个时钟沿采样。通过设置寄存器 `spi_prd_0` 和 `spi_prd_1`，还可以调整时钟的开始和结束电平持续时间、相位 0/1 的时间以及每帧数据之间的间隔。四种模式下的具体设置如下图所示：

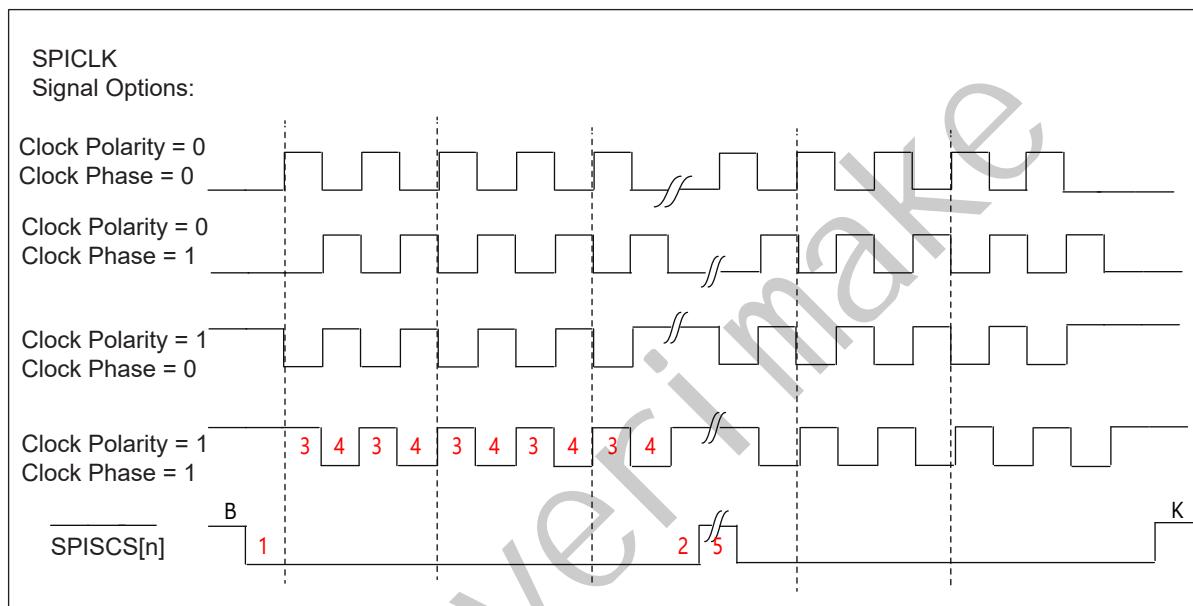


图 9.1: SPI 时序图

其中各数字含义如下：

- 1 是起始条件的长度，通过配置寄存器 `spi_prd_0` 中的 `cr_spi_prd_s`。
- 2 是停止条件的长度，通过配置寄存器 `spi_prd_0` 中的 `cr_spi_prd_p`。
- 3 是相位 0 的长度，通过配置寄存器 `spi_prd_0` 中的 `cr_spi_prd_d_ph_0`。
- 4 是相位 1 的长度，通过配置寄存器 `spi_prd_0` 中的 `cr_spi_prd_d_ph_1`。
- 5 是每帧数据之间的间隔，通过配置寄存器 `spi_prd_1` 中的 `cr_spi_prd_i`。

### 9.3.2 主设备持续传输模式

开启该模式后，在发送完当前数据而 FIFO 里还存在可用数据时，CS 信号不会被释放。

### 9.3.3 主从设备传输接收数据

主从设备传输接收数据时应设置相同的 framesize，可通过设置寄存器 spi\_config 中的 cr\_spi\_frame\_size。如果主设备和从设备的约定以 32bits 的 framesize 进行通信时，在某一帧数据中，主设备的 clk 由于异常不满足 32 时，会出现下列现象。

- 主设备发送的数据无法被送到从设备的 RX FIFO 中。从设备无法接收到数据。
- 从设备发送的数据时，会跳过这一帧数据，等下次主设备 clk 正常时，继续发送下一帧数据。

### 9.3.4 接收忽略功能

通过设置需要过滤掉的开始位和结束位，SPI 会将接收到数据中的对应数据段丢弃。如下图所示：

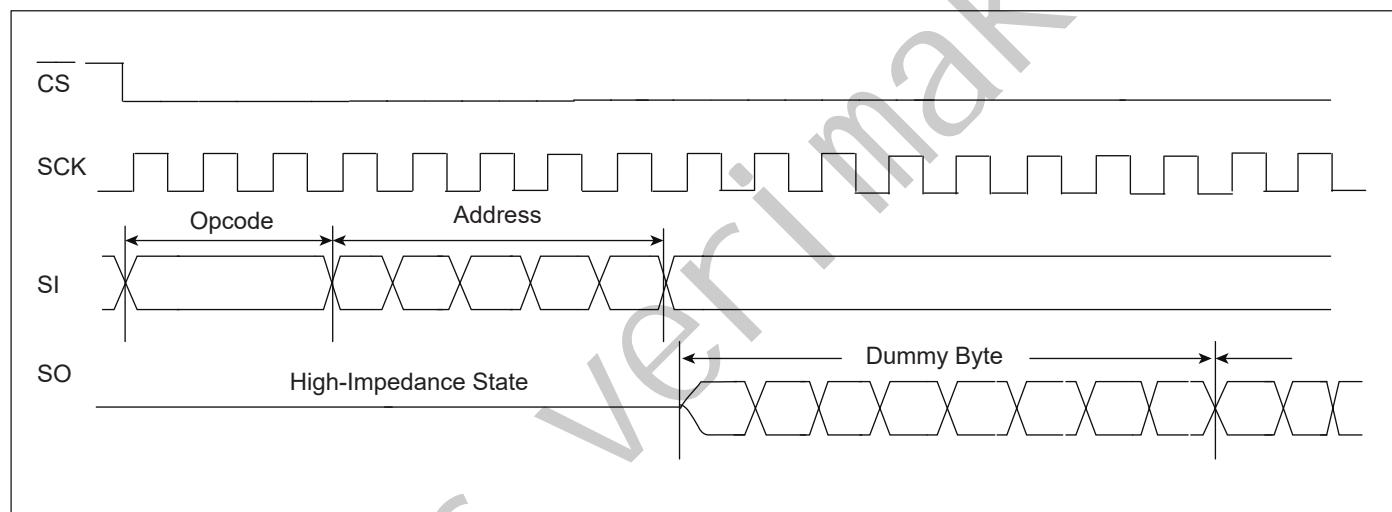


图 9.2: SPI Ignore 波形图

通过配置寄存器 spi\_config 中的 cr\_spi\_rxd\_ignr\_en 开启忽略功能。通过配置寄存器 spi\_rxd\_ignr 中的 cr\_spi\_rxd\_ignr\_s 设置忽略功能的起始位。通过配置寄存器 spi\_rxd\_ignr 中的 cr\_spi\_rxd\_ignr\_p 设置忽略功能的结束位。

上图中过滤的开始位设为 0，结束位设为 7 则 Dummy Byte 会被收到，结束位设为 15 则 Dummy Byte 会被丢弃。

### 9.3.5 滤波功能

通过使能该功能和设置门限值，SPI 会将小于或等于门限值宽度的数据过滤。通过配置寄存器 `spi_config` 中的 `cr_spi_deg_en` 使能该功能和配置 `cr_spi_deg_cnt` 设置门限值，SPI 会将达不到门限值宽度的数据过滤掉。数据宽度小于 `cr_spi_deg_cnt+1`；如下图所示，若想滤去数据宽度小于 4 的数据，需要将 `cr_urx_deg_cnt` 的值设置为 4。`input` 为初始数据，`output` 为滤波后的数据。

滤波逻辑过程：

- `tgl` 为 `input` 和 `output` 的异或结果。
- `deg_cnt` 从 0 开始计数，计数条件为 `tgl` 为高电平，并且 `reached` 为低电平。
- 若 `deg_cnt` 计数值达到 `cr_urx_deg_cnt` 设置的值时，`reached` 为高电平。
- 当 `reached` 为高电平时，将 `input` 输出到 `output`。
- 注释: `deg_cnt` 自加的条件：`tgl` 为高电平且 `reached` 为低电平，其余情况下 `deg_cnt` 会被清 0。

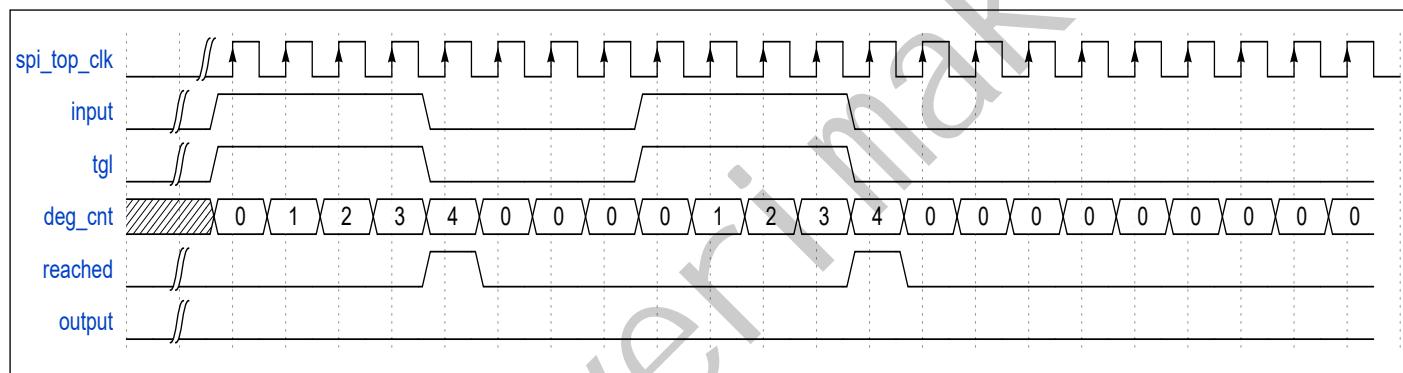


图 9.3: SPI 滤波波形图

### 9.3.6 可配置 MSB/LSB 传输

可配置 MSB/LSB 传输方式仅限于 1 个 byte 中的 8 个 bits 的优先传输顺序，通过配置寄存器 `spi_config` 中的 `cr_spi_bit_inv` 位设置字节内 bit 的传输顺序。0 表示 MSB，1 表示 LSB。以 frame size 等于 24 bits 传输数据为例，数据格式为：`Data[23:0]=0x123456`。当设置为 MSB 传输时，传输的顺序为：01010110(二进制，第 1 个字节：0x56);00110100(二进制，第 2 个字节：0x34);00010010(二进制，第 3 个字节：0x12)；当设置为 LSB 传输时，传输的顺序为：01101010(二进制，第 1 个字节：0x56);00101100(二进制，第 2 个字节：0x34);01001000(二进制，第 3 个字节：0x12)。

### 9.3.7 可调整字节传输顺序

可调整字节传输顺序方式仅限于 FIFO 中不同的 byte 间的优先传输顺序。通过配置寄存器 `spi_config` 中的 `cr_spi-byte_inv` 位设置 FIFO 内 byte 的传输顺序。0 表示优先发送低字节，1 表示优先发送高字节。同样以 frame size 等于 24 bits 传输数据为例，数据格式为：Data[23:0]=0x123456。当设置优先发送低字节，传输的顺序为：0x56(第 1 个字节：低字节); 0x34(第 2 个字节：中间字节); 0x12(第 3 个字节：高字节); 当设置优先发送高字节，传输的顺序为：0x12(第 3 个字节：高字节); 0x34(第 2 个字节：中间字节); 0x56(第 1 个字节：低字节);

字节传输顺序调整功能可以和 MSB/LSB 传输配置功能配合使用。

### 9.3.8 从模式超时机制

通过设定一个超时门限，当从模式下 SPI 超过该时间值未收到时钟信号时，会触发中断。

### 9.3.9 I/O 传输模式

芯片通信处理器可以响应来自 FIFO 的中断来执行 FIFO 填充和清空操作。每个 FIFO 都有一个可编程的 FIFO 触发阈值来触发中断。当寄存器 `spi_fifo_config_1` 中的 `rx_fifo_cnt` 大于 `rx_fifo_th` 触发阈值时，将产生 RX 请求中断，通知芯片通信处理器可以读取 RX FIFO。当寄存器 `spi_fifo_config_1` 中的 `tx_fifo_cnt` 大于 `tx_fifo_th` 时，将产生 TX 请求中断，通知芯片通信处理器重新填充 TX FIFO。可以通过查询 FIFO 状态寄存器来确定 FIFO 中的采样值以及 FIFO 的状态。软件负责确保正确的 RX FIFO 触发阈值和 TX FIFO 触发阈值，防止接收 FIFO overflow 和发送 FIFO underflow。

### 9.3.10 DMA 传输模式

SPI 支持 DMA 传输模式。使用该模式需要分别设置 TX 和 RX FIFO 的阈值，将寄存器 `spi_fifo_config_0` 中的 `spi-dma_tx_en` 置 1，则开启 DMA 发送模式。将寄存器 `spi_fifo_config_0` 中的 `spi_dma_rx_en` 置 1，则开启 DMA 接收模式。当该模式启用后，UART 会对 TX/RX FIFO 进行检查，一旦寄存器 `spi_fifo_config_1` 中的 `tx_fifo_cnt/rx_fifo_cnt` 大于 `tx_fifo_th/rx_fifo_th`，将会发起 DMA 请求，DMA 会按照设定将数据搬移至 TX FIFO 中或从 RX FIFO 中移出。

### 9.3.11 SPI 中断

SPI 有着丰富的中断控制，包括以下几种中断模式：

- SPI 传输结束中断
- TX FIFO 请求中断
- RX FIFO 请求中断
- 从模式传输超时中断
- 从模式 TX 过载中断
- TX/RX FIFO 溢出中断

在主模式下，SPI 传输结束中断会在每帧数据传输结束时触发；在从模式下，SPI 传输结束中断会在 CS 信号被释放时触发。TX/RX FIFO 请求中断会在其 FIFO 可用计数值大于其设定的阈值时触发，当条件不满足时该中断标志会自动清

除。从模式传输超时中断会在从模式下超过超时门限值未收到时钟信号时触发。从模式 TX 过载中断会在从模式下 TX 没有准备好传输时出发。如果 TX/RX FIFO 发生了上溢或者下溢，会触发 TX/RX FIFO 溢出中断，当 FIFO 清除寄存器 spi\_fifo\_config\_0 中的 tx\_fifo\_clr/rx\_fifo\_clr 被置 1 时，对应的 FIFO 会被清空，同时溢出中断标志会自动清除。可以通过寄存器 SPI\_INT\_STS 查询各中断状态和对相应的位写 1 清除中断。

## 9.4 寄存器描述

名称	描述
spi_config	
spi_int_sts	
spi_bus_busy	
spi_prd_0	
spi_prd_1	
spi_rxd_ignr	
spi_sto_value	
spi_fifo_config_0	
spi_fifo_config_1	
spi_fifo_wdata	
spi_fifo_rdata	
backup_io_en	

### 9.4.1 spi\_config

地址: 0x40019000

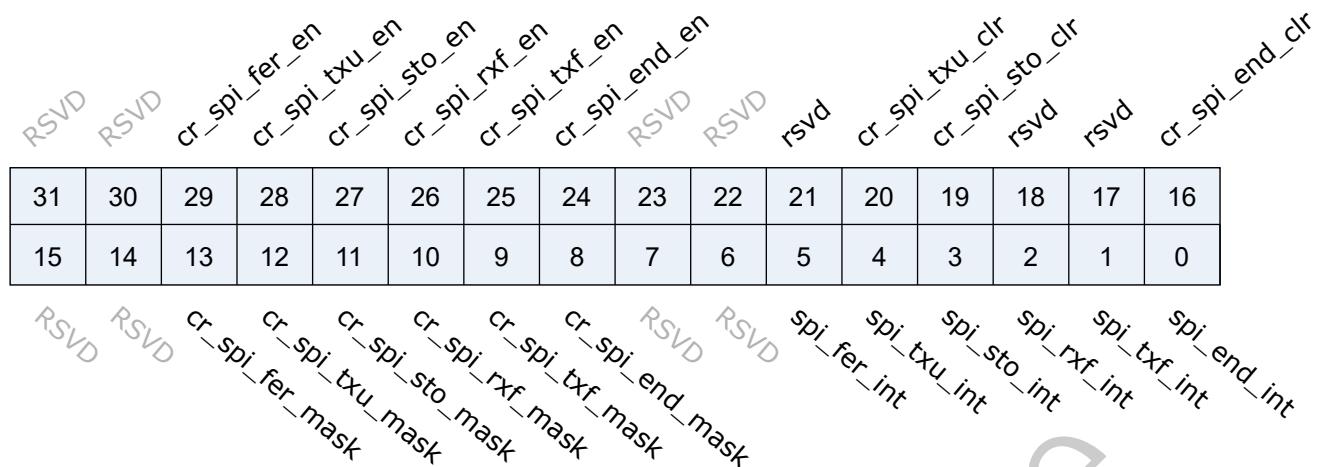
| RSVD |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 31   | 30   | 29   | 28   | 27   | 26   | 25   | 24   | 23   | 22   | 21   | 20   | 19   | 18   | 17   | 16   |      |      |      |
| 15   | 14   | 13   |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

*cr\_spi\_deg\_cnt*      *cr\_spi\_deg\_en*      *cr\_spi\_s\_3pin\_mode*      *cr\_spi\_m\_cont\_en*      *cr\_spi\_rxm\_ignr\_en*      *cr\_spi\_bit\_inv*      *cr\_spi\_sclk\_ph*      *cr\_spi\_sclk\_pol*      *cr\_spi\_frame\_size*      *cr\_spi\_s\_en*      *cr\_spi\_m\_en*

位	名称	权限	复位值	描述
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	cr_spi_s_3pin_mode	r/w	1'b0	SPI slave 3-pin mode 1'b0: 4-pin mode (SS_n is enabled) 1'b1: 3-pin mode (SS_n is disabled / don't care)
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
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

地址: 0x40019004



位	名称	权限	复位值	描述
31:30	RSVD			
29	cr_spi_fer_en	r/w	1'b1	Interrupt enable of spi_fer_int
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
23:22	RSVD			
21	rsvd	rsvd	1'b0	
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	rsvd	rsvd	1'b0	
17	rsvd	rsvd	1'b0	
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

位	名称	权限	复位值	描述
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'b1	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

地址: 0x40019008

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

位	名称	权限	复位值	描述
31:1	RSVD			
0	sts_spi_bus_busy	r	1'b0	Indicator of SPI bus busy

#### 9.4.4 spi\_prd\_0

地址: 0x40019010

cr\_spi\_prd\_d\_ph\_1

cr\_spi\_prd\_d\_ph\_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\_spi\_prd\_p

cr\_spi\_prd\_s

位	名称	权限	复位值	描述
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

地址: 0x40019014

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

位	名称	权限	复位值	描述
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

地址: 0x40019018

RSVD	cr_spi_rxd_ignr_s															
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 cr\_spi\_rxd\_ignr\_p

位	名称	权限	复位值	描述
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

地址: 0x4001901c

RSVD	cr_spi_sto_value															
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	

位	名称	权限	复位值	描述
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

地址: 0x40019080

| 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

rx\_fifo\_underrflow      rx\_fifo\_overflow      tx\_fifo\_underrflow      tx\_fifo\_overflow      tx\_fifo\_clr      spi\_dma\_rx\_en      spi\_dma\_tx\_en

位	名称	权限	复位值	描述
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
0	spi_dma_tx_en	r/w	1'b0	Enable signal of dma_tx_req/ack interface

## 9.4.9 spi\_fifo\_config\_1

地址: 0x40019084

Memory Map Register (0x00000000)															
RX FIFO								TX FIFO							
RX FIFO Control				RX FIFO Status				TX FIFO Control				TX FIFO Status			
RSVD				RSVD				RSVD				RSVD			
rx_fifo_th				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
RSVD				rx_fifo_cnt				RSVD				tx_fifo_cnt			

位	名称	权限	复位值	描述
31:29	RSVD			
28:24	rx_fifo_th	r/w	5'd0	RX FIFO threshold, dma_rx_req will not be asserted if rx_fifo_cnt is less than this value
23:21	RSVD			
20:16	tx_fifo_th	r/w	5'd0	TX FIFO threshold, dma_tx_req will not be asserted if tx_fifo_cnt is less than this value
15:14	RSVD			
13:8	rx_fifo_cnt	r	6'd0	RX FIFO available count (unit: byte)
7:6	RSVD			
5:0	tx_fifo_cnt	r	6'd32	TX FIFO available count (unit: byte)

#### 9.4.10 spi\_fifo\_wdata

地址: 0x40019088

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

位	名称	权限	复位值	描述
31:0	spi_fifo_wdata	w	x	<p>TX FIFO write data port</p> <p>Note: Partial valid if cr_spi_frame_size is set to different value:</p> <ul style="list-style-type: none"> <li>2'd0 (8-bit frame): Only [7:0] are valid and [31:8] are don't-care</li> <li>2'd1 (16-bit frame): Only [15:0] are valid and [31:16] are don't-care</li> <li>2'd2 (24-bit frame): Only [23:0] are valid and [31:24] are don't-care</li> <li>2'd3 (32-bit frame): Entire [31:0] are valid</li> </ul>



### 9.4.11 spi fifo rdata

地址: 0x4001908c

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

位	名称	权限	复位值	描述
31:0	spi_fifo_rdata	r	32'h0	<p>RX FIFO read data port</p> <p>Note: Partial valid if cr_spi_frame_size is set to different value:</p> <ul style="list-style-type: none"> <li>2'd0 (8-bit frame): Only [7:0] are valid and [31:8] are all 0s</li> <li>2'd1 (16-bit frame): Only [15:0] are valid and [31:16] are all 0s</li> <li>2'd2 (24-bit frame): Only [23:0] are valid and [31:24] are all 0s</li> <li>2'd3 (32-bit frame): Entire [31:0] is valid</li> </ul>

## 9.4.12 backup\_io\_en

地址: 0x400190fc

| 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    |      |      |

位	名称	权限	复位值	描述
31:1	RSVD			
0	backup_io_en	r/w	1'b0	

# 10

## UART

### 10.1 简介

通用异步收发传输器（Universal Asynchronous Receiver/Transmitter，通常称为 UART）是一种异步收发传输器，提供了与外部设备进行全双工数据交换的灵活方式。BL616/BL618 共有 2 组 UART，配合 DMA 使用，可以实现高效的数据通信。

### 10.2 主要特征

- 全双工异步通信
- 数据位长度可选择 5/6/7/8 比特
- 停止位长度可选择 0.5/1/1.5/2 比特
- 支持奇/偶/无校验比特
- 可侦测错误的起始比特
- 丰富的中断控制
- 支持硬件流控（RTS/CTS）
- 便捷的波特率编程
- 可配置 MSB/LSB 优先传输
- 普通/固定字符的自动波特率检测
- 32 字节发送/接收 FIFO
- 支持 DMA 传输模式
- 支持 10Mbps 及其以下波特率
- 支持 LIN 总线协议

- 支持 RS485 模式
- 时钟源可以选择 160M/BCLK/XCLK
- 支持滤波功能

## 10.3 功能描述

### 10.3.1 数据格式描述

正常的 UART 通信数据是由起始位、数据位、奇偶校验位、停止位组成的。BL616/BL618 的 UART 支持可配置的数据位、奇偶校验位和停止位，这些都在寄存器 `utx_config` 和 `urx_config` 中设置。一帧数据的波形如下图所示：

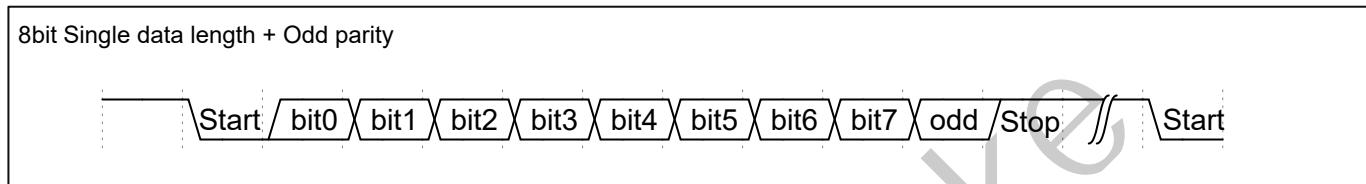


图 10.1: UART 数据格式

数据帧的起始位占用 1-bit，停止位可以通过配置寄存器 `utx_config` 中的 `cr_utx_bit_cnt_p` 实现 0.5/1/1.5/2 位宽。起始位为低电平，停止位为高电平。数据位宽可以通过寄存器 `utx_config` 中的 `cr_utx_bit_cnt_d` 配置为 5/6/7/8 位宽。当置位寄存器 `utx_config` 中的 `cr_utx_prt_en` 和寄存器 `urx_config` 中的 `cr_urx_prt_en` 时，数据帧会在数据之后添加一位奇偶校验位。寄存器 `utx_config` 中的 `cr_utx_prt_sel` 和寄存器 `urx_config` 中的 `cr_urx_prt_sel` 用于选择奇校验还是偶校验。当接收器检测到输入数据的校验位错误时会产生校验错误中断。但是接收的数据仍然会进入 FIFO。奇校验的计算方法：如果当前数据位 1 的个数是奇数个，奇校验位为 0；反之为 1。偶校验的计算方法：如果当前数据位 1 的个数是奇数个，偶校验位为 1；反之为 0。

### 10.3.2 基本架构

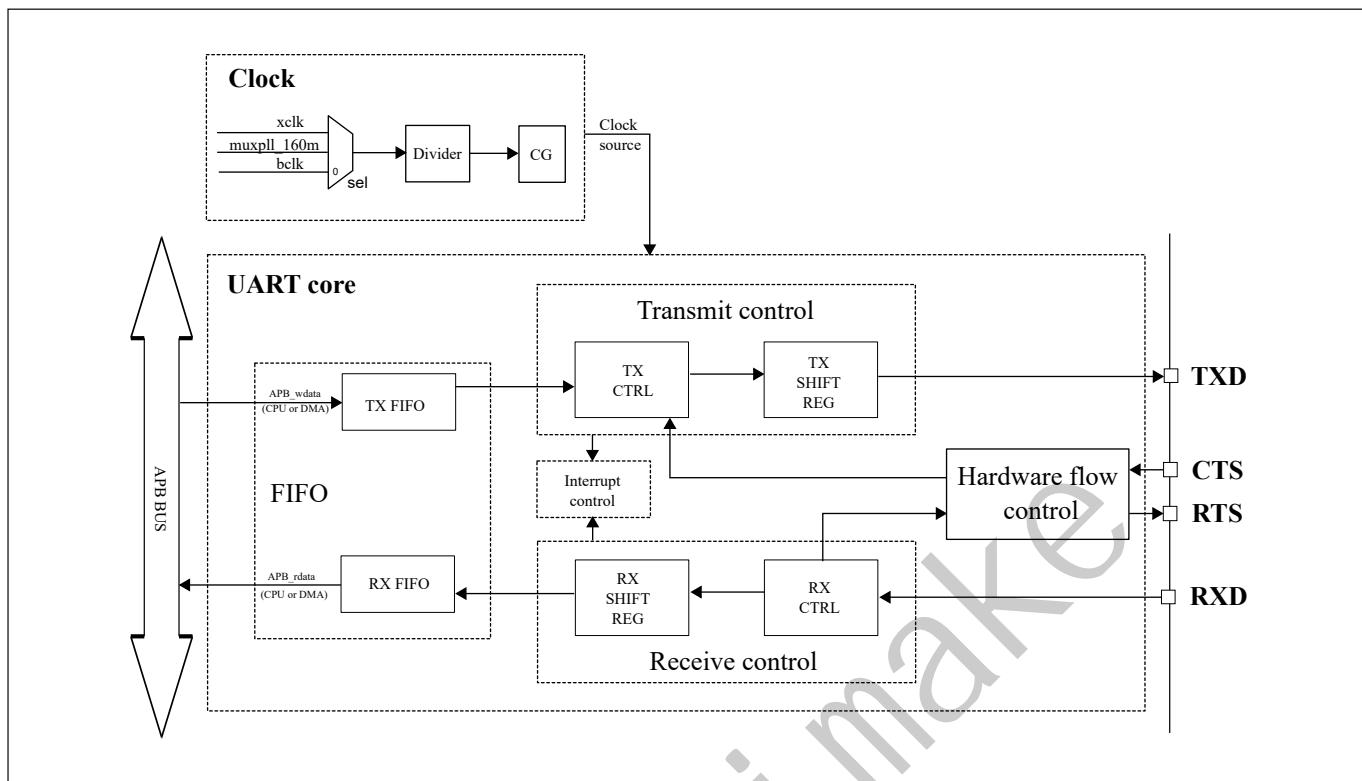


图 10.2: UART 基本架构图

UART 有 3 个时钟源: XCLK, 160MHz CLK 以及 BCLK。时钟中的分频器用于对时钟源进行分频, 然后产生时钟信号来驱动 UART 模块。

UART 控制器分为两个功能模块: 发送器和接收器。

### 10.3.3 发送器

发送器包含一个 32 字节的发送 FIFO, 用来存放待发送的数据。软件可以通过 APB 总线写 TX FIFO, 也可以通过 DMA 将数据搬入 TX FIFO。当发送使能位被设置时, FIFO 中存放的数据会从 TX 引脚输出。软件可以通过寄存器 `uart_fifo_config_1` 中的 `tx_fifo_cnt` 查询 TX FIFO 剩余可用空间计数值来检查发送器的状态。

发送器的自由运行 (FreeRun) 模式如下:

- 如果没有开启自由运行 (FreeRun) 模式, 则当发送字节达到指定长度时发送行为终止并产生中断, 如果要继续发送则需重新关闭再使能发送使能位。
- 如果开启自由运行 (FreeRun) 模式, 则当 TX FIFO 里存在数据时, 发送器就会进行发送, 不会因为发送字节达到指定长度而终止。

### 10.3.4 接收器

接收器包含一个 32 字节的接收 FIFO，用来存放接收到的数据。软件可以通过寄存器 `uart_fifo_config_1` 中的 `rx_fifo_cnt` 查询 RX FIFO 可用数据计数值来检查接收器的状态。寄存器 `URX_RTO_TIMER` 的低 8 位用于设定一个接收超时门限，当接收器超过该时间值未收到数据时，会触发中断。具体中断触发条件请参考 RX 超时中断小节。寄存器 `urx_config` 中的 `cr_urx_deg_en` 和 `cr_urx_deg_cnt` 用于使能去毛刺功能和设置门限值，其控制的是 UART 采样之前的滤波部分，UART 会将波形当中宽度低于门限值的毛刺滤掉，然后在将其送去采样。

### 10.3.5 波特率设定

$$\text{Baudrate} = \frac{\text{UART\_clk}}{\text{uart\_prd} + 1}$$

用户可单独设置 RX 与 TX 的波特率，以 TX 为例：`uart_prd` 的数值为寄存器 `UART_BIT_PRD` 的低 16 位 `cr_utx_bit_prd` 数值，由于 16 位位宽系数最大值为 65535，所以 UART 支持的最小波特率为：`UART_clk/65536`。

在 UART 对数据进行采样之前，会先对数据进行滤波，将波形当中的毛刺滤掉。然后会在上述 16 位系数的中间值时刻进行采样，这样根据不同的波特率调整不同的采样时刻，以保持其采到的始终是中间值，大大提高了灵活性与精度。采样过程如下图所示：

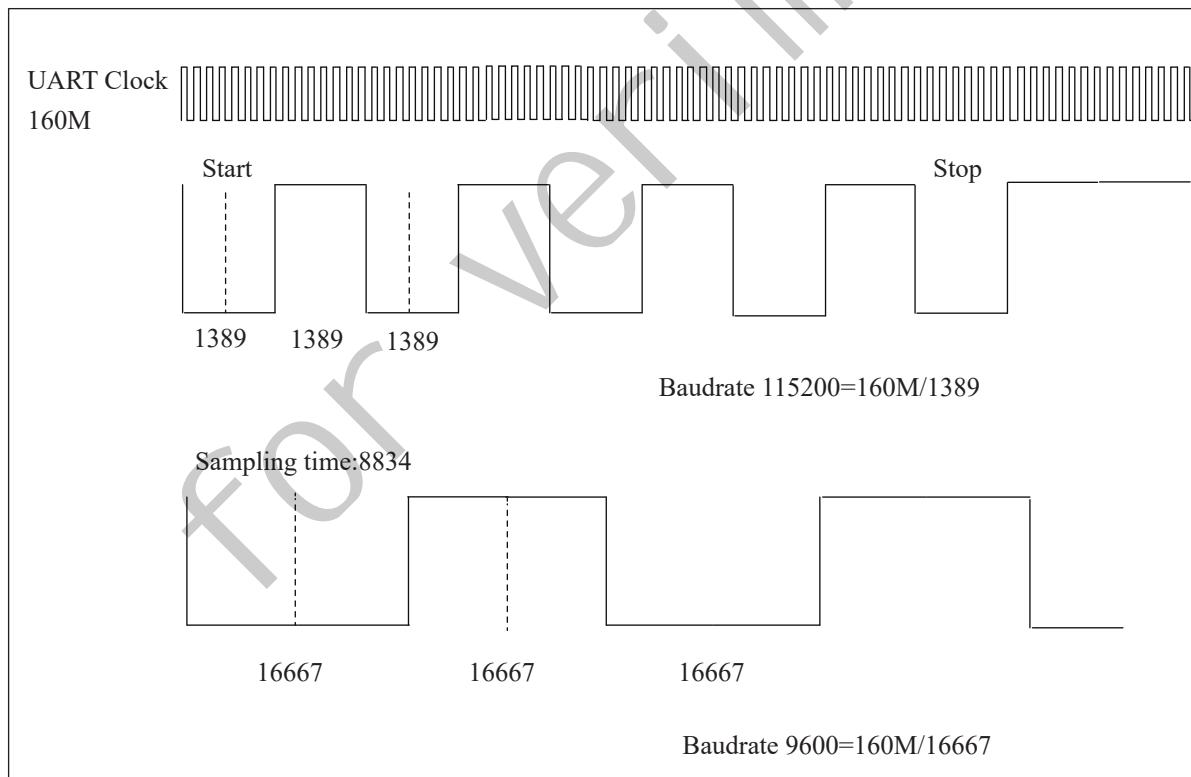


图 10.3: UART 采样波形图

### 10.3.6 滤波

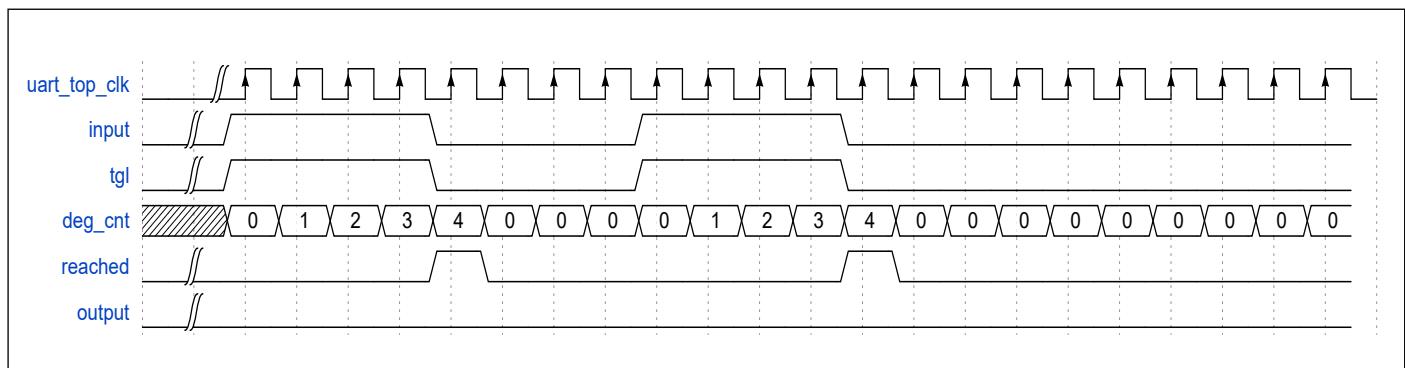


图 10.4: UART 滤波波形图

通过配置寄存器 `urx_config` 中的 `cr_urx_deg_en` 使能该功能，配置 `cr_urx_deg_cnt` 设置门限值，UART 会将达不到门限值宽度的数据过滤掉。

如上图所示，若想滤去数据宽度小于 4 的数据，需要将 `cr_urx_deg_cnt` 的值设置为 4。`input` 为初始数据，`output` 为滤波后的数据。

滤波逻辑过程：

- `tgl` 为 `input` 和 `output` 的异或结果。
- `deg_cnt` 从 0 开始计数，计数条件为 `tgl` 为高电平，并且 `reached` 为低电平。
- 若 `deg_cnt` 计数值达到 `cr_urx_deg_cnt` 设置的值时，`reached` 为高电平。
- 当 `reached` 为高电平时，将 `input` 输出到 `output`。
- 注释: `deg_cnt` 自加的条件：`tgl` 为高电平且 `reached` 为低电平，其余情况下 `deg_cnt` 会被清 0。

### 10.3.7 自动波特率检测

UART 模块支持自动波特率检测，该检测分为两种，一种是通用模式，一种是固定字符（方波）模式。由寄存器 `urx_config` 中的 `cr_urx_abr_en` 控制，每次开启时，这两种检测模式都会启用。

通用模式

对于接收到的任意字符数据，UART 模块会以 UART 的时钟周期为单位去计数起使位的位宽，该计数值被写入寄存器 `STS_URX_ABR_PRD` 的低 16 位 `sts_urx_abr_prd_start` 中用以计算波特率，因此当起始位之后的第一个数据位为 1 时即可得到正确的波特率，如 `LSB-FIRST` 下的'0x01'。

固定字符（方波）模式

该模式下，UART 模块在用时钟周期去计数起使位的位宽后，会继续计数后续数据位的位宽，并与起始位相比较，如果连续 7 个计数值之间的上下浮动在最大允许误差范围内，则通过检测，否则计数值会被丢弃。最大允许误差可以通过寄存器 `urx_abr_pw_tol` 中的 `cr_urx_abr_pw_tol` 位来设置，以 UART 的时钟周期为单位。因此，只有在 `LSB-FIRST`

下接收到固定字符'0x55'/'0xD5' 或 MSB-FIRST 下的'0xAA'/'0xAB' 时，UART 模块才会将满足最大允许误差范围的连续 7 个计数值写入寄存器 STS\_URX\_ABR\_PRD 的高 16 位 sts\_urx\_abr\_prd\_0x55 中。

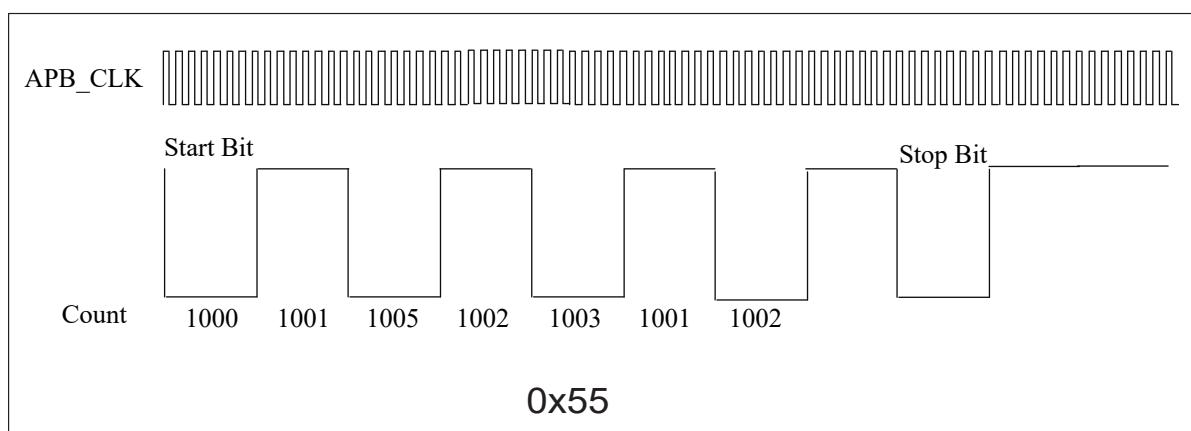


图 10.5: UART 固定字符模式波形图

如上图所示，假设设置的最大允许误差为 4，则对于接收到的某一波特率未知的数据，UART 用 UART\_CLK 去计数起始位的位宽为 1000，第二位的位宽为 1001，与前一位宽上下浮动不超过 4 个 UART\_CLK，则 UART 会继续计数第三位，第三位为 1005，与起始位相差超过 4，则检测不通过，数据丢弃。UART 会依次将数据位的前 6 位位宽与起始位进行比较。

计算检测到的波特率的公式如下：

$$\text{Baudrate} = \frac{\text{UART\_clk}}{\text{Count} + 1}$$

### 10.3.8 硬件流控

UART 支持 CTS/RTS 方式的硬件流控，以防止 FIFO 里的数据由于来不及处理而丢失。硬件流控连接如下图所示：

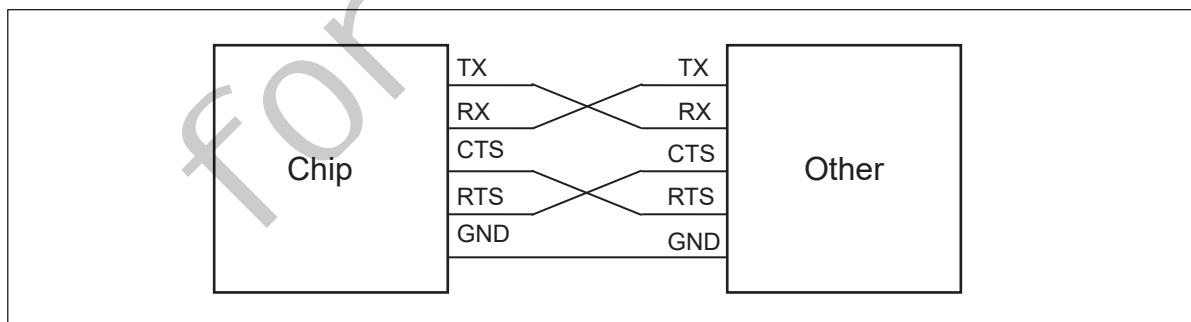


图 10.6: UART 硬件流控图

RTS (Require To Send, 发送请求) 为输出信号，用于指示芯片准备好可接收对方数据，低电平有效，低电平说明芯片可以接收数据。

CTS (Clear To Send, 清除发送) 为输入信号，用于判断芯片是否可以向对方发送数据，低电平有效，低电平说明芯片可以向对方发送数据。

当使用硬件流控功能时，芯片端输出信号 RTS 为低电平表示请求对方发送数据，RTS 为高电平表示通知对方中止数据发送。当芯片检测到输入信号 CTS 拉高时，TX 会停止发送数据，直到检测到 CTS 拉低时再继续发送。CTS 在通信过程中的任意时刻拉高或者拉低，不影响 TX 发送数据的连续性，对方收到的数据也是连续的。

发送器的硬件流控有两种方式：

- 硬件控制方式（寄存器 `uart_sw_mode` 中的 `cr_urx_rts_sw_mode` 等于 0）：接收未使能（寄存器 `urx_config` 中的 `cr_urx_en` 为 0）或 RX FIFO 剩余可用空间为一个字节时 RTS 拉高。
- 软件控制方式（寄存器 `uart_sw_mode` 中的 `cr_urx_rts_sw_mode` 等于 1）：软件可以通过配置寄存器 `urx_sw_mode` 中的 `cr_urx_rts_sw_val` 改变 RTS 的电平。

### 10.3.9 DMA 传输

UART 支持 DMA 传输。使用 DMA 传输，需要通过寄存器 `uart_fifo_config_1` 中的 `tx_fifo_th` 和 `rx_fifo_th` 分别设置 TX 和 RX FIFO 的阈值。当该模式使能后，如果 `uart_fifo_config_1` 中的 `tx_fifo_cnt` 大于 `tx_fifo_th`，则会触发 DMA TX 请求，配置好 DMA 后，DMA 在收到该请求时，会按照设定从内存中将数据搬运到 TX FIFO；如果 `uart_fifo_config_1` 中的 `rx_fifo_cnt` 大于 `rx_fifo_th`，则会触发 DMA RX 请求，配置好 DMA 后，DMA 在收到该请求时，会按照设定将 RX FIFO 的数据搬运到内存。

为了保证芯片 DMA TX Channel 搬运数据的正确性，Channel 配置中需要满足以下条件： $(transferWidth * burstSize) \leq (tx_fifo_th + 1)$ 。

为了保证芯片 DMA RX Channel 搬运数据的完整性，Channel 配置中需要满足以下条件： $(transferWidth * burstSize) = (rx_fifo_th + 1)$ 。

### 10.3.10 LIN 总线支持

本地互连网络（LIN）协议基于 Volvo 衍生公司 Volcano 通信技术公司（VCT）开发的 Volcano-Lite 技术。LIN 是 CAN 和 SAE J1850 协议的补充性协议，针对时间要求不高或不需要精确容错的应用（因为 LIN 没有 CAN 协议那样可靠）。LIN 的目标是易于使用，作为 CAN 协议的低成本替代品。LIN 在车辆中可以使用的场合包括车窗升降器、后视镜、雨刷和雨量传感器。

UART 模块支持 LIN 总线模式，LIN 总线采用主从模式，数据总是由主节点发起，主节点发送的帧（头）包含三个部分：同步间隔字段、同步字节字段和一个标识符字段。一个典型的 LIN 数据传输如图所示：

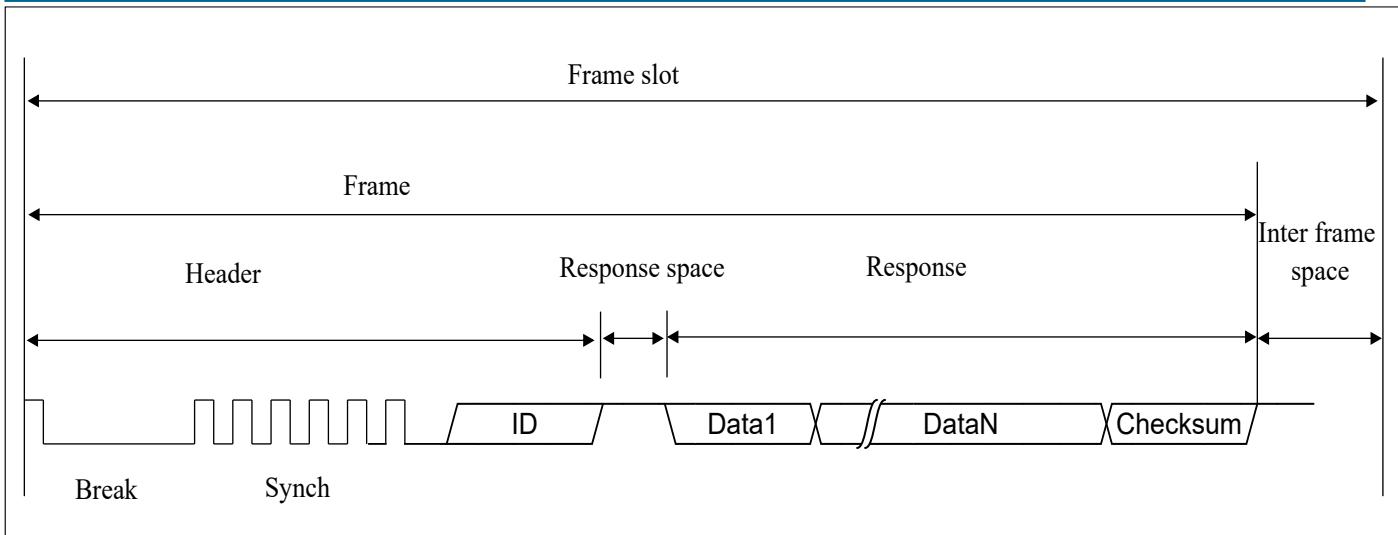


图 10.7: 一个典型的 LIN 帧

### 1. LIN 的 Break 域

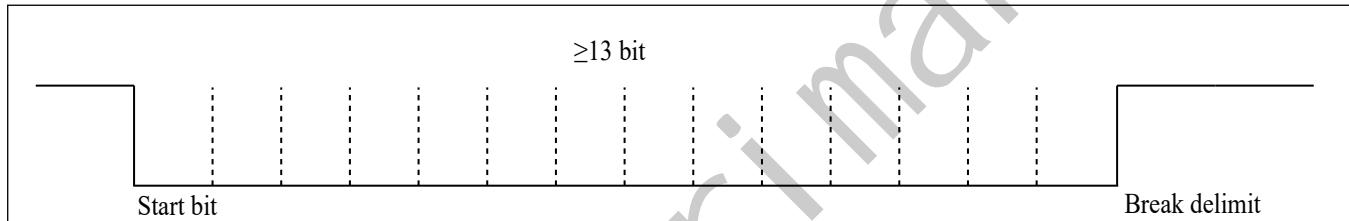


图 10.8: LIN 的 Break 域

同步间隔字段表示报文的开始，至少 13 个显性位（包括起始位）。同步间隔以一个“间隔分隔符”结束，该分隔符至少包含一个隐性位。LIN 帧中的 Break 长度可以通过 utx\_config 中的 cr\_utx\_bit\_cnt\_b 来设定。

### 2. LIN 的 Sync 域

发送同步字节字段来确定两个下降沿之间的时间，从而确定主节点使用的传输速率。位模式是 0x55（01010101，最大下降沿数量）。

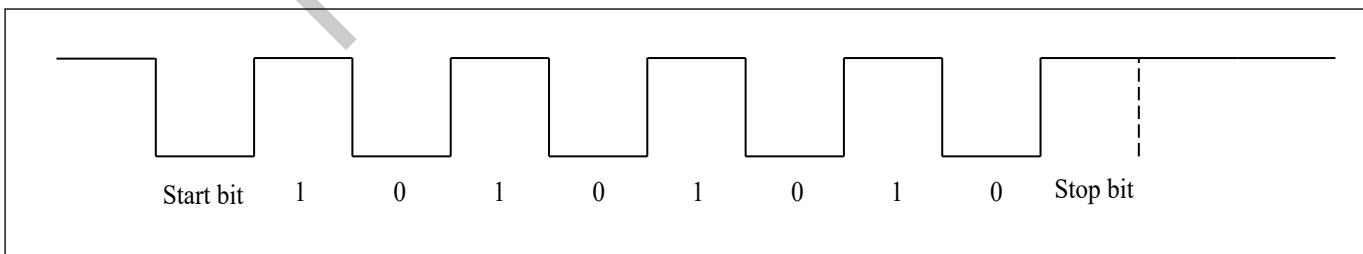


图 10.9: LIN 的 Sync 域

### 3. LIN 的 ID 域

标识符字段包含 6 位长的标识符和两个奇偶校验位。6 位标识符包含关于发送方和接收方的信息，以及响应中要求的

字节数。奇偶校验位如下进行计算：校验位 P0 是 ID0、ID1、ID2 和 ID4 之间进行逻辑“或”运算的结果。校验位 P1 是 ID1、ID3、ID4 和 ID5 之间逻辑“或”运算后再进行反转的结果。

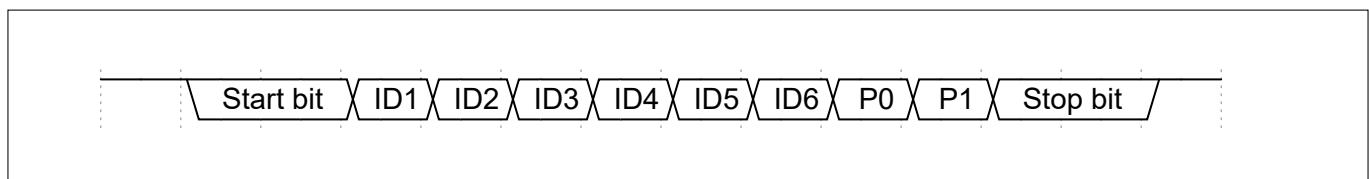


图 10.10: LIN 的 ID 域

从节点等待同步间隔字段，然后通过同步字节字段开始主从节点之间的同步。根据主节点发送的标识符，从节点将进行接收、发送或什么都不做。应该进行发送的从节点发送主节点请求的字节数，然后以一个检验和字段结束传输。

使用 LIN 模式需要设置寄存器 `utx_config` 中的 `cr_utx_lin_en`。通过配置 `cr_utx_bit_cnt_b`，应确保同步间隔段至少是由 13 位的显性电平组成。

### 10.3.11 RS485 模式

UART 模块支持 RS485 模式，通过设置寄存器 `UTX_RS485_CFG` 中的 `cr_utx_rs485_en` 可以让 UART 模块工作在 RS485 模式下，此时，可以通过外接一个 RS485 的收发器连接到 RS485 总线上，在该模式下，模块中的 RTS 引脚，会转为 RS485 收发器的 Dir 功能，当 UART 模块有数据需要发送时，会自动控制 RTS 引脚为高电平，让 RS485 收发器把数据发送到总线上，反之，当 UART 模块没有数据需要发送时，会自动控制 RTS 为低电平，让 RS485 收发器处在接收状态。

使用该模式需要设置寄存器 `UTX_RS485_CFG` 中的 `cr_utx_rs485_pol` 和 `cr_utx_rs485_en`。

## 10.4 UART 中断

UART 有着丰富的中断控制，包括以下几种中断模式：

- TX 传输完成中断
- RX 接收完成中断
- TX FIFO 请求中断
- RX FIFO 请求中断
- RX 超时中断
- RX 奇偶校验错误中断
- TX FIFO 溢出中断
- RX FIFO 溢出中断
- RX BCR 中断
- LIN 同步错误中断

- 通用模式自动波特率检测中断
- 固定字符模式自动波特率检测中断

#### 10.4.1 TX/RX 传输完成中断

TX 和 RX 可以通过寄存器 `utx_config` 和 `urx_config` 的高 16 位分别设置传输长度值，当传输的字节数达到这个数值时，就会触发对应的 TX/RX 传输完成中断，如果启用了 TX 自由运行（FreeRun）模式，则不会产生 TX 传输完成中断，TX 功能也不会停止。如果未启用 TX 自由运行（FreeRun）模式，则产生该中断的同时，TX 功能也会停止，用户如果需要继续使用 TX，必须重新启用 TX 功能。与 TX 不同，RX 功能在产生 RX 传输完成中断后仍可以正常使用。

对于 TX 而言，如果连续向 TX FIFO 中填入的数据大于设置的传输长度值，则 TX 引脚上只会传输设定长度值的数据，多余的数据会留存在 TX FIFO 中，重新启用 TX 功能后，TX FIFO 中剩余的数据将会发出。

例如：设置 TX 传输长度值为 64，启用 TX 功能，先向 TX FIFO 中填入 63 个字节，这 63 个字节会在引脚上传输，但是没有 TX 传输完成中断产生，再向 TX FIFO 中填入 1 字节，此时发送的长度达到 TX 设置的传输长度值，会产生发送完成中断，并且 TX 功能会停止。继续向 TX FIFO 中填入 1 字节，会发现引脚上没有数据传输，该字节还保留在 TX FIFO 中，将 TX 功能关闭后重新启用，该字节在引脚上发出。

对于 RX 而言，如果对方发送的数据长度超过了设置的传输长度，RX 在产生 RX 传输完成中断后依旧可以继续接收数据。

例如：设置 RX 传输长度值为 16，对方发送了 32 个字节数据，RX 会在收到 16 个字节数据时产生 RX 传输完成中断，并继续接收剩下的 16 个字节数据，全部保存在 RX FIFO 中。

#### 10.4.2 TX/RX FIFO 请求中断

当 `uart_fifo_config_1` 中的 `tx_fifo_cnt` 大于 `tx_fifo_th` 时，产生 TX FIFO 请求中断，当条件不满足时该中断标志会自动清除。当 `uart_fifo_config_1` 中的 `rx_fifo_cnt` 大于 `rx_fifo_th` 时，产生 RX FIFO 请求中断，当条件不满足时该中断标志会自动清除。TX/RX 均可支持多次发送/接收，并非一次性达到 `tx_fifo_th/rx_fifo_th` 设置的值。

例如：

1. 设置寄存器 `uart_fifo_config_1` 中的 `tx_fifo_th/rx_fifo_th` 为 16。
2. 置位寄存器 `utx_config` 中的 `cr_utx_frm_en`，使能 free run mode。
3. 设置寄存器 `uart_int_mask` 中的 `cr_utx_frdy_mask/cr_urx_frdy_mask` 为 0，将 TX/RX 的 FIFO 中断打开。
4. 设置寄存器 `utx_config/urx_config` 中的 `cr_utx_en/cr_urx_en`，使能 TX/RX。
5. TX FIFO 中断：TX 会一直进入 FIFO 中断，当芯片发送 128 字节后，将 `cr_utx_frdy_mask` 置 1 屏蔽该中断。如果想再次进入 TX FIFO 中断，`cr_utx_frdy_mask` 置 0 即可。
6. RX FIFO 中断：对方先发送 15 字节数，无中断产生，此时获取 `rx_fifo_cnt` 值为 15，再次发送 1 个字节达到 `rx_fifo_th` 设置的值时产生中断。产生发送中断后，对方再次发送数据，芯片可以收到数据。

### 10.4.3 RX 超时中断

RX 超时中断产生条件是：在上一次接收数据之后，接收器会开始计时，当计时值超过超时门限值仍未收到下一个数据时触发中断。超时门限值以通信的 bit 为单位。

当对方给芯片发送数据时，达到设置的超时时长后，会产生一次超时中断。

### 10.4.4 RX 奇偶校验错误中断

RX 奇偶校验错误中断会发生在奇偶校验出错时。但是不影响 RX 正确接收并解析对方发来的数据。RX 在接收数据的时候，会取前 8bit 作为数据位，忽略奇偶校验位，因此数据一致性完整。

例如设置寄存器 utx\_config/urx\_config 中的 cr\_utx\_prt\_en/cr\_urx\_prt\_en 使能奇偶校验；设置寄存器 utx\_config/urx\_config 中的 cr\_utx\_prt\_sel/cr\_urx\_prt\_sel 选择奇偶校验种类；当对方使用奇校验/偶校验给芯片发送数据时，芯片的 RX 不开启奇偶校验，RX 仍能收到正确的数据。

### 10.4.5 TX/RX FIFO 溢出中断

如果 TX/RX FIFO 发生了上溢或者下溢，会触发对应的溢出中断，当 FIFO 清除位：寄存器 UART\_FIFO\_CONFIG\_0 中的 tx\_fifo\_clr 和 rx\_fifo\_clr 被置 1 时，对应的 FIFO 会被清空，同时溢出中断标志会自动清除。可以通过寄存器 UART\_INT\_STS 查询溢出的具体中断状态，通过向寄存器 UART\_INT\_CLEAR 相应的位写 1 清除中断。

### 10.4.6 RX BCR 中断

当 RX 接收到的数据字节数达到寄存器 urx\_bcr\_int\_cfg 中的 cr\_urx\_bcr\_value 设置的值时，会产生 Byte Count Reached(BCR) 中断。

RX BCR 中断与 RX END 中断都可以重复产生，两者区别在于：RX END 中断内部的计数器在每次产生该中断时清零并重新开始计数，所以无法通过查询该计数器的状态得知当前总共收到了多少数据；而 RX BCR 中断内部的计数器在产生该中断时不会清零，而是继续累加，所以可以通过查询寄存器 urx\_bcr\_int\_cfg 中的 sts\_urx\_bcr\_count 位来获取当前接收的数据总量。如果想清除 RX BCR 中断内部的计数器，只需要向寄存器 uart\_int\_clear 中的 cr\_urx\_bcr\_clr 位写 1。

### 10.4.7 LIN 同步错误中断

当 utx\_config 中的 cr\_utx\_lin\_en 使能后，进入 LIN 模式，在 LIN 模式下，如果接收数据时，没有检测到 LIN 总线的同步段，则产生 LIN 同步错误中断。

#### 10.4.8 通用/固定字符模式自动波特率检测中断

在自动波特率模式下，根据配置对应的中断，在自动波特率检测通过时，可以产生通用模式自动波特率检测中断或者固定字符模式自动波特率检测中断。

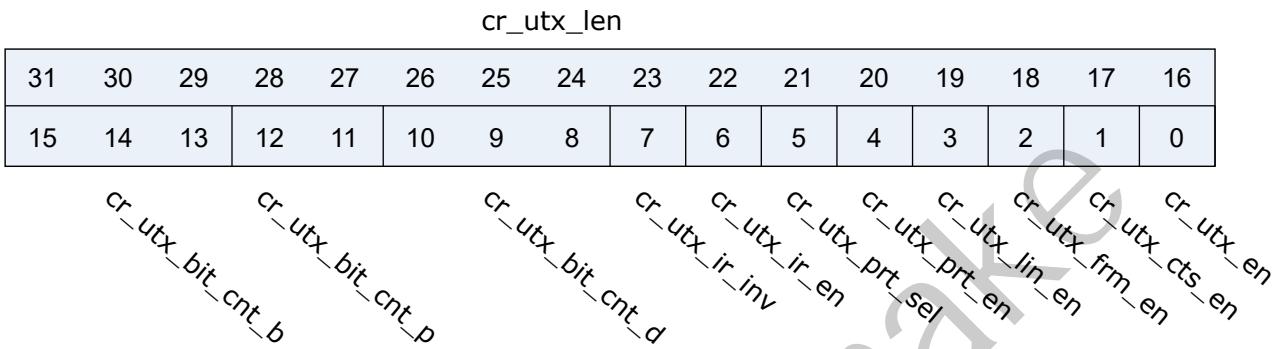
### 10.5 寄存器描述

名称	描述
utx_config	TX configure
urx_config	RX configure
uart_bit_prd	Baud rate setting
data_config	LSB/MSB-first select
utx_ir_position	IR mode TX setting
urx_ir_position	IR mode RX setting
urx_rto_timer	Time-out value setting
uart_sw_mode	software mode
uart_int_sts	UART interrupt status
uart_int_mask	UART interrupt mask
uart_int_clear	UART interrupt clear
uart_int_en	UART interrupt enable
uart_status	Bus busy status
sts_urx_abr_prd	Aute baud rate detection
urx_abr_prd_b01	ABR detection bit 0/1
urx_abr_prd_b23	ABR detection bit 2/3
urx_abr_prd_b45	ABR detection bit 4/5
urx_abr_prd_b67	ABR detection bit 6/7
urx_abr_pw_tol	0x55 ABR max allowable error
urx_bcr_int_cfg	BCR interrupt counter
utx_rs485_cfg	RS-485 configure
uart_fifo_config_0	FIFO status and DMA mode
uart_fifo_config_1	FIFO threshold and available count
uart_fifo_wdata	TX FIFO

名称	描述
uart_fifo_rdata	RX FIFO

### 10.5.1 utx\_config

地址: 0x40010000

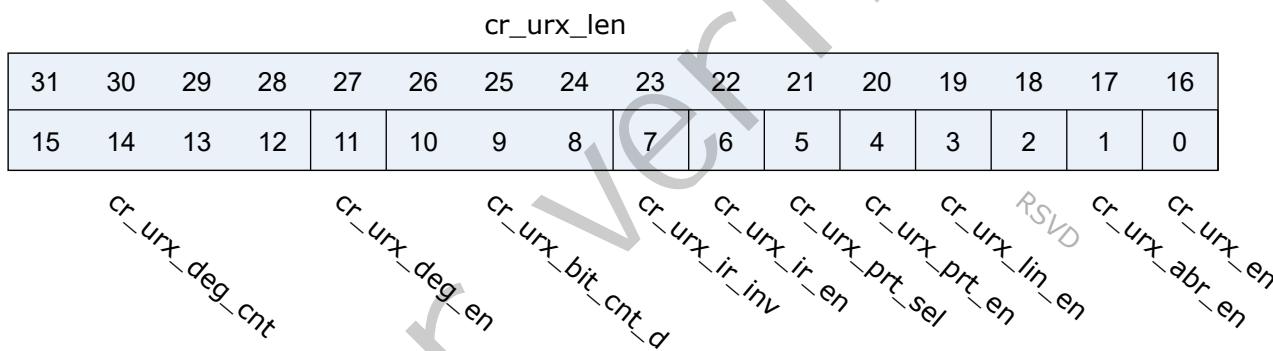


位	名称	权限	复位值	描述
31:16	cr_utx_len	r/w	16'd0	Length of UART TX data transfer (Unit: character/byte), TX end interrupt will be triggered when TX send cr_utx_len+1 bytes data (Don't-care if cr_utx_frm_en is enabled)
15:13	cr_utx_bit_cnt_b	r/w	3'd4	UART TX BREAK bit count (for LIN protocol) Note: Additional 8 bit times will be added since LIN Break field requires at least 13 bit times 4: 4+1+8=13 bit 5: 5+1+8=14 bit ... 4: 4+1+8=13 bit 5: 5+1+8=14 bit ... 4: 4+1+8=13 bit 5: 5+1+8=14 bit ...
12:11	cr_utx_bit_cnt_p	r/w	2'd1	UART TX STOP bit count (unit: 0.5 bit) 0: 0.5 bit 1: 1 bit 2: 1.5 bit 3: 2 bit
10:8	cr_utx_bit_cnt_d	r/w	3'd7	UART TX DATA bit count for each character 4: 5 bit 5: 6 bit 6: 7 bit 7: 8 bit
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

位	名称	权限	复位值	描述
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

### 10.5.2 urx\_config

地址: 0x40010004



位	名称	权限	复位值	描述
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, like cr_utx_bit_cnt_d
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

位	名称	权限	复位值	描述
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

### 10.5.3 uart\_bit\_prd

地址: 0x40010008

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

位	名称	权限	复位值	描述
31:16	cr_urx_bit_prd	r/w	16'd255	Period of each UART RX bit, RX baud rate = uart clock / (cr_urx_bit_prd + 1)
15:0	cr_utx_bit_prd	r/w	16'd255	Period of each UART TX bit, TX baud rate = uart clock / (cr_utx_bit_prd + 1)

### 10.5.4 data\_config

地址: 0x4001000c

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

位	名称	权限	复位值	描述
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

### 10.5.5 utx\_ir\_position

地址: 0x40010010

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

位	名称	权限	复位值	描述
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

### 10.5.6 urx\_ir\_position

地址: 0x40010014

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

位	名称	权限	复位值	描述
31:16	RSVD			
15:0	cr_urx_ir_pos_s	r/w	16'd111	START position of UART RXD pulse recovered from IR signal

### 10.5.7 urx\_rto\_timer

地址: 0x40010018

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

位	名称	权限	复位值	描述
31:8	RSVD			
7:0	cr_urx_rto_value	r/w	8'd15	Time-out value for triggering RTO interrupt (unit: bit time)

### 10.5.8 uart\_sw\_mode

地址: 0x4001001c



| 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    |      |      |      |      |

位	名称	权限	复位值	描述
31:4	RSVD			
3	cr_urx_rts_sw_val	r/w	1'b0	UART RX RTS output SW control value 0: Low level 1: High level
2	cr_urx_rts_sw_mode	r/w	1'b0	UART RX RTS output SW control mode enable
1	cr_utx_txd_sw_val	r/w	1'b0	UART TX TXD output SW control value 0: Low level 1: High level
0	cr_utx_txd_sw_mode	r/w	1'b0	UART TX TXD output SW control mode enable

### 10.5.9 uart\_int\_sts

地址: 0x40010020

位	名称	权限	复位值	描述
31:12	RSVD			
11	urx_ad5_int	r	1'b0	UART RX ABR Detection finish interrupt using codeword 0x55

位	名称	权限	复位值	描述
10	urx_ads_int	r	1'b0	UART RX ABR Detection finish interrupt using START bit
9	urx_bcr_int	r	1'b0	UART RX byte count reached interrupt
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_frdy_int	r	1'b0	UART RX FIFO ready ( $rx\_fifo\_cnt > rx\_fifo\_th$ ) interrupt, auto-cleared when data is popped
2	utx_frdy_int	r	1'b1	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)

### 10.5.10 uart\_int\_mask

地址: 0x40010024

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_ad5_mask cr_urx_bcr_mask cr_urx_lse_mask cr_urx_fer_mask cr_utx_fer_mask cr_urx_pce_mask cr_urx_rto_mask cr_urx_frdy_mask cr_urx_end_mask cr_utx_end_mask															

位	名称	权限	复位值	描述
31:12	RSVD			
11	cr_urx_ad5_mask	r/w	1'b1	Interrupt mask of urx_ad5_int

位	名称	权限	复位值	描述
10	cr_urx_ads_mask	r/w	1'b1	Interrupt mask of urx_ads_int
9	cr_urx_bcr_mask	r/w	1'b1	Interrupt mask of urx_bcr_int
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
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_frdy_mask	r/w	1'b1	Interrupt mask of urx_frdy_int
2	cr_utx_frdy_mask	r/w	1'b1	Interrupt mask of utx_frdy_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

### 10.5.11 uart\_int\_clear

地址: 0x40010028

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 cr\_urx\_ad5\_clr cr\_urx\_ads\_clr cr\_urx\_bcr\_clr rsrvd rsrvd cr\_urx\_lse\_clr cr\_urx\_pce\_clr cr\_urx\_rto\_clr rsrvd cr\_urx\_end\_clr cr\_utx\_end\_clr

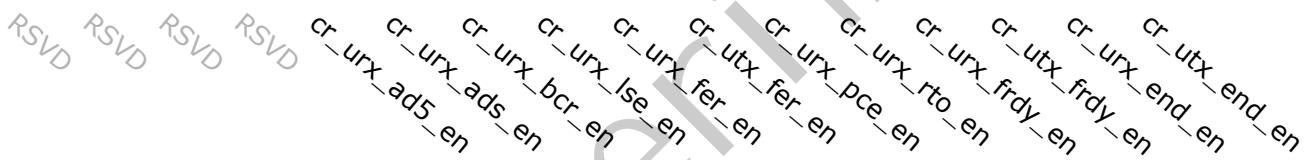
位	名称	权限	复位值	描述
31:12	RSVD			
11	cr_urx_ad5_clr	w1c	1'b0	Interrupt clear of urx_ad5_int
10	cr_urx_ads_clr	w1c	1'b0	Interrupt clear of urx_ads_int
9	cr_urx_bcr_clr	w1c	1'b0	Interrupt clear of urx_bcr_int
8	cr_urx_lse_clr	w1c	1'b0	Interrupt clear of urx_lse_int
7	rsrvd	rsrvd	1'b0	
6	rsrvd	rsrvd	1'b0	

位	名称	权限	复位值	描述
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	rsvd	rsvd	1'b0	
2	rsvd	rsvd	1'b0	
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

### 10.5.12 uart\_int\_en

地址: 0x4001002c

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



位	名称	权限	复位值	描述
31:12	RSVD			
11	cr_urx_ad5_en	r/w	1'b1	Interrupt enable of urx_ad5_int
10	cr_urx_ads_en	r/w	1'b1	Interrupt enable of urx_ads_int
9	cr_urx_bcr_en	r/w	1'b1	Interrupt enable of urx_bcr_int
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_frdy_en	r/w	1'b1	Interrupt enable of urx_frdy_int
2	cr_utx_frdy_en	r/w	1'b1	Interrupt enable of utx_frdy_int
1	cr_urx_end_en	r/w	1'b1	Interrupt enable of urx_end_int

位	名称	权限	复位值	描述
0	cr_utx_end_en	r/w	1'b1	Interrupt enable of utx_end_int

### 10.5.13 uart\_status

地址: 0x40010030

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		
15	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD		
sts_urx_bus_busy	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	sts_utx_bus_busy	RSVD		
sts_utx_bus_busy	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD		
31:2	RSVD																
1	sts_urx_bus_busy							r	1'b0		Indicator of UART RX bus busy 0: Idle 1: Busy						
0	sts_utx_bus_busy							r	1'b0		Indicator of UART TX bus busy 0: Idle 1: Busy						

### 10.5.14 sts\_urx\_abr\_prd

地址: 0x40010034

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

位	名称	权限	复位值	描述
31:16	sts_urx_abr_prd_0x55	r	16'd0	Bit period of Auto Baud Rate detection using codeword 0x55, baud rate = uart clock / (sts_urx_abr_prd_0x55 + 1)
15:0	sts_urx_abr_prd_start	r	16'd0	Bit period of Auto Baud Rate detection using START bit, baud rate = uart clock / (sts_urx_abr_prd_start + 1)

### 10.5.15 urx\_abr\_prd\_b01

地址: 0x40010038

sts\_urx\_abr\_prd\_bit1

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\_bit0

位	名称	权限	复位值	描述
31:16	sts_urx_abr_prd_bit1	r	16'd0	Bit period of Auto Baud Rate detection - bit[1]
15:0	sts_urx_abr_prd_bit0	r	16'd0	Bit period of Auto Baud Rate detection - bit[0]

### 10.5.16 urx\_abr\_prd\_b23

地址: 0x4001003c

sts\_urx\_abr\_prd\_bit3

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\_bit2

位	名称	权限	复位值	描述
31:16	sts_urx_abr_prd_bit3	r	16'd0	Bit period of Auto Baud Rate detection - bit[3]
15:0	sts_urx_abr_prd_bit2	r	16'd0	Bit period of Auto Baud Rate detection - bit[2]

### 10.5.17 urx\_abr\_prd\_b45

地址: 0x40010040

sts\_urx\_abr\_prd\_bit5

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\_bit4

位	名称	权限	复位值	描述
31:16	sts_urx_abr_prd_bit5	r	16'd0	Bit period of Auto Baud Rate detection - bit[5]
15:0	sts_urx_abr_prd_bit4	r	16'd0	Bit period of Auto Baud Rate detection - bit[4]

### 10.5.18 urx\_abr\_prd\_b67

地址: 0x40010044

sts\_urx\_abr\_prd\_bit7

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\_bit6

位	名称	权限	复位值	描述
31:16	sts_urx_abr_prd_bit7	r	16'd0	Bit period of Auto Baud Rate detection - bit[7]
15:0	sts_urx_abr_prd_bit6	r	16'd0	Bit period of Auto Baud Rate detection - bit[6]

### 10.5.19 urx\_abr\_pw\_tol

地址: 0x40010048

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\_abr\_pw\_tol

位	名称	权限	复位值	描述
31:8	RSVD			
7:0	cr_urx_abr_pw_tol	r/w	8'd3	Auto Baud Rate detection pulse-width tolerance for using codeword 0x55

### 10.5.20 urx\_bcr\_int\_cfg

地址: 0x40010050

sts\_urx\_bcr\_count

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\_bcr\_value

位	名称	权限	复位值	描述
31:16	sts_urx_bcr_count	r	16'd0	Current byte count of urx_bcr_int counter, auto-cleared by cr_urx_bcr_clr
15:0	cr_urx_bcr_value	r/w	16'hFFFF	Byte count setting for urx_bcr_int counter

### 10.5.21 utx\_rs485\_cfg

地址: 0x40010054

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 cr\_utx\_rs485\_en cr\_utx\_rs485\_polarity

位	名称	权限	复位值	描述
31:2	RSVD			

位	名称	权限	复位值	描述
1	cr_utx_rs485_pol	r/w	1'b1	DE pin polarity in RS-485 transceiver mode 1'b0: DE is active-low 1'b1: DE is active-high
0	cr_utx_rs485_en	r/w	1'b0	Enable signal for RS-485 transceiver mode 1'b0: Disabled, normal UART 1'b1: Enabled, IO is connected to RS-485 transceiver and RTS_O becomes DE function

## 10.5.22 uart\_fifo\_config\_0

地址: 0x40010080

位	名称	权限	复位值	描述
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, RX FIFO will be empty when write 1 to this bit
2	tx_fifo_clr	w1c	1'b0	Clear signal of TX FIFO, TX FIFO will be empty when write 1 to this bit
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

### 10.5.23 uart\_fifo\_config\_1

地址: 0x40010084

uart_fifo_config_1															
rx_fifo_th								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

位	名称	权限	复位值	描述
31:29	RSVD			
28:24	rx_fifo_th	r/w	5'd0	RX FIFO threshold, dma_rx_req and rx_fifo_int will not be asserted if rx_fifo_cnt is less than this value
23:21	RSVD			
20:16	tx_fifo_th	r/w	5'd0	TX FIFO threshold, dma_tx_req and tx_fifo_int will not be asserted if tx_fifo_cnt is less than this value
15:14	RSVD			
13:8	rx_fifo_cnt	r	6'd0	RX FIFO available count, means byte count of data received in RX FIFO
7:6	RSVD			
5:0	tx_fifo_cnt	r	6'd32	TX FIFO available count, means empty space remained in TX FIFO

### 10.5.24 uart\_fifo\_wdata

地址: 0x40010088

uart_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

位	名称	权限	复位值	描述
31:8	RSVD			
7:0	uart_fifo_wdata	w	x	TX FIFO, size is 32*1=32-byte

### 10.5.25 uart\_fifo\_rdata

地址: 0x4001008c

| 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

位	名称	权限	复位值	描述
31:8	RSVD			
7:0	uart_fifo_rdata	r	8'h0	RX FIFO, size is 32*1=32-byte

# 11

## I2C

### 11.1 简介

I2C (Inter-Integrated Circuit) 是一种串行通讯总线，使用多主从架构，用来连接低速外围装置。每个器件都有一个唯一的地址识别，并且都可以作为发送器或接收器。如果有多个主机同时操作一个从机，数据传输可以通过冲突检测和仲裁机制防止数据被破坏。BL616/BL618 包含 2 组 I2C 控制器主机，可灵活配置 slaveAddr、subAddr 以及传输数据，方便与从设备通信，提供 2 个 word 深度的 fifo，提供中断功能，可搭配 DMA 使用提高效率，可灵活调整时钟频率。

### 11.2 主要特征

- 支持主机模式
- 支持多主机模式和仲裁功能
- 时钟频率可灵活调整
- 可同时设定 7 位地址格式和 10 位地址格式
- 支持 DMA 传输模式
- 支持多种中断机制

### 11.3 功能描述

引脚列表：

表 11.1: I2C 引脚

名称	类型	描述
I2Cx_SCL	输入/输出	I2C 串行时钟信号
I2Cx_SDA	输入/输出	I2C 串行数据信号

### 11.3.1 起始和停止条件

所有传输都由起始条件 (START condition) 开始，以停止条件 (STOP condition) 结束。起始条件和停止条件一般都由主机产生，总线在起始条件后处于总线忙的状态，在停止条件后至下一次起始条件前处于空闲状态。

起始条件：SCL 为高电平时 SDA 产生一高至低的电平转换；

停止条件：SCL 为高电平时 SDA 产生一低至高的电平转换。

波形示意图如下：

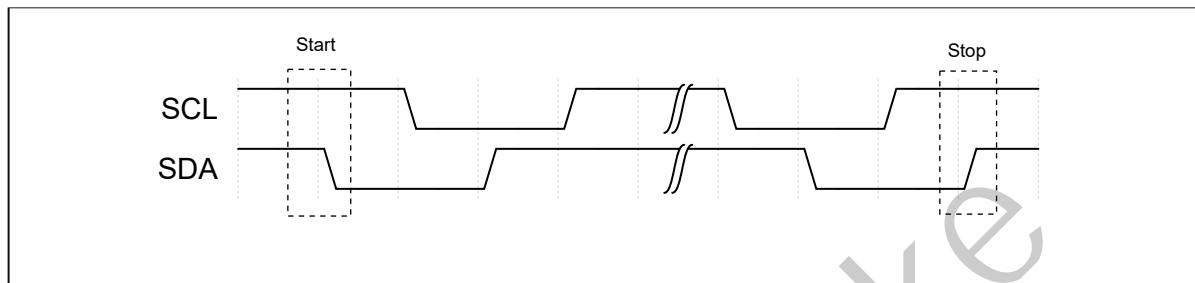


图 11.1: I<sup>2</sup>C 起始和停止条件

### 11.3.2 数据传输格式

## 1. 7 位寻址模式

传输的第一个8位为寻址字节，包括7位从机地址和1位方向位。数据由主机发送或接收是由主机所送出的第1个字节的第8位控制，若为0表示数据由主机发送；为1则表示数据由主机接收，紧接着从机发出应答位(ACK)，主机在收到应答后，开始传输指定长度的数据。在数据传输完成后，主机发出停止信号，数据传输格式如下：

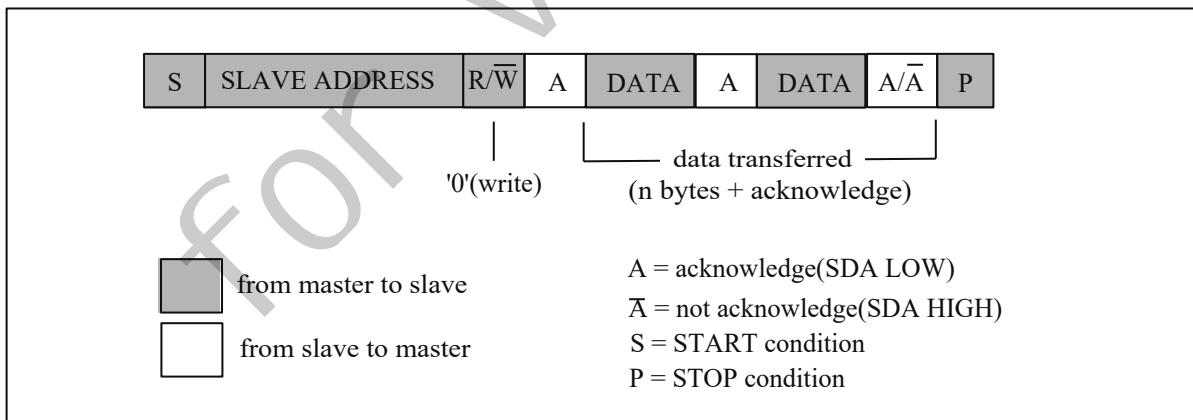


图 11.2: 主发送和从接收的数据格式

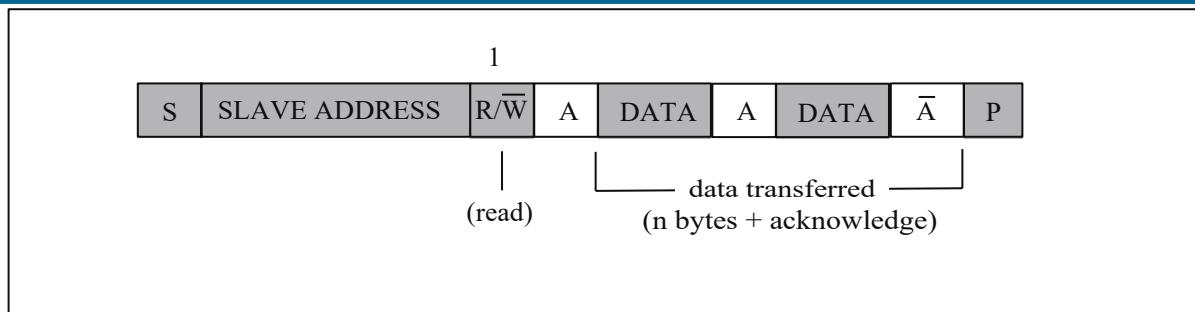


图 11.3: 主接收和从发送的数据格式

主发送和从接收的时序

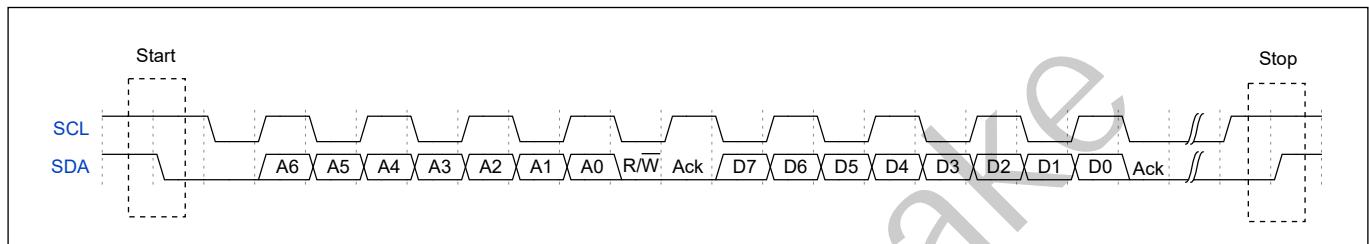


图 11.4: 主发送和从接收的时序

主接收和从发送的时序

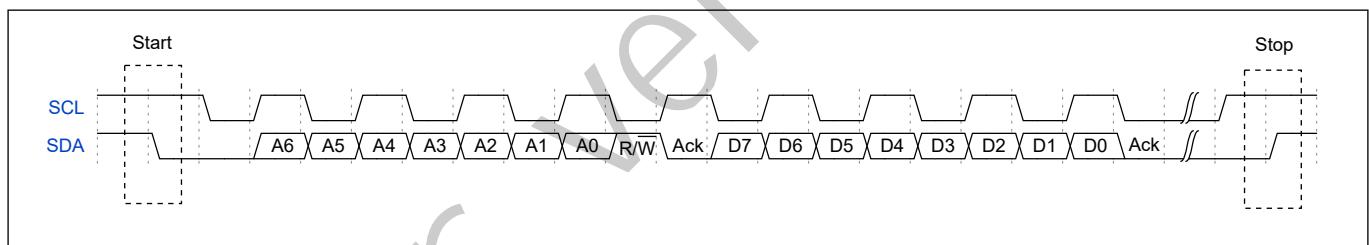


图 11.5: 主接收和从发送的时序

## 2. 10 位寻址模式

使用 10 位寻址模式时，需要将寄存器 i2c\_config 中的 cr\_i2c\_10b\_addr\_en 置 1。

10bit 的从机地址由开始条件 (S) 或重复开始条件 (Sr) 后的两个字节组成。第一个字节的前 7 位是 1111 0XX，XX 是 10 bit 地址的最高有效位的前两位。第一个字节的第 8 bit 是读写位，决定传输方向。尽管 1111 XXX 有 8 种可能的组合，只有 1111 0XX 这四种可以用于 10 bit 寻址，剩下的 1111 1XX 这四种是为将来 I2C 扩展用的。数据传输格式如下：

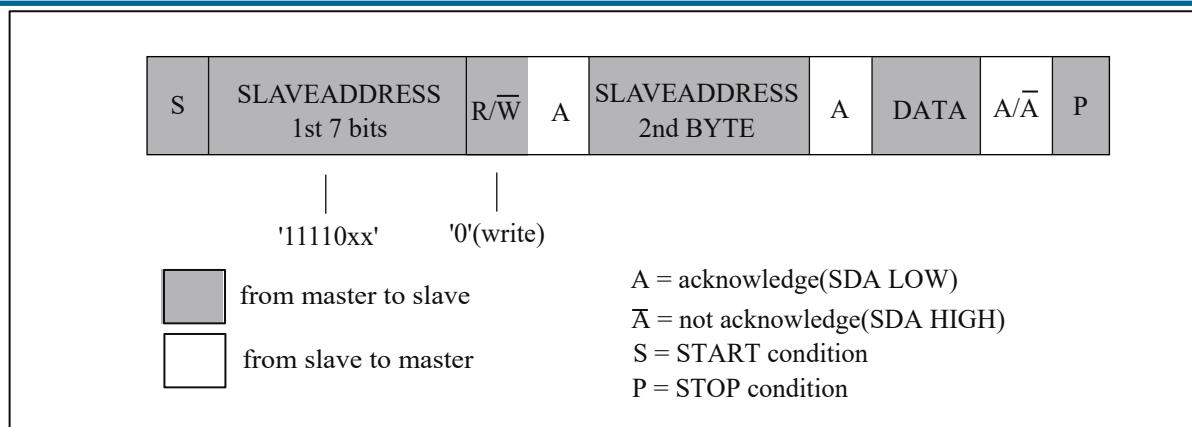


图 11.6: 主发送和从接收的数据格式 (10bit 从地址)

当接收到开始条件后的 10bit 地址时，从机开始匹配接收到的第一个字节 (1111 0XX)，并检查第八个 bit(读写位) 是否为 0。有可能多个设备都匹配并产生应答 A。接下来所有从机开始匹配第二个字节的 8 个 bit(XXXX XXXX)，这时就只有一个从机匹配并产生应答 A，被主机寻址匹配的从机会保持被寻址的状态，直到接收到终止条件或者是重复开始条件。

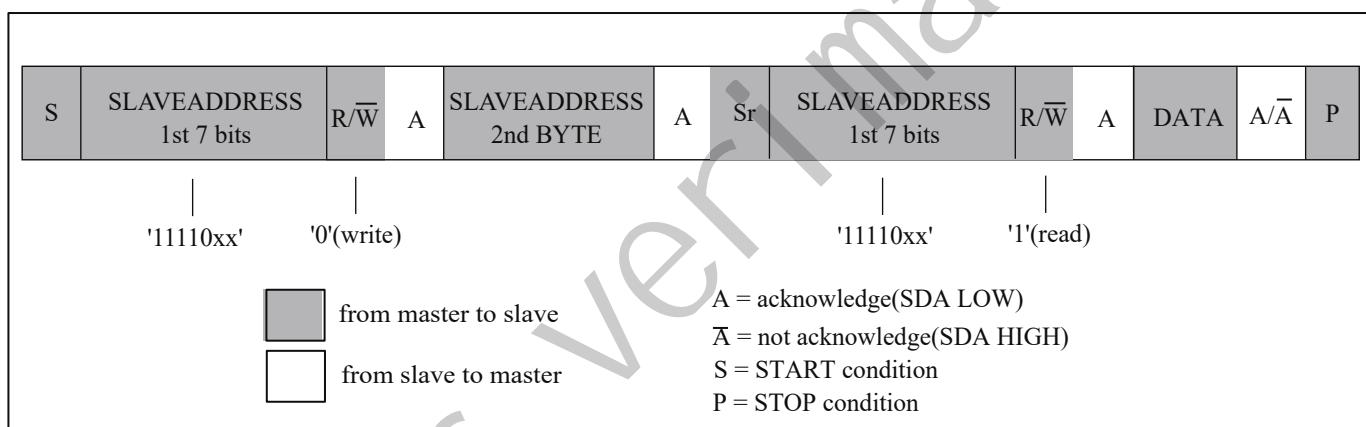


图 11.7: 主接收和从发送的数据格式 (10bit 从地址)

在第二个应答 A 之前，处理过程与上面的主-发送器寻址从-接收器一致。在重复开始条件 (Sr) 之后，匹配的从机会保持被寻址的状态。从机会检查 Sr 之后的第一个字节的前 7bit 是否正确，并测试第 8bit 是否为 1 (读)。如果匹配，从机就认定它作为发送器寻址成功并产生应答 A。从-发送器会保持被寻址的状态，直到接收到终止条件 (P) 或者重复开始条件 (Sr)。接收到重复开始条件 (Sr) 后，所有的从机会与 11110XX 匹配并测试第八位 (读写位)。然而它们不会寻址到，因为对于 10bit 设备，读写位是 1，或者对于 7bit 的设备，1111 0XX 的从机地址不匹配。

### 11.3.3 仲裁

当 I2C 总线存在多个主机时，可能会发生多个主机同时启动传输的情况，此时必须要依靠仲裁机制来决定哪个主机有权利完成接下来的数据传输，其余主机则须放弃对总线的控制权，等到总线再次空出来后才能再次启动传输。

在传输过程中，所有主机都需要在 SCL 为高电平时检查 SDA 是否与自己所想发送的数据相符，当 SDA 电平与预期不同时，表示有别的主机也在同时进行传输，而发现 SDA 电平不同的主机则失去此次对总线的控制权，由其他主机完成数据传输。

两主机同时传输数据并启动仲裁机制的波形示意图如下：

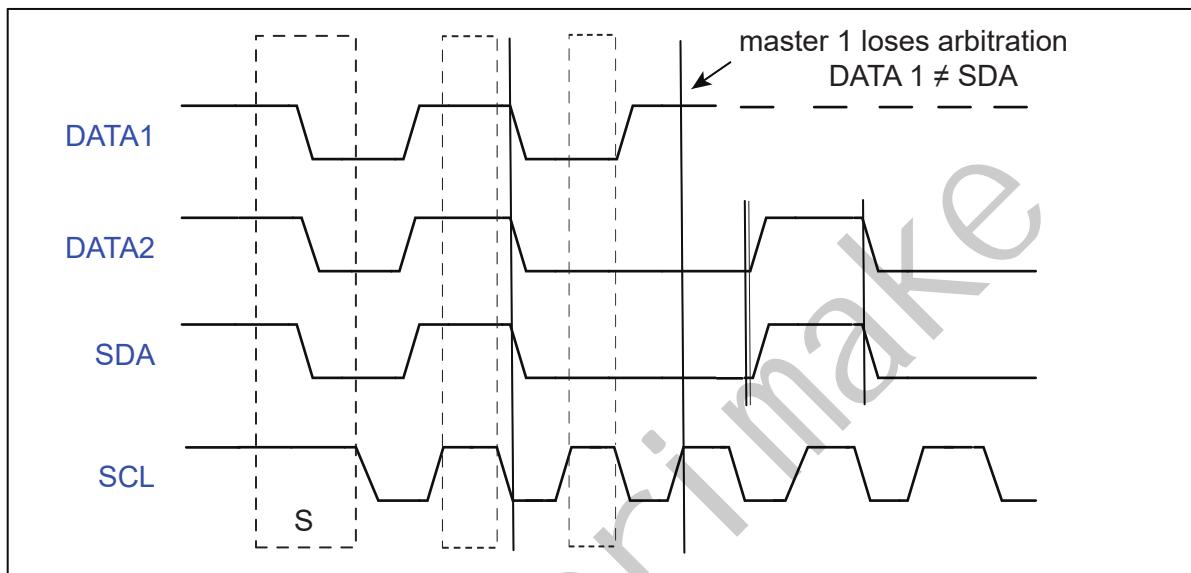


图 11.8: 同时传输数据波形示意图

## 11.4 I2C 时钟设定

I2C 的时钟可由 `bclk(bus clock)` 和 `xclk` 而来，可以在其基础上做分频处理。寄存器 `i2c_prd_data` 可以对数据段的时钟做分频处理。`i2c` 模块将数据发送分为 4 个阶段，每个阶段在寄存器中用单独一个字节来控制，每个阶段的采样个数是可以设置的，4 个采样数共同决定了 `i2c clock` 的分频系数。例如 `bclk` 为 32M，寄存器 `i2c_prd_data` 在不做配置默认情况下的值是 `0x0f0f0f0f`，那么 I2C 的时钟频率为  $32M / ((15 + 1) * 4) = 500K$ 。同理，寄存器 `i2c_prd_start` 和 `i2c_prd_stop` 也会分别对起始位和停止位的时钟做分频处理。

11.5 I2C 配置流程

### 11.5.1 配置项

- 读写标志位
  - 从设备地址
  - 从设备寄存器地址

- 从设备寄存器地址长度
- 数据 (发送时, 配置发送的数据; 接收时, 存储接收到的数据)
- 数据长度
- 使能信号

### 11.5.2 读写标志位

I2C 支持发送和接收两种工作状态, 寄存器 i2c\_config 中的 cr\_i2c\_pkt\_dir 表示发送或者接收状态, 设置为 0 时, 表示发送状态, 设置为 1 时, 表示接收状态。

### 11.5.3 从设备地址

每个对接 I2C 的从设备, 都会有唯一设备地址, 通常该地址是 7 bit, 将从设备地址写入寄存器 i2c\_config 中的 cr\_i2c\_slv\_addr, I2C 在将从设备地址发送出去之前, 会自动左移 1 位, 并在最低位补上发送接收方向位。

### 11.5.4 从设备寄存器地址

从设备寄存器地址表示 I2C 需要对从设备某个寄存器做读写操作的寄存器地址。将从设备寄存器地址写入寄存器 i2c\_sub\_addr, 同时需要将寄存器 i2c\_config 中的 cr\_i2c\_sub\_addr\_en 置 1。如果将寄存器 i2c\_config 中的 cr\_i2c\_sub\_addr\_en 置 0, 那么 I2C 主机发送时会跳过从设备寄存器地址段。

### 11.5.5 从设备寄存器地址长度

可以通过寄存器 i2c\_config 中的 cr\_i2c\_sub\_addr\_bc 设置从设备寄存器地址长度 (从设备寄存器地址长度为写入寄存器的值 + 1), 详细配置参考寄存器描述。

### 11.5.6 数据

数据部分表示需要发送到从设备的数据, 或者需要从从设备接收到的数据。当 I2C 发送数据时, 需要将数据依次以 word 为单位写入寄存器 i2c\_fifo\_wdata 中。当 I2C 接收数据时, 需要依次以 word 为单位从寄存器 i2c\_fifo\_rdata 中将数据读出来。

### 11.5.7 数据长度

可以通过寄存器 i2c\_config 中的 cr\_i2c\_pkt\_len 设置发送数据长度 (发送数据长度为写入寄存器的值 + 1), 发送数据最大长度为 256。

## 11.5.8 使能信号

将以上几项配置完成后，再将使能信号寄存器 i2c\_config 中的 cr\_i2c\_m\_en 置 1，就自动启动 I2C 发送流程。

当读写标志位配置为 0 时，I2C 发送数据，以发送 2 字节为例，发送流程：

1. 起始位
2. (从设备地址左移 1 位 + 0) + ACK
3. 从设备寄存器地址 + ACK
4. 1 字节数据 + ACK
5. 1 字节数据 + ACK
6. 停止位

当读写标志位配置为 1 时，I2C 接收数据，以接收 2 字节为例，主机发送流程：

1. 起始位
2. (从设备地址左移 1 位 + 0) + ACK
3. 从设备寄存器地址 + ACK
4. 起始位
5. (从设备地址左移 1 位 + 1) + ACK
6. 1 字节数据 + ACK
7. 1 字节数据 + ACK
8. 停止位

## 11.6 FIFO 管理

I2C FIFO 深度为 2 个 word，I2C 分为 RX FIFO 和 TX FIFO。寄存器 i2c\_fifo\_config\_1 中的 rx\_fifo\_cnt 表示 RX FIFO 中有多少数据（单位 word）需要读取。寄存器 i2c\_fifo\_config\_1 中的 tx\_fifo\_cnt 表示 TX FIFO 中剩余多少空间（单位 Word）可供写入。

I2C FIFO 状态：

- RX FIFO underflow：当 RX FIFO 中的数据被读取完毕或者为空，继续从 RX FIFO 中读取数据，则寄存器 i2c\_fifo\_config\_0 中的 rx\_fifo\_underflow 会被置 1；
- RX FIFO overflow：当 I2C 接收数据直到 RX FIFO 的 2 个 word 被填满后，在没有读取 RX FIFO 的情况下，I2C 再次接收到数据，寄存器 i2c\_fifo\_config\_0 中的 rx\_fifo\_overflow 会被置 1；
- TX FIFO underflow：当向 TX FIFO 中填入的数据大小不满足配置的 I2C 数据长度（i2c\_config 中的 cr\_i2c\_pkt\_len），并且已经没有新数据继续填入 TX FIFO 中时，寄存器 i2c\_fifo\_config\_0 中的 tx\_fifo\_underflow 会被置 1；

- TX FIFO overflow: 当 TX FIFO 的 2 个 word 被填满后, 在 TX FIFO 中的数据没有发出去之前, 再次向 TX FIFO 中填入数据, 寄存器 i2c\_fifo\_config\_0 中的 tx\_fifo\_overflow 会被置 1。

## 11.7 DMA 功能

I2C 可以使用 DMA 进行数据的发送和接收。将寄存器 i2c\_fifo\_config\_0 中的 i2c\_dma\_tx\_en 置 1, 则开启 DMA 发送模式, 为 I2C 分配好通道后, DMA 会将数据从存储区搬运到 i2c\_fifo\_wdata 寄存器中。将寄存器 i2c\_fifo\_config\_0 中的 i2c\_dma\_rx\_en 置 1, 则开启 DMA 接收模式, 为 I2C 分配好通道后, DMA 会将 i2c\_fifo\_rdata 寄存器中的数据搬运到存储区中。I2C 模块使用 DMA 功能时, 数据部分将由 DMA 自动完成搬运, 不需要 CPU 再将数据写入 I2C TX FIFO 或者从 I2C RX FIFO 中读取数据。

### 11.7.1 DMA 发送流程

1. 配置读写标志位为 0
2. 配置从设备地址
3. 配置从设备寄存器地址
4. 配置从设备寄存器地址长度
5. 数据长度
6. 使能信号寄存器置 1
7. 配置 DMA transfer size
8. 配置 DMA 源地址 transfer width
9. 配置 DMA 目的地地址 transfer width(需要注意 I2C 使用 DMA 功能时, 目的地地址 transfer width 需要设置为 32bits, 以 word 对齐使用)
10. 配置 DMA 源地址为存储发送数据的内存地址
11. 配置 DMA 目的地地址为 I2C TX FIFO 地址, 即 i2c\_fifo\_wdata
12. 使能 DMA

### 11.7.2 DMA 接收流程

1. 配置读写标志位为 1
2. 配置从设备地址
3. 配置从设备寄存器地址
4. 配置从设备寄存器地址长度
5. 数据长度
6. 使能信号寄存器置 1

7. 配置 DMA transfer size
8. 配置 DMA 源地址 transfer width(需要注意 I2C 使用 DMA 功能时，源地址 transfer width 需要设置为 32bits，以 word 对齐使用)
9. 配置 DMA 目的地址 transfer width
10. 配置 DMA 源地址为 I2C RX FIFO 地址，即 i2c\_fifo\_rdata
11. 配置 DMA 目的地址为存储接收数据的内存地址
12. 使能 DMA

## 11.8 中断

I2C 包括如下几种中断：

- I2C\_TRANS\_END\_INT: I2C 传输结束中断，当 I2C 完成一次传输时产生该中断；
- I2C\_TX\_FIFO\_READY\_INT: 当 i2c\_fifo\_config\_1 中的 tx\_fifo\_cnt 大于 tx\_fifo\_th 时，产生 TX FIFO 请求中断，当条件不满足时该中断标志会自动清除；
- I2C\_RX\_FIFO\_READY\_INT: 当 i2c\_fifo\_config\_1 中的 rx\_fifo\_cnt 大于 rx\_fifo\_th 时，产生 RX FIFO 请求中断，当条件不满足时该中断标志会自动清除；
- I2C\_NACK\_RECV\_INT: 当 I2C 模块检测到 NACK 状态时，产生 NACK 中断；
- I2C\_ARB\_LOST\_INT: I2C 仲裁丢失中断；
- I2C\_FIFO\_ERR\_INT: 当 TX/RX FIFO 发生 overflow 或 underflow 时，产生 FIFO ERROR 中断。

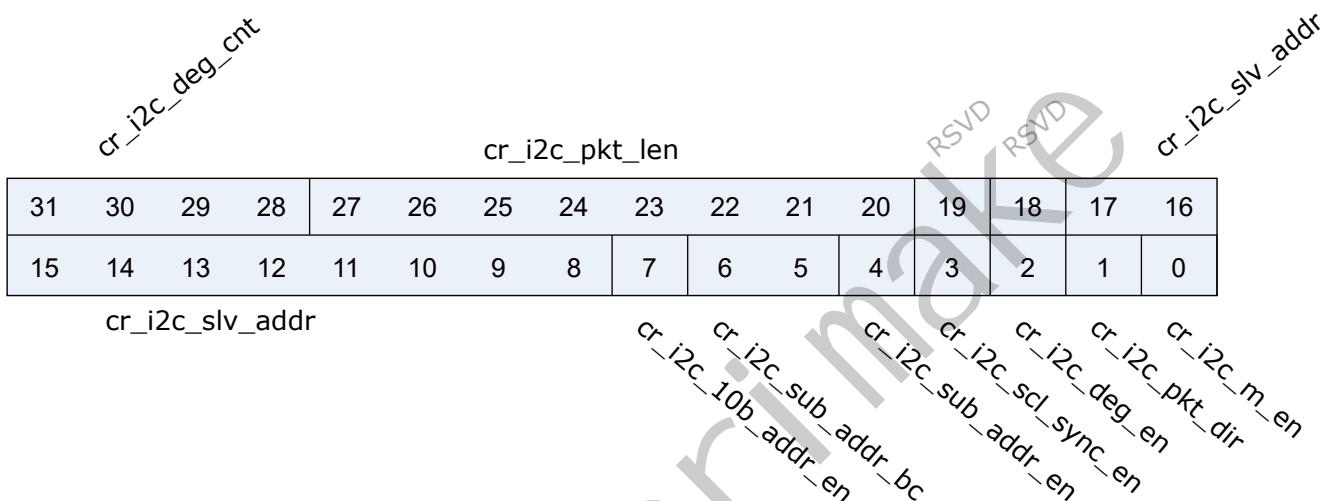
## 11.9 寄存器描述

名称	描述
i2c_config	
i2c_int_sts	
i2c_sub_addr	
i2c_bus_busy	
i2c_prd_start	
i2c_prd_stop	
i2c_prd_data	
i2c_fifo_config_0	
i2c_fifo_config_1	

名称	描述
i2c_fifo_wdata	
i2c_fifo_rdata	

### 11.9.1 i2c\_config

地址: 0x40018000

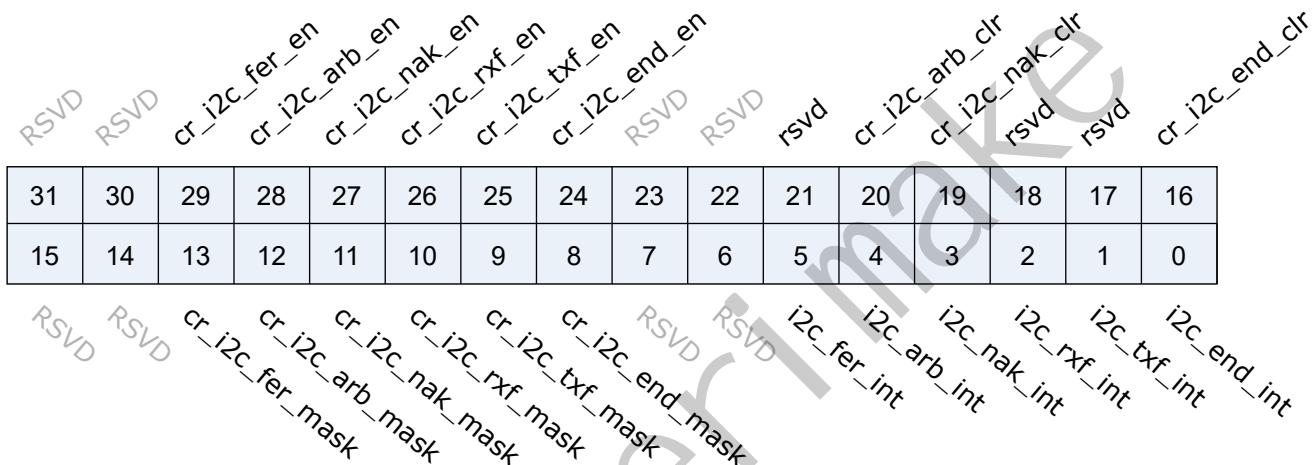


位	名称	权限	复位值	描述
31:28	cr_i2c_deg_cnt	r/w	4'd0	De-glitch function cycle count
27:20	cr_i2c_pkt_len	r/w	8'd0	Packet length (unit: byte)
19:18	RSVD			
17:8	cr_i2c_slv_addr	r/w	10'h0	Slave address for I2C transaction (target address)
7	cr_i2c_10b_addr_en	r/w	1'b0	Slave address 10-bit mode enable
6:5	cr_i2c_sub_addr_bc	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)

位	名称	权限	复位值	描述
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

地址: 0x40018004



位	名称	权限	复位值	描述
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
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	

位	名称	权限	复位值	描述
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'b1	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

地址: 0x40018008

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

位	名称	权限	复位值	描述
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]

位	名称	权限	复位值	描述
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

地址: 0x4001800c

| 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    |      |      |

位	名称	权限	复位值	描述
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

地址: 0x40018010

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							
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

位	名称	权限	复位值	描述
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

位	名称	权限	复位值	描述
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

地址: 0x40018014

cr_i2c_prd_p_ph_3								cr_i2c_prd_p_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_p_ph_1								cr_i2c_prd_p_ph_0							

位	名称	权限	复位值	描述
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

地址: 0x40018018

cr_i2c_prd_d_ph_3								cr_i2c_prd_d_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_d_ph_1								cr_i2c_prd_d_ph_0							

位	名称	权限	复位值	描述
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
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

地址: 0x40018080

位	名称	权限	复位值	描述
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

地址: 0x40018084

RSVD	<i>rx_fifo_th</i>	RSVD	<i>tx_fifo_th</i>														
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		

位	名称	权限	复位值	描述
31:25	RSVD			
24	rx_fifo_th	r/w	1'd0	RX FIFO threshold, dma_rx_req will not be asserted if rx_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

地址: 0x40018088

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

位	名称	权限	复位值	描述
31:0	i2c_fifo_wdata	w	x	

### 11.9.11 i2c\_fifo\_rdata

地址: 0x4001808c

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

位	名称	权限	复位值	描述
31:0	i2c_fifo_rdata	r	32'h0	

## 12.1 简介

脉冲宽度调制（Pulse width modulation，简称 PWM）是一种利用微处理器的数字信号对模拟电路进行控制的一种非常有效的技术，广泛应用在从测量、通信到功率控制与变换的许多领域中。

## 12.2 主要特征

- 支持 1 组 PWM 信号生成，每组包含 4 通道 PWM 信号输出，每通道可以设置为 2 路互补 PWM
- 三种时钟源可选择（总线时钟 `<bclk>`、晶振时钟 `<xtal_ck>`、慢速时钟 `<32k>`），搭配 16-bit 时钟分频器
- 每组 PWM 都可以独立设置为不同的周期
- 每通道 PWM 都有双门限值设定，可以设定不同的占空比和相位，增加脉冲弹性
- 每通道 PWM 都有独立的死区时间设定
- 每路 PWM 输出引脚都可以设定不同的有效电平
- 每路 PWM 都有独立的连接开关用来选择是否与内部计数器相连，并可设定不连接时的默认输出电平
- 刹车信号可以将 PWM 输出电平置于预先设定的状态
- 多达 9 种可用于触发 ADC 转换的触发源
- 如下事件发生时可产生中断：计数器溢出、门限值比较匹配、PWM 周期数达到设定值

## 12.3 功能描述

### 12.3.1 时钟与分频器

每个 PWM 计数器时钟来源都有三种选择，来源如下：

- A. bclk - 芯片的总线时钟
- B. XTAL - 外部晶振时钟
- C. f32k - 系统 RTC 时钟

每个计数器都有各自的 16-bit 分频器，可通过 APB 将选择到的时钟进行分频，PWM 计数器将以分频后的时钟作为计数周期单位，每经过一个计数周期进行数一的动作。

### 12.3.2 有效电平

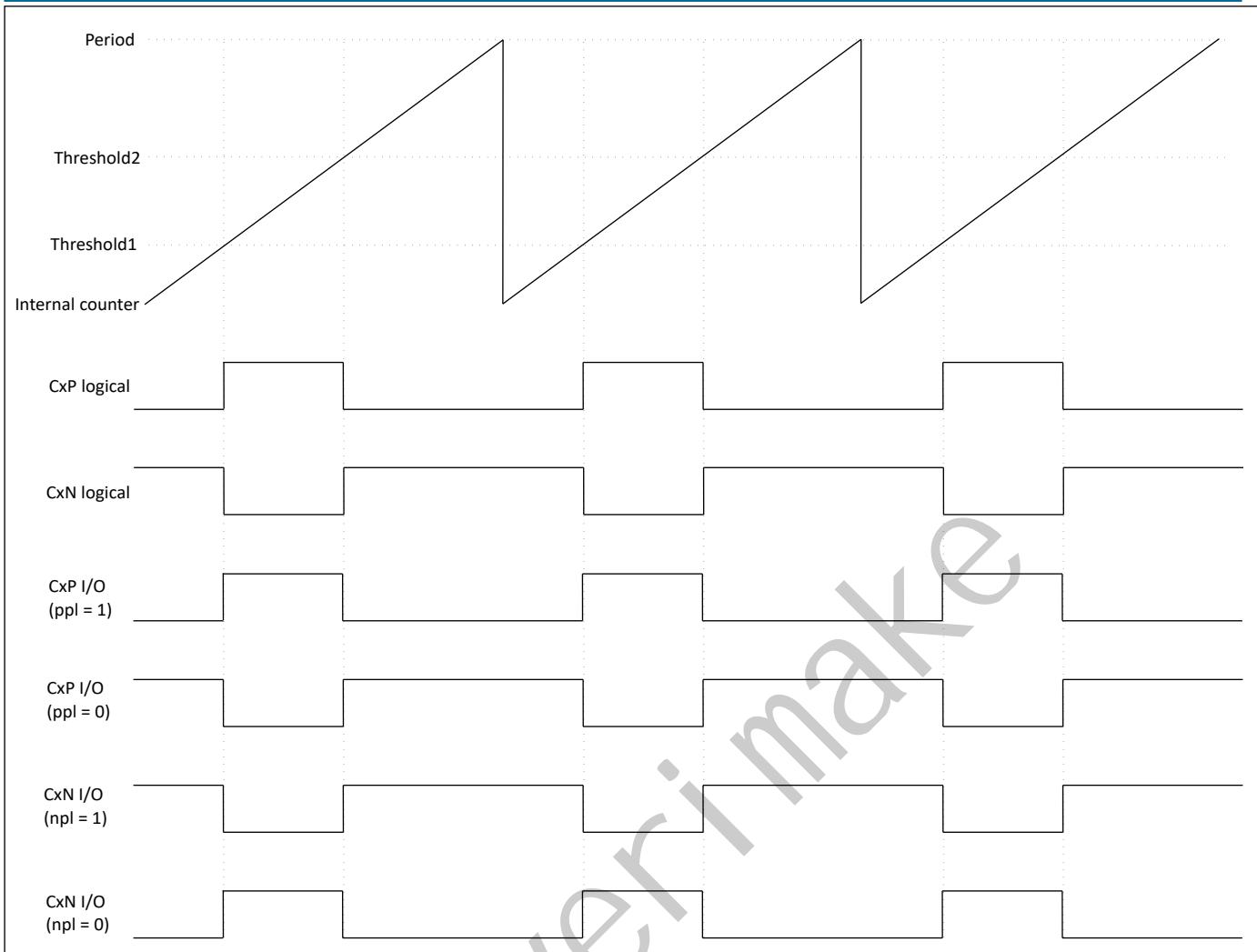
不同的应用场景所需要的有效电平状态是不一样的，一部分是低电平有效一部分是高电平有效，每路 PWM 输出的有效电平可独立设定。

当 `pwm_mcx_config1` 寄存器中的位 `<pwm_chy_ppl>` 设置为 1 时表示 `pwmx (x=0)` 的通道 `y` (`y=0,1,2,3`) 的正向通道被设置成高电平有效，即 PWM 模块输出逻辑 1 时其对应的引脚输出高电平，输出逻辑 0 时其对应的引脚输出低电平。如果 `<pwm_chy_ppl>` 设置为 0 则表示低电平有效，即 PWM 模块输出逻辑 1 时其对应的引脚输出低电平，输出逻辑 0 时其对应的引脚输出高电平。互补通道的设置位为 `<pwm_chy_npl>`，其设置规则与正向通道相同。

### 12.3.3 脉冲产生原理

当 `pwm_mcx_config1` 寄存器中的位 `<pwm_chy_pen>` 设置为 0 时表示 `pwmx (x=0)` 的通道 `y` (`y=0,1,2,3`) 的正向通道输出被设置为一个确定的逻辑状态，此时状态由位 `<pwm_chy_psi>` 确定，当 `<pwm_chy_psi>` 为 0 时对应的正向通道引脚输出逻辑 0，当 `<pwm_chy_psi>` 为 1 时对应的正向通道引脚输出逻辑 1，实际输出电平需要和 `<pwm_chy_ppl>` 共同确定。互补通道的设置位为 `<pwm_chy_nen>`，其设置规则与正向通道相同。

当 `pwm_mcx_config1` 寄存器中的位 `<pwm_chy_pen>` (`<pwm_chy_nen>`) 设置为 1 时表示 `pwmx (x=0)` 的通道 `y` (`y=0,1,2,3`) 的正向（互补）通道输出由内部计数器与两个阈值的比较结果确定，当计数器的值在两个阈值中间时，正向通道输出逻辑 1，互补通道输出逻辑 0，当计数器的值在两个阈值之外时，正向通道输出逻辑 0，互补通道输出逻辑 1。具体的波形如下图所示。



### 12.3.4 刹车

当 `pwm_mcx_config0 (x=0)` 的位 `<pwm_sw_break_en>` 为 1 时即产生刹车信号, 此时 PWM 输出的电平都会被修改为预设定的状态。预设定状态由 `pwm_mcx_config1 (x=0)` 寄存器中的位 `<pwm_chy_pbs>、<pwm_chy_nbs>` ( $y=0,1,2,3$ ) 设定, 当该位设置为 1 时, 刹车信号产生后该路 PWM 输出逻辑 1, 当该位设置为 0 时, 刹车信号产生后该路 PWM 输出逻辑 0。`<pwm_sw_break_en>` 为 0 时即退出刹车状态, PWM 恢复之前的运行方式。刹车信号不会触发中断。

### 12.3.5 死区

PWM 每个通道都可以独立设置不同的死区时间, 当死区设定值不为 0 时, PWM 的正向通道在与阈值 1 匹配时会延迟产生跳变的电平, 在阈值 2 匹配时立即改变电平状态。PWM 的互补通道在与阈值 1 匹配时会立即改变电平状态, 在阈值 2 匹配时会延迟产生跳变的电平。死区长度由 `pwm_mcx_dead_time` 寄存器设置。

### 12.3.6 周期与占空比计算

PWM 的周期由两部分决定,一个是时钟分频系数,一个是时钟持续周期。时钟分频系数由寄存器 `PWMx_CLK_DIV[15:0]`( $x$  为 0) 进行设置,用于对 PWM 的源时钟进行分频。时钟持续周期由寄存器 `PWMx_PERIOD[15:0]`( $x$  为 0) 进行设置,用于设置 PWM 的一个周期由多少个分频后的时钟周期组成。即  $\text{PWM 的周期} = \text{PWM 源时钟} / \text{PWMx_CLK_DIV}[15:0] / \text{PWMx\_PERIOD}[15:0]$ 。

### 12.3.7 PWM 中断

对于每一组 PWM,可以由 `pwm_mcx_period` 寄存器的高 16 位设置周期计数值,当 PWM 的周期数达到这个计数值时,将产生 PWM 中断。

当 PWM 计数值达到周期数或计数值与阈值匹配都将产生 PWM 中断。

### 12.3.8 ADC 联动

当计数器与门限值发生匹配或者计数值达到周期数时都可以产生内部触发 ADC 启动转换的信号,需要注意的是只有 PWM0 可以触发 ADC 启动转换, PWM1 不具备该特性。具体的触发源由 `pwm_mcx_config0` 寄存器的位 `<pwm_adc_trg_src>` 设置。这种特性在定时采样的应用场景中较为常用。以下举例说明。

应用场景: 在 BLDC 的应用中,有这样一种应用需求,在用 PWM 控制电机的转速的同时,还需要检测流过线圈的电流。在一个 PWM 周期内, PWM 控制功率器件导通后,经过一段特定的时间电流达到稳定,此时需要采样电流值,这意味着触发 ADC 转换的时刻点与 PWM 之间有严格的相位差。比如 PWM 的通道 0 用于驱动电机的其中一相,需要产生 10KHz 占空比为 20% 的方波,并且需要在方波高电平的中间时刻点执行 ADC 采样,则该 PWM 周期为 100us,当时钟源为 1MHz 时,周期计数值为 100,可以将通道 1 的两个阈值分别设置为 0 和 20,此时计数器在 0~20 之间时,PWM 输出高电平,否则输出低电平。将通道 2 的阈值 L 设置为 10,且将 `pwm_mc0_config0` 寄存器中的 `pwm_adc_trg_src` 设置为 4 即 `pwm_ch2l_int` 可以触发 ADC 转换,则计数器数到 10 时即通道 1 产生的高电平中间时刻会启动 ADC 开始采样转换,这样就可以保证每次采样都满足精确的时间要求,且不需要 CPU 干预,从而提高了性能。

## 12.4 寄存器描述

名称	描述
<code>pwm_int_config</code>	
<code>pwm_mc0_config0</code>	
<code>pwm_mc0_config1</code>	
<code>pwm_mc0_period</code>	
<code>pwm_mc0_dead_time</code>	
<code>pwm_mc0_ch0_thre</code>	
<code>pwm_mc0_ch1_thre</code>	
<code>pwm_mc0_ch2_thre</code>	



名称	描述
pwm_mc0_ch3_thre	
pwm_mc0_int_sts	
pwm_mc0_int_mask	
pwm_mc0_int_clear	
pwm_mc0_int_en	

### 12.4.1 pwm\_int\_config

地址: 0x4000a400

## 12.4.2 pwm\_mc0\_config0

地址: 0x4000a440

reg_clk_sel	pwm_sts_stop	pwm_stop_mode	pwm_stop_en	pwm_ext_break_pl	pwm_ext_break_en	pwm_sw_break_en	pwm_adc_trg_src	pwm_stop_on_rept	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
pwm_clk_div															

位	名称	权限	复位值	描述
31:30	reg_clk_sel	r/w	2'd0	PWM clock source select, 2'b00-xclk ; 2'b01-bclk ; others-f32k_clk
29	pwm_sts_stop	r	1'b0	PWM stop status
28	pwm_stop_mode	r/w	1'b1	PWM stop mode, 1'b1 - graceful ; 1'b0 - abrupt
27	pwm_stop_en	r/w	1'b0	PWM stop enable
26	pwm_ext_break_pl	r/w	1'b0	PWM external break source polarity 1'b0: Active-LOW 1'b1: Active-HIGH
25	pwm_ext_break_en	r/w	1'b0	PWM external break source enable 1'b0: Disabled, external break signal is masked 1'b1: Enabled, external break signal is effective
24	pwm_sw_break_en	r/w	1'b0	PWM break enable 1'b0: Disabled, normal operation 1'b1: Enabled, PWM output will be determined by CxPBS/CxNBS
23:20	pwm_adc_trg_src	r/w	4'hF	Select signal of ADC triggering source 4'd0: pwm_ch0l_int (Channel 0 ThresholdL reached) 4'd1: pwm_ch0h_int (Channel 0 ThresholdH reached) 4'd2: pwm_ch1l_int (Channel 1 ThresholdL reached) 4'd3: pwm_ch1h_int (Channel 1 ThresholdH reached) 4'd4: pwm_ch2l_int (Channel 2 ThresholdL reached) 4'd5: pwm_ch2h_int (Channel 2 ThresholdH reached) 4'd6: pwm_ch3l_int (Channel 3 ThresholdL reached) 4'd7: pwm_ch3h_int (Channel 3 ThresholdH reached) 4'd8: pwm_prde_int (Period End reached) Others: Disabled

位	名称	权限	复位值	描述
19	pwm_stop_on_rept	r/w	1'b0	PWM stopped when rept_int is asserted 1'b0: Disabled, PWM keeps running when rept_int is asserted 1'b1: Enabled, PWM stops when rept_int is asserted. Clear rept_int to restore operation
18:16	RSVD			
15:0	pwm_clk_div	r/w	16'b0	PWM clock division

### 12.4.3 pwm\_mc0\_config1

地址: 0x4000a444

pwm_ch3_nbs	pwm_ch3_pbs	pwm_ch2_nbs	pwm_ch2_pbs	pwm_ch1_nbs	pwm_ch1_pbs	pwm_ch0_nbs	pwm_ch0_pbs	pwm_ch3_npl	pwm_ch3_ppl	pwm_ch2_npl	pwm_ch2_ppl	pwm_ch1_npl	pwm_ch1_ppl	pwm_ch0_npl	pwm_ch0_ppl
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_ch3_nsi	pwm_ch3_nen	pwm_ch3_psi	pwm_ch3_pen	pwm_ch2_nsi	pwm_ch2_nen	pwm_ch2_psi	pwm_ch2_pen	pwm_ch1_nsi	pwm_ch1_nen	pwm_ch1_psi	pwm_ch1_pen	pwm_ch0_nsi	pwm_ch0_nen	pwm_ch0_psi	pwm_ch0_pen

位	名称	权限	复位值	描述
31	pwm_ch3_nbs	r/w	1'b0	PWM channel 3 negative break state
30	pwm_ch3_pbs	r/w	1'b0	PWM channel 3 positive break state
29	pwm_ch2_nbs	r/w	1'b0	PWM channel 2 negative break state
28	pwm_ch2_pbs	r/w	1'b0	PWM channel 2 positive break state
27	pwm_ch1_nbs	r/w	1'b0	PWM channel 1 negative break state
26	pwm_ch1_pbs	r/w	1'b0	PWM channel 1 positive break state
25	pwm_ch0_nbs	r/w	1'b0	PWM channel 0 negative break state
24	pwm_ch0_pbs	r/w	1'b0	PWM channel 0 positive break state
23	pwm_ch3_npl	r/w	1'b1	PWM channel 3 negative polarity
22	pwm_ch3_ppl	r/w	1'b1	PWM channel 3 positive polarity
21	pwm_ch2_npl	r/w	1'b1	PWM channel 2 negative polarity

位	名称	权限	复位值	描述
20	pwm_ch2_ppl	r/w	1'b1	PWM channel 2 positive polarity
19	pwm_ch1_npl	r/w	1'b1	PWM channel 1 negative polarity
18	pwm_ch1_ppl	r/w	1'b1	PWM channel 1 positive polarity
17	pwm_ch0_npl	r/w	1'b1	PWM channel 0 negative polarity
16	pwm_ch0_ppl	r/w	1'b1	PWM channel 0 positive polarity
15	pwm_ch3_nsi	r/w	1'b1	PWM channel 3 negative set idle state
14	pwm_ch3_nen	r/w	1'b0	PWM channel 3 negative enable pwm out
13	pwm_ch3_psi	r/w	1'b0	PWM channel 3 positive set idle state
12	pwm_ch3_pen	r/w	1'b0	PWM channel 3 positive enable pwm out
11	pwm_ch2_nsi	r/w	1'b1	PWM channel 2 negative set idle state
10	pwm_ch2_nen	r/w	1'b0	PWM channel 2 negative enable pwm out
9	pwm_ch2_psi	r/w	1'b0	PWM channel 2 positive set idle state
8	pwm_ch2_pen	r/w	1'b0	PWM channel 2 positive enable pwm out
7	pwm_ch1_nsi	r/w	1'b1	PWM channel 1 negative set idle state
6	pwm_ch1_nen	r/w	1'b0	PWM channel 1 negative enable pwm out
5	pwm_ch1_psi	r/w	1'b0	PWM channel 1 positive set idle state
4	pwm_ch1_pen	r/w	1'b0	PWM channel 1 positive enable pwm out
3	pwm_ch0_nsi	r/w	1'b1	PWM channel 0 negative set idle state
2	pwm_ch0_nen	r/w	1'b0	PWM channel 0 negative enable pwm out
1	pwm_ch0_psi	r/w	1'b0	PWM channel 0 positive set idle state
0	pwm_ch0_pen	r/w	1'b0	PWM channel 0 positive enable pwm out

#### 12.4.4 pwm\_mc0\_period

地址: 0x4000a448

pwm\_int\_period\_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

pwm\_period

位	名称	权限	复位值	描述
31:16	pwm_int_period_cnt	r/w	16'd0	PWM interrupt period counter threshold
15:0	pwm_period	r/w	16'd0	PWM period setting

### 12.4.5 pwm\_mc0\_dead\_time

地址: 0x4000a44c

pwm_ch3_dtg								pwm_ch2_dtg							
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_ch1_dtg								pwm_ch0_dtg							

位	名称	权限	复位值	描述
31:24	pwm_ch3_dtg	r/w	8'h0	PWM Channel 3 dead time generator (DTG) DTG[7:5]=0xx => DT = DTG[7:0]*1 (unit: divided PWM clock) DTG[7:5]=10x => DT = (64+DTG[5:0])*2 (unit: divided PWM clock) DTG[7:5]=110 => DT = (32+DTG[4:0])*8 (unit: divided PWM clock) DTG[7:5]=111 => DT = (32+DTG[4:0])*16 (unit: divided PWM clock)
23:16	pwm_ch2_dtg	r/w	8'h0	PWM Channel 2 dead time generator (DTG) DTG[7:5]=0xx => DT = DTG[7:0]*1 (unit: divided PWM clock) DTG[7:5]=10x => DT = (64+DTG[5:0])*2 (unit: divided PWM clock) DTG[7:5]=110 => DT = (32+DTG[4:0])*8 (unit: divided PWM clock) DTG[7:5]=111 => DT = (32+DTG[4:0])*16 (unit: divided PWM clock)

位	名称	权限	复位值	描述
15:8	pwm_ch1_dtg	r/w	8'h0	PWM Channel 1 dead time generator (DTG) DTG[7:5]=0xx => DT = DTG[7:0]*1 (unit: divided PWM clock) DTG[7:5]=10x => DT = (64+DTG[5:0])*2 (unit: divided PWM clock) DTG[7:5]=110 => DT = (32+DTG[4:0])*8 (unit: divided PWM clock) DTG[7:5]=111 => DT = (32+DTG[4:0])*16 (unit: divided PWM clock)
7:0	pwm_ch0_dtg	r/w	8'h0	PWM Channel 0 dead time generator (DTG) DTG[7:5]=0xx => DT = DTG[7:0]*1 (unit: divided PWM clock) DTG[7:5]=10x => DT = (64+DTG[5:0])*2 (unit: divided PWM clock) DTG[7:5]=110 => DT = (32+DTG[4:0])*8 (unit: divided PWM clock) DTG[7:5]=111 => DT = (32+DTG[4:0])*16 (unit: divided PWM clock)

#### 12.4.6 pwm\_mc0\_ch0\_thre

地址: 0x4000a450

pwm\_ch0\_threH

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\_ch0\_threL

位	名称	权限	复位值	描述
31:16	pwm_ch0_threH	r/w	16'd0	PWM HIGH counter threshold, can't be smaller than threL
15:0	pwm_ch0_threL	r/w	16'd0	PWM LOW counter threshold, can't be larger than threH

### 12.4.7 pwm\_mc0\_ch1\_thre

地址: 0x4000a454

pwm\_ch1\_threH

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\_ch1\_threL

位	名称	权限	复位值	描述
31:16	pwm_ch1_threH	r/w	16'd0	PWM HIGH counter threshold, can't be smaller than threL
15:0	pwm_ch1_threL	r/w	16'd0	PWM LOW counter threshold, can't be larger than threH

### 12.4.8 pwm\_mc0\_ch2\_thre

地址: 0x4000a458

pwm\_ch2\_threH

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\_ch2\_threL

位	名称	权限	复位值	描述
31:16	pwm_ch2_threH	r/w	16'd0	PWM HIGH counter threshold, can't be smaller than threL
15:0	pwm_ch2_threL	r/w	16'd0	PWM LOW counter threshold, can't be larger than threH

### 12.4.9 pwm\_mc0\_ch3\_thre

地址: 0x4000a45c

pwm\_ch3\_threH

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\_ch3\_threL

位	名称	权限	复位值	描述
31:16	pwm_ch3_threH	r/w	16'd0	PWM HIGH counter threshold, can't be smaller than threL
15:0	pwm_ch3_threL	r/w	16'd0	PWM LOW counter threshold, can't be larger than threH

## 12.4.10 pwm\_mc0\_int\_sts

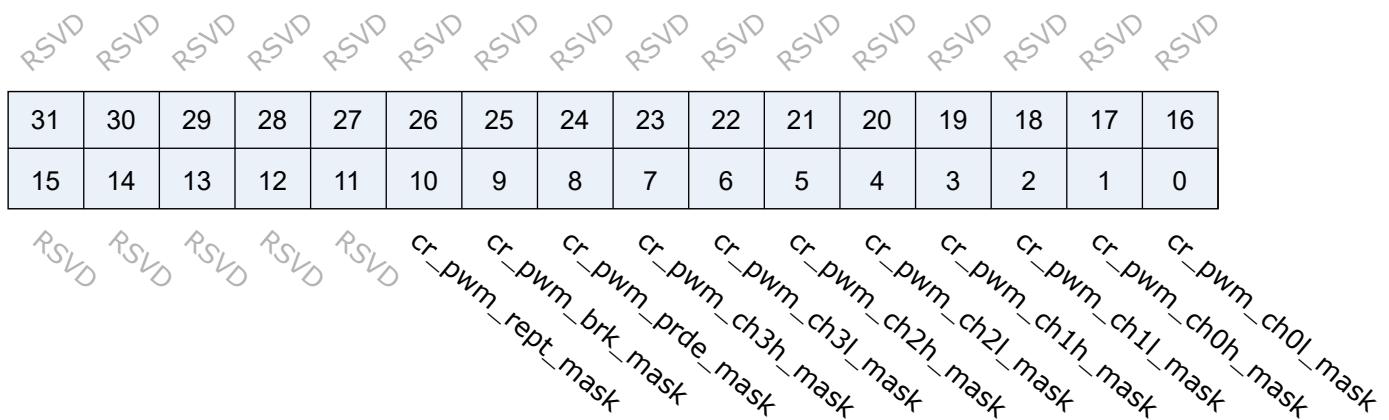
地址: 0x4000a460

| 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    |      |      |

位	名称	权限	复位值	描述
31:11	RSVD			
10	pwm_rept_int	r	1'b0	PWM repeat count interrupt status
9	pwm_brk_int	r	1'b0	PWM break interrupt status, triggered when ext_break is asserted Note: pwm_sw_break_en will NOT trigger this interrupt
8	pwm_prde_int	r	1'b0	PWM period end interrupt status
7	pwm_ch3h_int	r	1'b0	PWM Channel 3 ThresholdH interrupt status
6	pwm_ch3l_int	r	1'b0	PWM Channel 3 ThresholdL interrupt status
5	pwm_ch2h_int	r	1'b0	PWM Channel 2 ThresholdH interrupt status
4	pwm_ch2l_int	r	1'b0	PWM Channel 2 ThresholdL interrupt status
3	pwm_ch1h_int	r	1'b0	PWM Channel 1 ThresholdH interrupt status
2	pwm_ch1l_int	r	1'b0	PWM Channel 1 ThresholdL interrupt status
1	pwm_ch0h_int	r	1'b0	PWM Channel 0 ThresholdH interrupt status
0	pwm_ch0l_int	r	1'b0	PWM Channel 0 ThresholdL interrupt status

### 12.4.11 pwm\_mc0\_int\_mask

地址: 0x4000a464



位	名称	权限	复位值	描述
31:11	RSVD			
10	cr_pwm_rept_mask	r/w	1'b1	Interrupt mask of pwm_rept_int
9	cr_pwm_brk_mask	r/w	1'b1	Interrupt mask of pwm_brk_int
8	cr_pwm_prde_mask	r/w	1'b1	Interrupt mask of pwm_prde_int
7	cr_pwm_ch3h_mask	r/w	1'b1	Interrupt mask of pwm_ch3h_int
6	cr_pwm_ch3l_mask	r/w	1'b1	Interrupt mask of pwm_ch3l_int
5	cr_pwm_ch2h_mask	r/w	1'b1	Interrupt mask of pwm_ch2h_int
4	cr_pwm_ch2l_mask	r/w	1'b1	Interrupt mask of pwm_ch2l_int
3	cr_pwm_ch1h_mask	r/w	1'b1	Interrupt mask of pwm_ch1h_int
2	cr_pwm_ch1l_mask	r/w	1'b1	Interrupt mask of pwm_ch1l_int
1	cr_pwm_ch0h_mask	r/w	1'b1	Interrupt mask of pwm_ch0h_int
0	cr_pwm_ch0l_mask	r/w	1'b1	Interrupt mask of pwm_ch0l_int

### 12.4.12 pwm\_mc0\_int\_clear

地址: 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

cr\_pwm\_rept\_clr    cr\_pwm\_brk\_clr    cr\_pwm\_prde\_clr    cr\_pwm\_ch3h\_clr    cr\_pwm\_ch3l\_clr    cr\_pwm\_ch2h\_clr    cr\_pwm\_ch2l\_clr    cr\_pwm\_ch1h\_clr    cr\_pwm\_ch1l\_clr    cr\_pwm\_ch0h\_clr    cr\_pwm\_ch0l\_clr

位	名称	权限	复位值	描述
31:11	RSVD			
10	cr_pwm_rept_clr	w1c	1'b0	Interrupt clear of pwm_rept_int
9	cr_pwm_brk_clr	w1c	1'b0	Interrupt clear of pwm_brk_int
8	cr_pwm_prde_clr	w1c	1'b0	Interrupt clear of pwm_prde_int
7	cr_pwm_ch3h_clr	w1c	1'b0	Interrupt clear of pwm_ch3h_int
6	cr_pwm_ch3l_clr	w1c	1'b0	Interrupt clear of pwm_ch3l_int
5	cr_pwm_ch2h_clr	w1c	1'b0	Interrupt clear of pwm_ch2h_int
4	cr_pwm_ch2l_clr	w1c	1'b0	Interrupt clear of pwm_ch2l_int
3	cr_pwm_ch1h_clr	w1c	1'b0	Interrupt clear of pwm_ch1h_int
2	cr_pwm_ch1l_clr	w1c	1'b0	Interrupt clear of pwm_ch1l_int
1	cr_pwm_ch0h_clr	w1c	1'b0	Interrupt clear of pwm_ch0h_int
0	cr_pwm_ch0l_clr	w1c	1'b0	Interrupt clear of pwm_ch0l_int

### 12.4.13 pwm\_mc0\_int\_en

地址: 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

cr\_pwm\_rept\_en    cr\_pwm\_brk\_en    cr\_pwm\_prde\_en    cr\_pwm\_ch3h\_en    cr\_pwm\_ch3l\_en    cr\_pwm\_ch2h\_en    cr\_pwm\_ch2l\_en    cr\_pwm\_ch1h\_en    cr\_pwm\_ch1l\_en    cr\_pwm\_ch0h\_en    cr\_pwm\_ch0l\_en

位	名称	权限	复位值	描述
31:11	RSVD			
10	cr_pwm_rept_en	r/w	1'b1	Interrupt enable of pwm_rept_int
9	cr_pwm_brk_en	r/w	1'b1	Interrupt enable of pwm_brk_int
8	cr_pwm_prde_en	r/w	1'b1	Interrupt enable of pwm_prde_int
7	cr_pwm_ch3h_en	r/w	1'b1	Interrupt enable of pwm_ch3h_int
6	cr_pwm_ch3l_en	r/w	1'b1	Interrupt enable of pwm_ch3l_int
5	cr_pwm_ch2h_en	r/w	1'b1	Interrupt enable of pwm_ch2h_int
4	cr_pwm_ch2l_en	r/w	1'b1	Interrupt enable of pwm_ch2l_int
3	cr_pwm_ch1h_en	r/w	1'b1	Interrupt enable of pwm_ch1h_int
2	cr_pwm_ch1l_en	r/w	1'b1	Interrupt enable of pwm_ch1l_int
1	cr_pwm_ch0h_en	r/w	1'b1	Interrupt enable of pwm_ch0h_int
0	cr_pwm_ch0l_en	r/w	1'b1	Interrupt enable of pwm_ch0l_int

## 13.1 简介

芯片内置两个 32-bit 定时器，定时器可独立控制配置其参数与时钟频率。

芯片内有一个看门狗定时器，不可预知的软件或硬件行为有可能导致应用程序工作失常，看门狗定时器可以帮助系统从中恢复，如果当前阶段超过预定时间，但没有喂狗或关闭看门狗定时器，可依设定触发中断或系统复位。

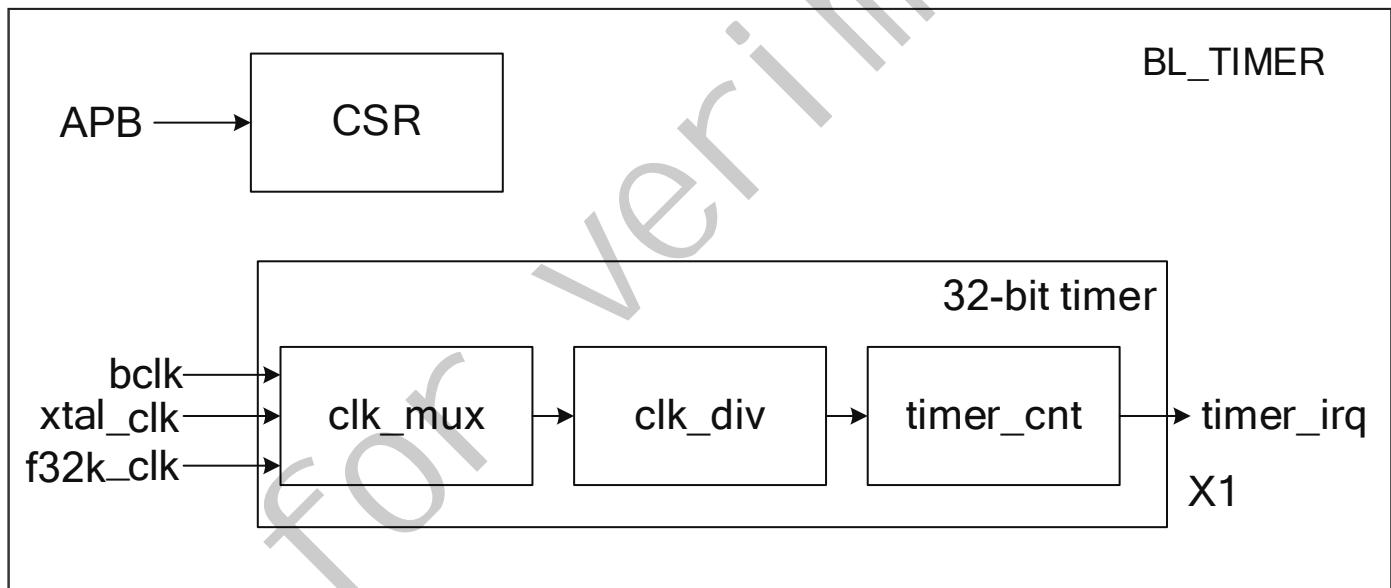


图 13.1: 定时器框图

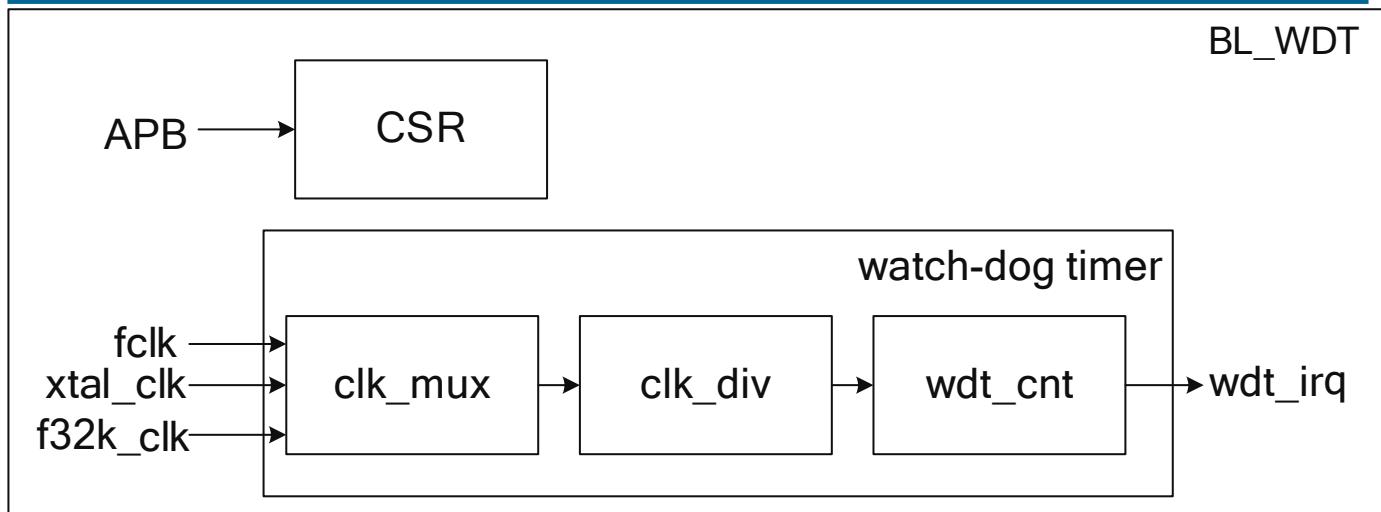


图 13.2: 看门狗定时器框图

## 13.2 主要特征

- 多种时钟来源，最高可支持 80M 时钟
- 8-bit 时钟分频器，分频系数为 1-256
- 两个 32-bit 定时器：channel 0 和 channel 1
- 定时器包含三组报警值设定，可设定报警值溢出时报警
- 支持 Free Run 模式和 Pre\_load 模式
- 一个 16-bit 看门狗定时器
- 支持写入密码保护，防止误设定造成系统异常
- 支持中断或复位两种看门狗溢出方式
- 支持测量外部 GPIO 的脉冲宽度

## 13.3 功能描述

Watchdog 定时器时钟有 5 种选择：

- BCLK--总线时钟
- 32K--32K 时钟
- 1K--1K 时钟
- XTAL--外部晶振
- GPIO--外部 GPIO

通过寄存器 TCCR 中的 `cs_wdt` 配置。

定时器时钟来源都有 5 种选择，来源如下：

- BCLK--总线时钟
- 32K--32K 时钟
- 1K--1K 时钟（32K 的分频）
- XTAL--外部晶振
- GPIO--外部 GPIO

通过寄存器 TCCR 中的 `cs_2` 和 `cs_3` 配置。

定时器有各自的 8-bit 分频器，可对其选择的时钟源进行 1-256 的分频，具体来说设定为 0 时表示不分频，设定为 1 时进行 2 分频以此类推，最大分频系数为 256，定时器将以分频后的时钟作为计数周期单位。

通过寄存器 TCDR 中的 `tcdr2`、`tcdr3`、`wcdr` 配置。

### 13.3.1 通用定时器工作原理

通用定时器都包含三组比较器，一个计数器以及一个预加载寄存器，当设定好时钟源，启动定时器后，计数器开始向上累加计数，当计数器的值与比较器相等的时候，比较标志置位同时可以产生比较中断。

配置寄存器 TICR2 中的 `tclr2_0` 设置 channel 0 比较器 0 的值，配置寄存器 TICR2 中的 `tclr2_1` 设置 channel 0 比较器 1 的值，配置寄存器 TICR2 中的 `tclr2_2` 设置 channel 0 比较器 2 的值。配置寄存器 TPLVR2 中的 `tplvr2` 设置 channel 0 预加载值。

配置寄存器 TICR3 中的 `tclr3_0` 设置 channel 1 比较器 0 的值，配置寄存器 TICR3 中的 `tclr3_1` 设置 channel 1 比较器 1 的值，配置寄存器 TICR3 中的 `tclr3_2` 设置 channel 1 比较器 2 的值。配置寄存器 TPLVR3 中的 `tplvr3` 设置 channel 1 预加载值。

计数器的初始值取决于定时的模式，在 FreeRun 模式下，计数器的初始值是 0，然后累加计数，当达到计数最大值后，然后从 0 再次开始计数。

在 PreLoad 模式下，计数器的初始值是 PreLoad 寄存器的值，然后向上累加计数，当满足 PreLoad 条件时，计数器的值被置为 PreLoad 寄存器的值，然后计数器再次开始向上累加计数，在定时器的计数过程中，一旦计数器的值与三个比较器中的某比较值一致，该比较器的比较标志就会置位，并可以产生相应的比较中断。

配置寄存器 TCMR 中的 `timer2_mode` 设置 channel 0 的计数模式，配置寄存器 TCMR 中的 `timer3_mode` 设置 channel 1 的计数模式。

若预加载寄存器的值为 10，比较器 0 的值为 13，比较器 1 的值为 16，比较器 2 的值为 19，则定时器在 PreLoad 的模式下工作时序如下图：

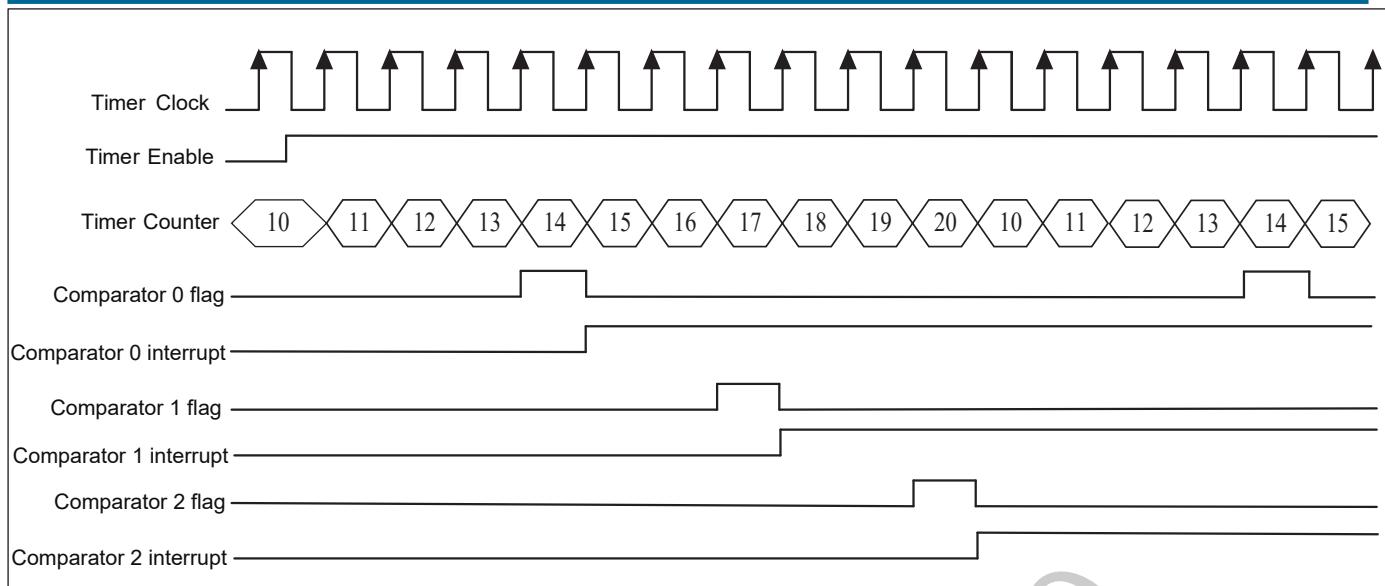


图 13.3: 定时器在 PreLoad 模式下工作时序

在 FreeRun 模式下，定时器工作时序与 PreLoad 基本相同，只是计数器会从 0 开始累计到最大值，期间产生的比较标志和比较中断的机制与 FreeRun 模式相同。

channel 0 可以使用内部的时钟源计算外部 gpio 的脉冲宽度。通过设置寄存器 GPIO 中的 `timer2_gpio_en` 开启该功能，通过设置 `timer2_gpio_inv` 位，判断获取的是外部 gpio 的高电平还是低电平的宽度，如果该位为 0，表示高电平；如果该位为 1，表示低电平；另外需要将外部的 gpio 功能设置为 `gpio_tmr_clk` 功能。通过配置 GLB 模块中寄存器 `dig_clk_cfg2` 中的 `gpio_tmr_clk_sel[13:12]` 位；同时需要将寄存器 `dig_clk_cfg2[11:8]` 中的某一位配置为 0，需要和 `gpio_tmr_clk_sel` 配套使用。具体如下：

- 若 `gpio_tmr_clk_sel[13:12]` 配置为 0，则寄存器 `dig_clk_cfg2` 中的 `chip_clk_out_0_en` 设置为 0
- 若 `gpio_tmr_clk_sel[13:12]` 配置为 1，则寄存器 `dig_clk_cfg2` 中的 `chip_clk_out_1_en` 设置为 0
- 若 `gpio_tmr_clk_sel[13:12]` 配置为 2，则寄存器 `dig_clk_cfg2` 中的 `chip_clk_out_2_en` 设置为 0
- 若 `gpio_tmr_clk_sel[13:12]` 配置为 3，则寄存器 `dig_clk_cfg2` 中的 `chip_clk_out_3_en` 设置为 0

配置完成后，使能 timer。当寄存器 GPIO 中的 `gpio_lat_ok` 置 1 后，获取寄存器 `GPIO_LAT2` 和寄存器 `GPIO_LAT1` 的值。外部 gpio 的脉冲宽度的计算方式： $(\text{GPIO\_LAT2} - \text{GPIO\_LAT1}) * \text{timer}$  内部时钟源的 1 个周期的宽度；

例如：timer 的内部时钟源为 80M，外部 gpio 的频率为 2M，且占空比为 1: 1，将 `timer2_gpio_inv` 位写 1，表示计算外部 gpio 低电平的宽度。按照上述配置完成后，得到寄存器 `GPIO_LAT2` 和寄存器 `GPIO_LAT1` 的差为 20，则外部 gpio 的低电平宽度为： $20 * (1 / 80000000) = 1 / 4000000$ ；将 `timer2_gpio_inv` 位写 0，表示计算外部 gpio 高电平的宽度。按照上述配置完成后，得到寄存器 `GPIO_LAT2` 和寄存器 `GPIO_LAT1` 的差为 20，则外部 gpio 的高电平宽度为： $20 * (1 / 80000000) = 1 / 4000000$ ；

### 13.3.2 看门狗定时器工作原理

Watchdog 定时器包含一个计数器和一个比较器，计数器从 0 开始累加计数，如果计数器被复位（喂狗），则从 0 再次开始向上计数，当计数器的值与比较器相等的时候，可以产生一个比较中断信号或者系统复位信号，用户可以根据需要选择使用其中一个。看门狗计数器会在每个计数周期单位上加 1，软件可以在任何时间点通过 APB 将看门狗计数器归零。

配置寄存器 WMR 中的 wmr 设置比较值，

若比较器的值为 6，Watchdog 的工作时序如下图所示

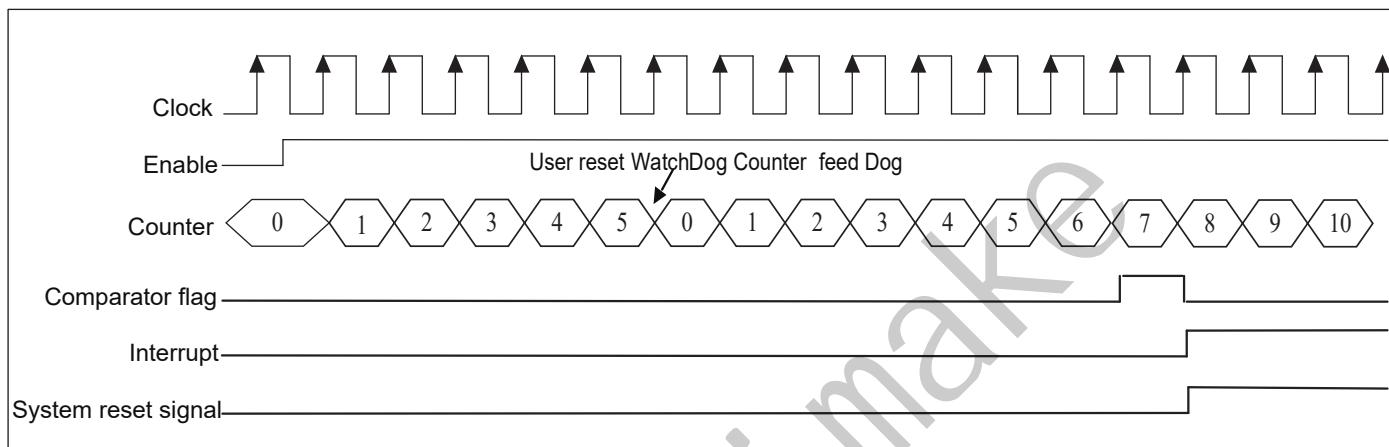


图 13.4: Watchdog 工作时序

### 13.3.3 报警设定

计数器有三个比较值提供软件设定，并可设定比较值是否触发报警中断，当计数器与比较值吻合且设定会报警时，计数器会通过中断通知处理器。软件可以通过 APB 读取目前是否发生报警和是哪个比较值触发报警中断，当清理报警中断时亦会同步清理报警状态。

### 13.3.4 看门狗报警

看门狗可设定一组比较值，当软件因为系统错误，来不及将看门狗计数器归零，导致看门狗计数器超过比较值时，便会触发看门狗报警，报警方式有两种，第一种是通过中断通知软件进行必要的处置，第二种是进入系统看门狗复位，看门狗复位被触发时，会通知系统复位控制器，并做好系统复位前准备，当一切就绪后进入系统看门狗复位，值得注意的是，软件可通过 APB 读取 WSR 寄存器得知是否曾经发生过看门狗系统复位。

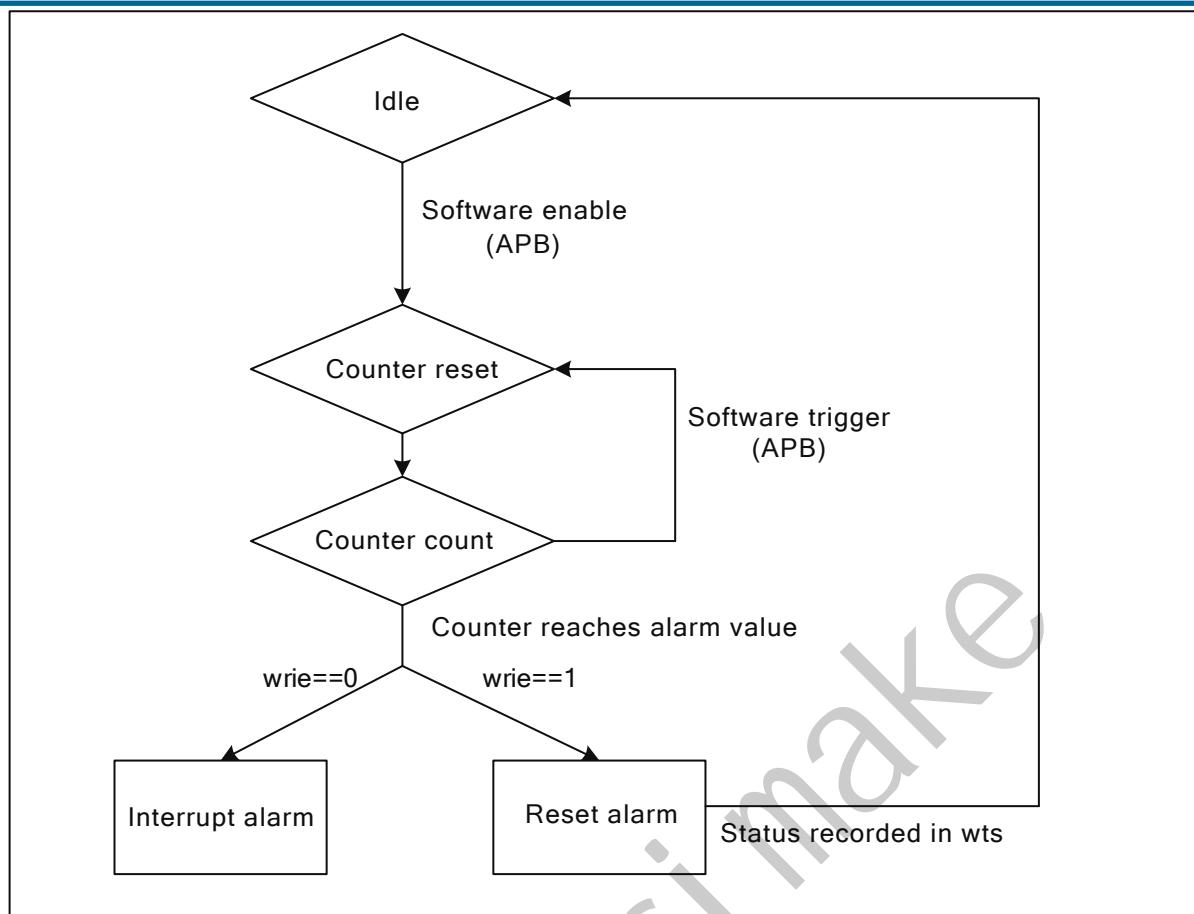


图 13.5: 看门狗报警机制

## 13.4 寄存器描述

名称	描述
TCCR	Timer Clock Source
TMR2_0	Timer2 Match Value 0
TMR2_1	Timer2 Match Value 1
TMR2_2	Timer2 Match Value 2
TMR3_0	Timer3 Match Value 0
TMR3_1	Timer3 Match Value 1
TMR3_2	Timer3 Match Value 2
TCR2	Timer2 Counter Value
TCR3	Timer3 Counter Value
TSR2	Timer2 Match Status

名称	描述
TSR3	Timer3 Match Status
TIER2	Timer2 Match Interrupt Enable
TIER3	Timer3 Match Interrupt Enable
TPLVR2	Timer2 Pre-Load Value
TPLVR3	Timer3 Pre-Load Value
TPLCR2	Timer2 Pre-Load Control
TPLCR3	Timer3 Pre-Load Control
WMER	Watch-dog reset/interrupt Mode
WMR	Watch-dog Match Value
WVR	Watch-dog Counter Value
WSR	Watch-dog Reset Status
TICR2	Timer2 Interrupt Clear
TICR3	Timer3 Interrupt Clear
WICR	WDT Interrupt Clear
TCER	Timer Counter Enable/Clear
TCMR	Timer Counter Mode
TILR2	Timer2 Match Interrupt Mode
TILR3	Timer3 Match Interrupt Mode
WCR	WDT Counter Reset
WFAR	WDT Access Key1
WSAR	WDT Access Key2
TCVWR2	Timer2 Counter Latch Value
TCVWR3	Timer3 Counter Latch Value
TCVSYN2	Timer2 Counter Sync Value
TCVSYN3	Timer3 Counter Sync Value
TCDR	Timer Division
GPIO	GPIO Mode
GPIO_LAT1	GPIO Latch Value1
GPIO_LAT2	GPIO Latch Value2

### 13.4.1 TCCR

地址: 0x4000a500

ID								tmr_rsv							
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 cs\_wdt cs\_3 cs\_2

位	名称	权限	复位值	描述
31:24	ID	r	8'ha5	
23:16	tmr_rsv	rsvd	0	
15:12	RSVD			
11:8	cs_wdt	r/w	4'd1	WDT 0:fclk / 1:f32k / 2:1k / 3:32M / 4:GPIO / 5:No clock
7:4	cs_3	r/w	4'd5	Timer3 0:fclk / 1:f32k / 2:1k / 3:32M / 4:GPIO / 5:No clock
3:0	cs_2	r/w	4'd5	Timer2 0:fclk / 1:f32k / 2:1k / 3:32M / 4:GPIO / 5:No clock

### 13.4.2 TMR2\_0

地址: 0x4000a510

tmr2_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
tmr2_0															

位	名称	权限	复位值	描述
31:0	tmr2_0	r/w	32'hffffffff	Timer2 Match Value 0

### 13.4.3 TMR2\_1

地址: 0x4000a514

tmr2\_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

tmr2\_1

位	名称	权限	复位值	描述
31:0	tmr2_1	r/w	32'hffffffff	Timer2 Match Value 1

### 13.4.4 TMR2\_2

地址: 0x4000a518

tmr2\_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

tmr2\_2

位	名称	权限	复位值	描述
31:0	tmr2_2	r/w	32'hffffffff	Timer2 Match Value 2

### 13.4.5 TMR3\_0

地址: 0x4000a51c

tmr3\_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

tmr3\_0

位	名称	权限	复位值	描述
31:0	tmr3_0	r/w	32'hffffffff	Timer3 Match Value 0

### 13.4.6 TMR3\_1

地址: 0x4000a520

tmr3\_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

tmr3\_1

位	名称	权限	复位值	描述
31:0	tmr3_1	r/w	32'hffffffff	Timer3 Match Value 1

### 13.4.7 TMR3\_2

地址: 0x4000a524

tmr3\_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

tmr3\_2

位	名称	权限	复位值	描述
31:0	tmr3_2	r/w	32'hffffffff	Timer3 Match Value 2

### 13.4.8 TCR2

地址: 0x4000a52c

tcr2\_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

tcr2\_cnt

位	名称	权限	复位值	描述
31:0	tcr2_cnt	r	0	Timer2 Counter Value

### 13.4.9 TCR3

地址: 0x4000a530

tcr3\_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

tcr3\_cnt

位	名称	权限	复位值	描述
31:0	tcr3_cnt	r	0	Timer3 Counter Value

### 13.4.10 TSR2

地址: 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	

位	名称	权限	复位值	描述
31:3	RSVD			
2	tsr2_2	r	0	Timer2 match value 2 status/Clear interrupt would also clear this bit
1	tsr2_1	r	0	Timer2 match value 1 status/Clear interrupt would also clear this bit
0	tsr2_0	r	0	Timer2 match value 0 status/Clear interrupt would also clear this bit

### 13.4.11 TSR3

地址: 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 tsr3\_2 tsr3\_1 tsr3\_0

位	名称	权限	复位值	描述
31:3	RSVD			
2	tsr3_2	r	0	Timer3 match value 2 status/Clear interrupt would also clear this bit
1	tsr3_1	r	0	Timer3 match value 1 status/Clear interrupt would also clear this bit
0	tsr3_0	r	0	Timer3 match value 0 status/Clear interrupt would also clear this bit

### 13.4.12 TIER2

地址: 0x4000a544

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 tier2\_2 tier2\_1 tier2\_0

位	名称	权限	复位值	描述
31:3	RSVD			
2	tier2_2	r/w	0	Timer2 match value 2 interrupt enable
1	tier2_1	r/w	0	Timer2 match value 1 interrupt enable
0	tier2_0	r/w	0	Timer2 match value 0 interrupt enable

### 13.4.13 TIER3

地址: 0x4000a548

| 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 tier3\_2 tier3\_1 tier3\_0

位	名称	权限	复位值	描述
31:3	RSVD			
2	tier3_2	r/w	0	Timer3 match value 2 interrupt enable
1	tier3_1	r/w	0	Timer3 match value 1 interrupt enable
0	tier3_0	r/w	0	Timer3 match value 0 interrupt enable

### 13.4.14 TPLVR2

地址: 0x4000a550

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

位	名称	权限	复位值	描述
31:0	tplvr2	r/w	0	Timer2 Pre-Load Value

### 13.4.15 TPLVR3

地址: 0x4000a554

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

位	名称	权限	复位值	描述
31:0	tplvr3	r/w	0	Timer3 Pre-Load Value

### 13.4.16 TPLCR2

地址: 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    |      |      |

位	名称	权限	复位值	描述
31:2	RSVD			
1:0	tplcr2	r/w	0	Timer2 pre-load control 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

地址: 0x4000a560

位	名称	权限	复位值	描述
31:2	RSVD			

位	名称	权限	复位值	描述
1:0	tplcr3	r/w	0	Timer3 pre-load control 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

地址: 0x4000a564

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	wrie	we

位	名称	权限	复位值	描述
31:2	RSVD			
1	wrie	r/w	0	WDT reset/interrupt mode 1'b0 - WDT expiration to generate interrupt 1'b1 - WDT expiration to generate reset source
0	we	r/w	0	WDT enable register

### 13.4.19 WMR

地址: 0x4000a568

RSVD	wdt_align																
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	

位	名称	权限	复位值	描述
31:17	RSVD			
16	wdt_align	r/w	0	WDT compare value update align interrupt
15:0	wmr	r/w	16'hffff	WDT counter match value

### 13.4.20 WVR

地址: 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    |      |      |      |      |

wdt\_cnt

位	名称	权限	复位值	描述
31:16	RSVD			
15:0	wdt_cnt	r	0	WDT counter value

### 13.4.21 WSR

地址: 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    |      |      |      |      |

位	名称	权限	复位值	描述
31:1	RSVD			
0	wts	w	0	WDT reset status Write 0 to clear the WDT reset status Read 1 indicates reset was caused by the WDT

### 13.4.22 TICR2

地址: 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

位	名称	权限	复位值	描述
31:3	RSVD			
2	tclr2_2	w	0	Timer2 Interrupt clear for match comparator 2
1	tclr2_1	w	0	Timer2 Interrupt clear for match comparator 1
0	tclr2_0	w	0	Timer2 Interrupt clear for match comparator 0

### 13.4.23 TICR3

地址: 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

位	名称	权限	复位值	描述
31:3	RSVD			
2	tclr3_2	w	0	Timer3 Interrupt clear for match comparator 2
1	tclr3_1	w	0	Timer3 Interrupt clear for match comparator 1
0	tclr3_0	w	0	Timer3 Interrupt clear for match comparator 0

### 13.4.24 WICR

地址: 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	wiclr

位	名称	权限	复位值	描述
31:1	RSVD			
0	wiclr	w	0	WDT Interrupt Clear

### 13.4.25 TCER

地址: 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	tcr3_cnt_clr

位	名称	权限	复位值	描述
31:7	RSVD			
6	tcr3_cnt_clr	r/w	0	Timer3 count clear
5	tcr2_cnt_clr	r/w	0	Timer2 count clear
4:3	RSVD			
2	timer3_en	r/w	0	Timer3 count enable
1	timer2_en	r/w	0	Timer2 count enable
0	RSVD			

### 13.4.26 TCMR

地址: 0x4000a588

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	RSVD	RSVD	RSVD	RSVD	RSVD										
										timer3_align	timer2_align	RSVD	RSVD	timer2_mode	RSVD

位	名称	权限	复位值	描述
31:7	RSVD			
6	timer3_align	r/w	0	Timer3 compare value update align interrupt
5	timer2_align	r/w	0	Timer2 compare value update align interrupt
4:3	RSVD			
2	timer3_mode	r/w	0	0:pre-load mode 1:free run mode
1	timer2_mode	r/w	0	0:pre-load mode 1:free run mode
0	RSVD			

### 13.4.27 TILR2

地址: 0x4000a590

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	tilr2_2	tilr2_1													
														tilr2_0	

位	名称	权限	复位值	描述
31:3	RSVD			
2	tilr2_2	r/w	0	0:level 1:edge
1	tilr2_1	r/w	0	0:level 1:edge

位	名称	权限	复位值	描述
0	tilr2_0	r/w	0	0:level 1:edge

### 13.4.28 TILR3

地址: 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	

位	名称	权限	复位值	描述
31:3	RSVD			
2	tilr3_2	r/w	0	0:level 1:edge
1	tilr3_1	r/w	0	0:level 1:edge
0	tilr3_0	r/w	0	0:level 1:edge

### 13.4.29 WCR

地址: 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	

位	名称	权限	复位值	描述
31:1	RSVD			
0	wcr	w	0	WDT Counter Reset

### 13.4.30 WFAR

地址: 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

位	名称	权限	复位值	描述
31:16	RSVD			
15:0	wfar	w	0	WDT access key1 - 16'hBABA

### 13.4.31 WSAR

地址: 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

位	名称	权限	复位值	描述
31:16	RSVD			
15:0	wsar	w	0	WDT access key2 - 16'hEB10

### 13.4.32 TCVWR2

地址: 0x4000a5a8

tcr2\_cnt\_lat

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\_cnt\_lat

位	名称	权限	复位值	描述
31:0	tcr2_cnt_lat	r	0	Timer2 Counter Latch Value

### 13.4.33 TCVWR3

地址: 0x4000a5ac

tcr3\_cnt\_lat

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\_cnt\_lat

位	名称	权限	复位值	描述
31:0	tcr3_cnt_lat	r	0	Timer3 Counter Latch Value

### 13.4.34 TCVSYN2

地址: 0x4000a5b4

tcr2\_cnt\_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

tcr2\_cnt\_sync

位	名称	权限	复位值	描述
31:0	tcr2_cnt_sync	r	0	Timer2 Counter Sync Value (continue readable)

### 13.4.35 TCVSYN3

地址: 0x4000a5b8

tcr3\_cnt\_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

tcr3\_cnt\_sync

位	名称	权限	复位值	描述
31:0	tcr3_cnt_sync	r	0	Timer3 Counter Sync Value (continue readable)

### 13.4.36 TCDR

地址: 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

位	名称	权限	复位值	描述
31:24	wcdr	r/w	0	WDT clock division value register
23:16	tcdr3	r/w	0	Timer3 clock division value register
15:8	tcdr2	r/w	0	Timer2 clock division value register
7:0	RSVD			

### 13.4.37 GPIO

地址: 0x4000a5c0

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	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD
gpio_lat_ok	RSVD														

位	名称	权限	复位值	描述
31	gpio_lat_ok	r	0	Latch Done. Pulse width = (GPIO_LAT2 - GPIO_LAT1) * (Timer2 Cycle)
30:8	RSVD			
7	wdt_gpio_inv	r/w	0	WDT gpio polarity 0:pos 1:neg
6	timer3_gpio_inv	r/w	0	Timer3 gpio polarity 0:pos 1:neg
5	timer2_gpio_inv	r/w	0	Timer2 gpio polarity 0:pos 1:neg
4:2	RSVD			
1	timer2_gpio_en	r/w	0	Timer2 gpio measure enable
0	RSVD			

### 13.4.38 GPIO\_LAT1

地址: 0x4000a5c4

gpio\_lat1

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\_lat1

位	名称	权限	复位值	描述
31:0	gpio_lat1	r	0	Pos-Edge Latch Timer2

### 13.4.39 GPIO\_LAT2

地址: 0x4000a5c8

gpio\_lat2

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\_lat2

位	名称	权限	复位值	描述
31:0	gpio_lat2	r	0	Neg-Edge Latch Timer2

## 14.1 简介

I2S 全称 Inter-IC Sound, Integrated Interchip Sound, 或简写 IIS, 是飞利浦在 1986 年定义（1996 年修订）的数字音频传输标准，用于数字音频数据在系统内部器件之间传输。I2S 是比较简单的数字接口协议，没有地址或设备选择机制。在 I2S 总线上，只能同时存在一个主设备和发送设备。主设备可以是发送设备，也可以是接收设备，或是协调发送设备和接收设备的其它控制设备。在 I2S 系统中，提供时钟（BCLK 和 FS）的设备为主设备。I2S 将时钟信号与数据信号分开传输，使得接收端不用从数据信号还原时钟，进而降低接收端的设计难度。

## 14.2 主要特征

- 支持主模式以及从模式
- 支持的 I2S 协议
  - Normal I2S 格式 (飞利浦格式)
  - Left-Justified 格式
  - Right-Justified 格式
  - PCM 格式
  - TDM/TDM64 格式
- 支持每通道 8/16/24/32 比特数据宽度
- 支持每帧 16/32/48/64 比特数据宽度
  - I<sup>2</sup>S 格式时每个通道 FS 位宽必须大于每个通道的数据位宽
  - PCM/TDM 格式数据位宽将会更为灵活
- 除单声道/双声道模式之外，同时还支持四声道与六声道模式（TDM 模式）
- 支持数据 MSB/LSB 切换

- 支持 DMA 传输模式
- 支持 8/11.025/16/22.05/32/44.1/48/96/192 KHz 采样率
- 支持播放单声道音频复制为双声道模式
- 支持低于 16bits 双声道录音数据合并为低于 32bits FIFO 宽度数据
- 支持动态静音切换功能
- 支持全双工或半双工模式
- 支持主时钟输出给外部音频设备
- 支持输出信号极性翻转
- 数据发送 FIFO 的宽度为 32 位，深度 16
- 数据接收 FIFO 的宽度为 32 位，深度 16

### 14.3 功能描述

I2S 模块基本框图如图所示。

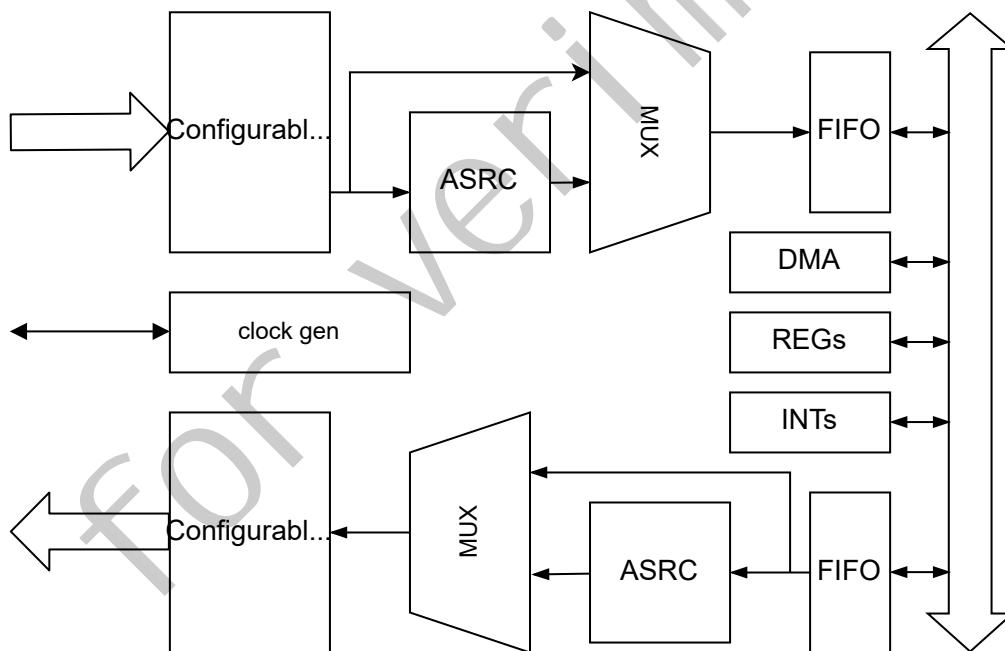


图 14.1: I2S 基本框图

引脚列表:

表 14.1: I2S 引脚

名称	类型	描述
I2S_DI	输入	串行数据输入
I2S_DO	输出	串行数据输出
I2S_BCLK	输入/输出	同步传输时钟, 作为主机时为输出, 作为从机时为输入
I2S_FS	输入/输出	数据起始/结束表示信号, 作为主机时为输出, 作为从机时为输入
I2S_RCLK_O	输出	主时钟输出信号

## 14.4 功能描述

### 14.4.1 数据格式描述

I2S 模块支持标准 I2S 协议, 左对齐模式, 右对齐模式。标准 I2S 是左对齐模式的特殊情况。左右对齐模式由 I2S\_CONFIG[17:16] 配置。

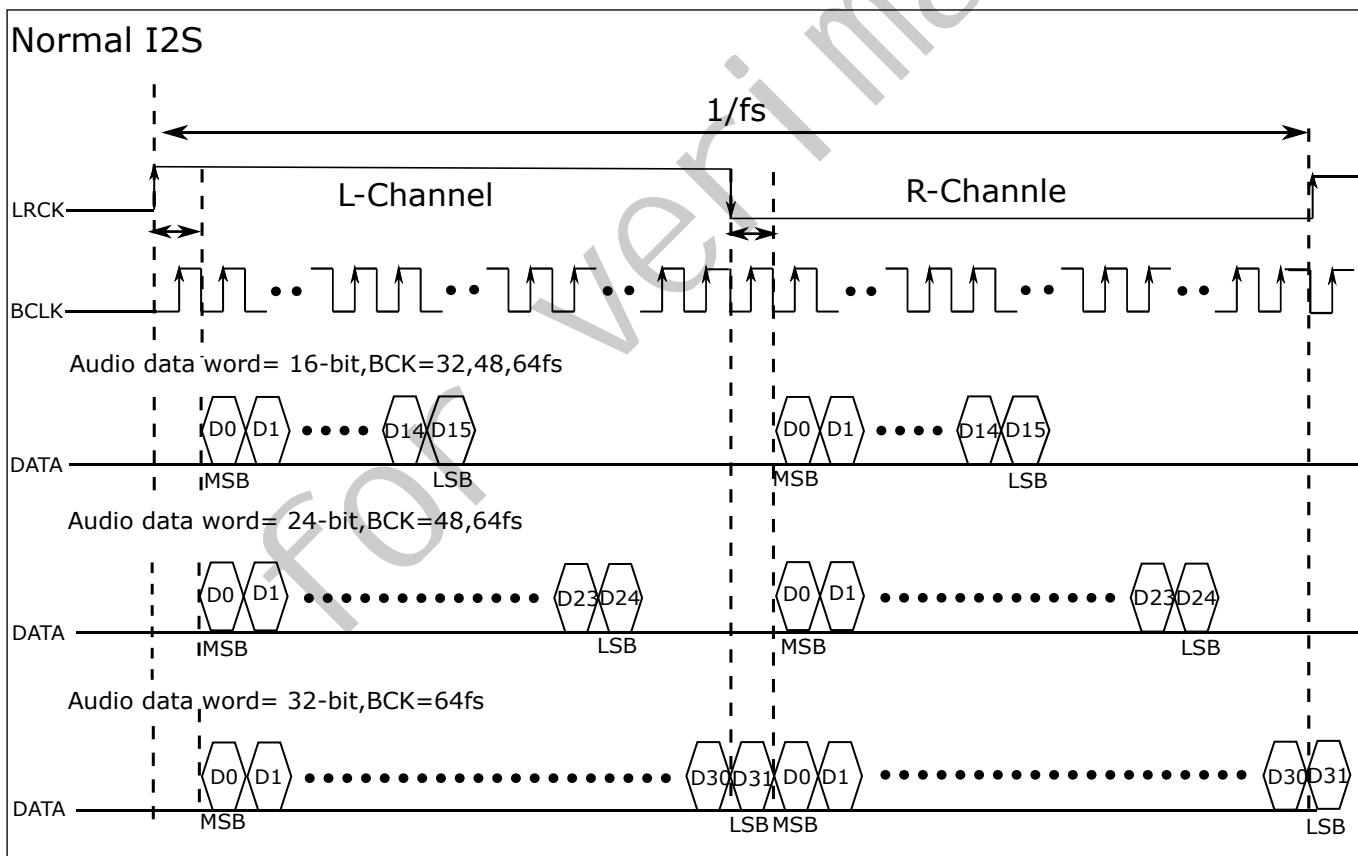
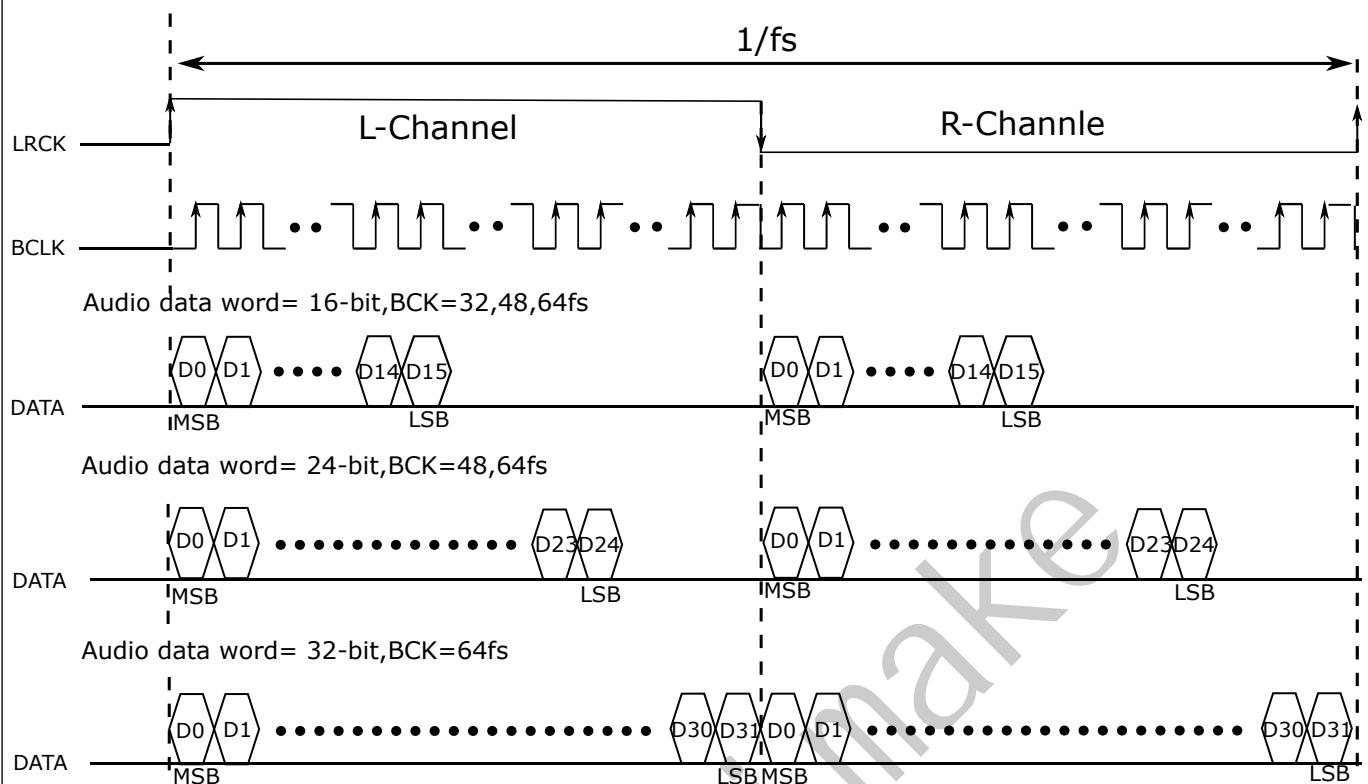


图 14.2: 标准 I2S 数据格式

由 I2S\_CONFIG[25:20] 来配置 OFFSET, 左对齐模式与标准 I2S 的唯一区别在于对 I2S\_CONFIG[25:20] 的配置不同。

## Left-Justified



## Right-Justified

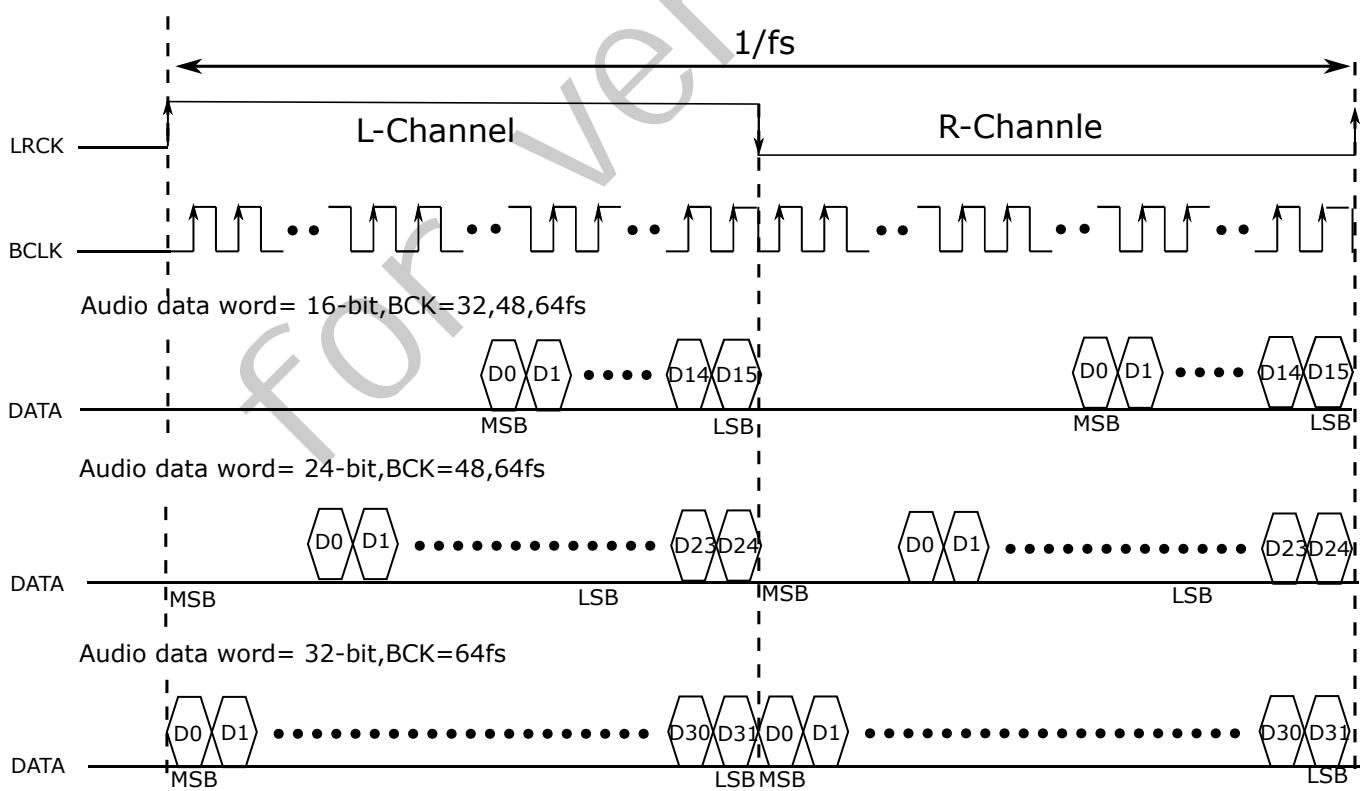


图 14.3: I2S 左对齐/右对齐数据格式

## DSP-MTK TDM64

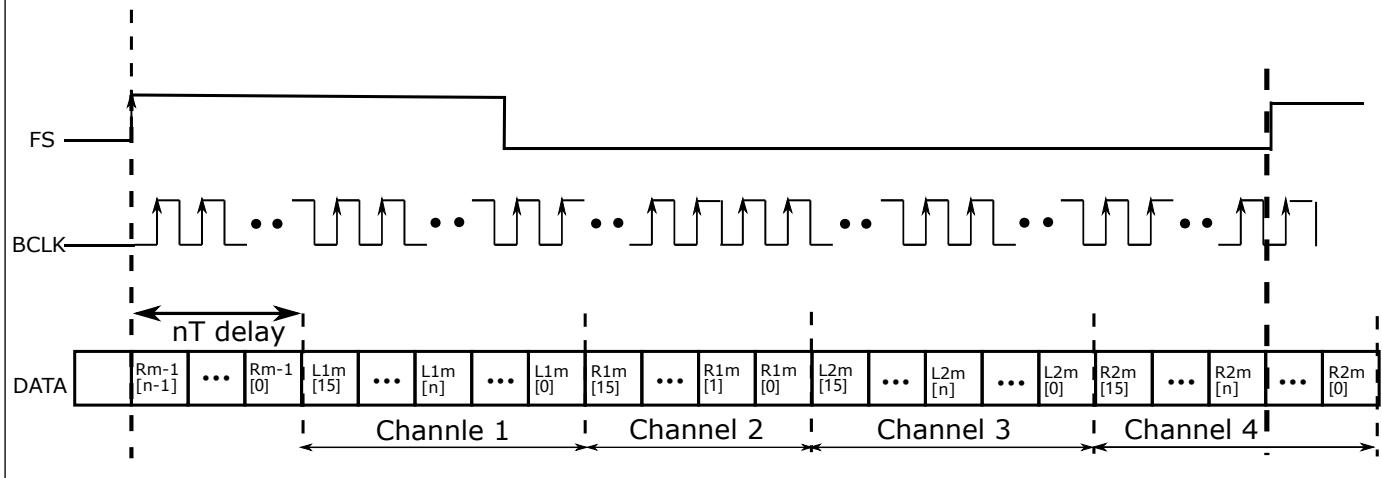


图 14.4: I2S TDM64 模式六通道录音

可以通过 I2S\_CONFIG[6] 来控制单个脉冲的宽度，当选择为 0 时，FS 信号线高电平脉冲的宽度为 data size 的宽度，当选择为 1 的时候，FS 信号线高电平脉冲的宽度为 1。一般情况下，多通道的 TDM64 模式，此寄存器配置为 1。

不同数据格式 I2S\_CONFIG 配置如表所示：

### 14.4.2 基本架构图

### 14.4.3 时钟源

I2S 的时钟源由 audio PLL 提供，时钟中的分频器用于对时钟源进行分频，然后产生时钟信号来驱动 I2S 模块。如下图所示：

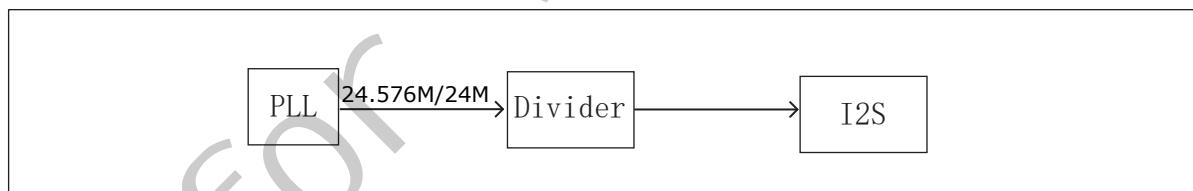


图 14.5: I2S 时钟

### 14.4.4 I2S 中断

I2S 有着丰富的中断控制，包括以下几种中断模式：

- TX FIFO 请求中断
- RX FIFO 请求中断
- overrun/undderun error 中断

当 I2S\_FIFO\_CONFIG\_1 中 TX\_FIFO\_CNT 大于 TX\_FIFO\_TH 时，产生 TX FIFO 请求中断。当条件不满足时该中断

标志会自动清除。

当 I2S\_FIFO\_CONFIG\_1 中 RX\_FIFO\_CNT 大于 RX\_FIFO\_TH 时, 产生 RX FIFO 请求中断。当条件不满足时该中断标志会自动清除。

如果 TX/RX FIFO 发生了上溢或者下溢, 会触发 error 中断, 当异常消失后, 标志位会自动清空。

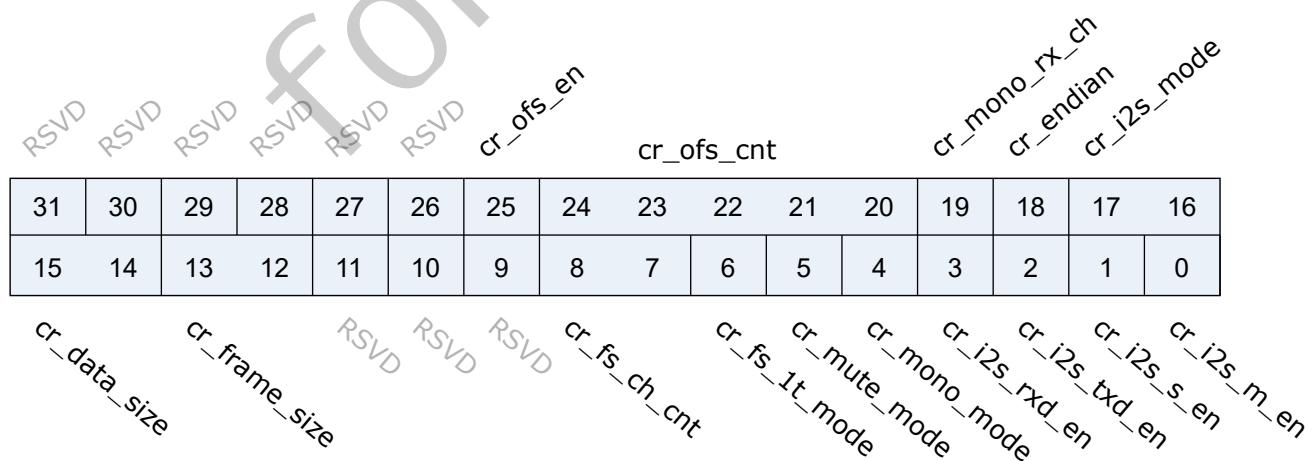
I2S 的所有中断的使能位以及中断标志位都在 I2S\_INT\_STS 寄存器。

## 14.5 寄存器描述

名称	描述
i2s_config	
i2s_int_sts	
i2s_bclk_config	
i2s_fifo_config_0	
i2s_fifo_config_1	
i2s_fifo_wdata	
i2s_fifo_rdata	
i2s_io_config	

### 14.5.1 i2s\_config

地址: 0x40017000

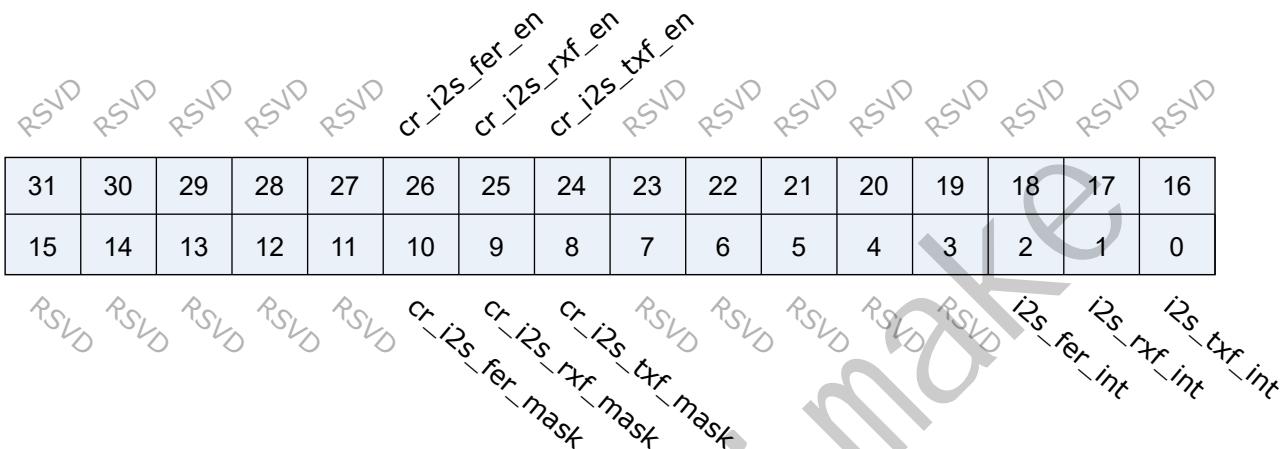


位	名称	权限	复位值	描述
31:26	RSVD			
25	cr_ofs_en	r/w	1'b0	Offset enable 1'b0: Disabled, 1'b1: Enabled
24:20	cr_ofs_cnt	r/w	5'd0	Offset cycle count (unit: cycle of I2S BCLK) 5'd0: 1 cycle 5'd1: 2 cycles ...
19	cr_mono_rx_ch	r/w	1'b0	RX mono mode channel select signal 1'b0: L-channel 1'b1: R-channel
18	cr_endian	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			
8:7	cr_fs_ch_cnt	r/w	2'd0	Channel count of each frame 2'd0: FS 2-channel mode 2'd1: FS 3-channel mode (DSP mode only) 2'd2: FS 4-channel mode (DSP mode only) 2'd3: FS 6-channel mode (DSP mode only) Note: cr_mono_mode & cr_fifo_lr_merge will be invalid in 3-channel mode Note: frame_size must equal data_size in 3/4/6-channel mode
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
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

位	名称	权限	复位值	描述
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

#### 14.5.2 i2s\_int\_sts

地址: 0x40017004



位	名称	权限	复位值	描述
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
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'b1	I2S TX FIFO ready ( $tx\_fifo\_cnt > tx\_fifo\_th$ ) interrupt, auto-cleared when data is pushed

### 14.5.3 i2s\_bclk\_config

地址: 0x40017010

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															
---------------	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

位	名称	权限	复位值	描述
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)

### 14.5.4 i2s\_fifo\_config\_0

地址: 0x40017080

i2s_fifo_config_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

位	名称	权限	复位值	描述
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)

位	名称	权限	复位值	描述
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

#### 14.5.5 i2s\_fifo\_config\_1

地址: 0x40017084

RSVD	RSVD	RSVD	RSVD	RSVD	rx_fifo_th	RSVD	RSVD	RSVD	RSVD	tx_fifo_th
31	30	29	28	27	26	25	24	23	22	21
15	14	13	12	11	10	9	8	7	6	5
rx_fifo_cnt					tx_fifo_cnt					

位	名称	权限	复位值	描述
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

位	名称	权限	复位值	描述
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

#### 14.5.6 i2s\_fifo\_wdata

地址: 0x40017088

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

位	名称	权限	复位值	描述
31:0	i2s_fifo_wdata	w	x	

#### 14.5.7 i2s\_fifo\_rdata

地址: 0x4001708c

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

位	名称	权限	复位值	描述
31:0	i2s_fifo_rdata	r	32'h0	



## 14.5.8 i2s\_io\_config

地址: 0x400170fc

| 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    |      |      |

位	名称	权限	复位值	描述
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

## 15.1 简介

芯片内置一个 PWM 调制模块，用来输出模拟信号驱动喇叭实现音频播放。

## 15.2 主要特征

- 集成 1 路 16bit DAC PWM, 可支持 1 路模拟 PWM 差分输出
  - 采样率: 8k~48k
  - 信噪比 (A-W): 95dB 增益
  - 谐波失真 + 噪声: -70dB @ 0dB 增益
- 独立的数字音量控制，并可以支持音量柔和调节/静音
- 32 位宽度 FIFO，深度为 32
- 支持 DMA 传输模式
- 差值滤波器
- 立体声数据 Mixer 选择器
- 与传统 DAC 联动

## 15.3 时钟树

用户首先启动 Audio PLL 选择对应的频率值，经过一次分频后，进入模块再通过 `aupwm_0` 的 `au_pwm_mode` 选择分频系数此寄存器数值与分频比如下所示

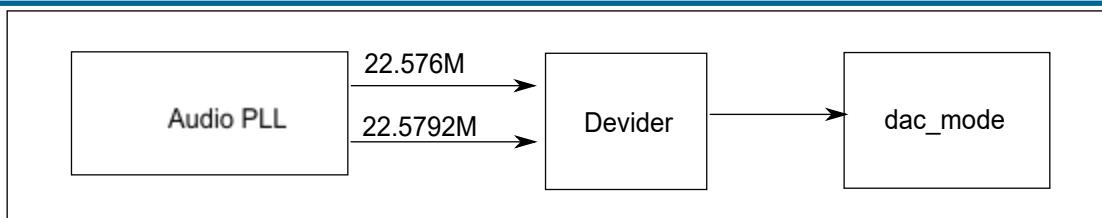


图 15.1: 时钟示意图

## 15.4 功能描述

AudioPWM 模块基本框图如图所示。

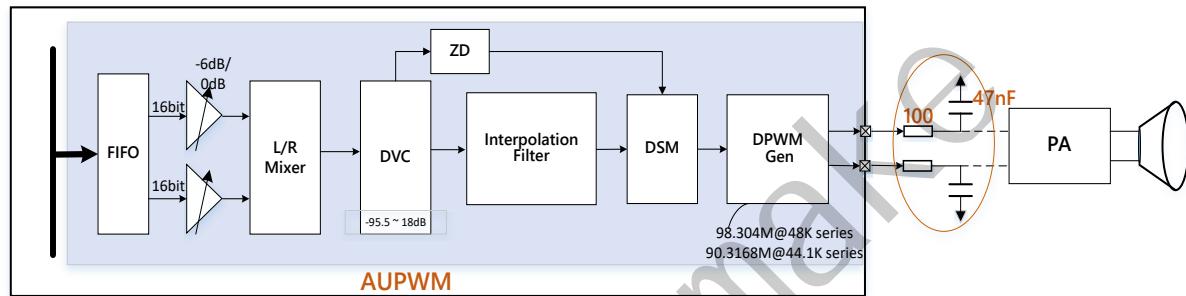


图 15.2: 模块示意图

AudioPWM 会将 TX FIFO 中的数据通过差值滤波器后进行 DSM 调制，最终会输出占空比与 TX FIFO 数据相关的 PWM 信号。此信号再通过 RC 低通滤波器，即可驱动喇叭播放音频数据。

### 15.4.1 AudioPWM 中断

AudioPWM 有着丰富的中断控制，包括以下几种中断模式：

- TX FIFO 请求中断
- TX FIFO underrun 中断
- TX FIFO overrun 中断
- 音量调节完成中断

当 TX\_FIFO\_CTRL 中 TX\_DRQ\_CNT 大于 TX\_TRG\_LEVEL 时，产生 TX FIFO 请求中断。当条件不满足时该中断标志会自动清除。

当 TX FIFO 中并没有数据，但是用户却通过 TX\_FIFO\_CTRL 中的 TX\_CH\_EN 使能了 TX FIFO 状态机，则会进入 TX FIFO underrun 中断。

当用户填入超过 TX FIFO 最大深度的数据的时候，会导致 TX FIFO 溢出，从而产生 TX FIFO overrun 中断。

Audio PWM 音量控制支持渐入/渐出的功能，因此当音量调节完成之后会进入到音量调节完成中断，提示渐入/渐出调

节完成。

### 15.4.2 FIFO 格式控制

AUPWM\_TX\_FIFO\_CTRL 可以控制音频数据储存在 FIFO 的格式。

FIFO 控制器支持如下四种数据存储格式，由 FIFO\_CTRL[25:24] 来决定。

- Mode 0:

DATA[15:0] = {FIFO[31:16]}

- Mode 1:

DATA[15:0] = {FIFO[23:8]}

- Mode 2:

DATA[15:0] = {FIFO[19:4]}

- Mode 3:

DATA[15:0] = {FIFO[15:0]}

最高有效位的分布

- Mode 0:

有效数据的最高位在 31 bits

- Mode 1:

有效数据的最高位在 23 bits

- Mode 2:

有效数据的最高位在 19 bits

- Mode 3:

有效数据的最高位在 15 bits

当待播放的音频文件是 16bit 的宽度时，选择 Mode3 即可。因为我们 DAC 的最大分辨率就是 16bits。其他模式存在的意义在于，如果待播放的音频文件宽度为 32/24/20 Bits 时，用户需要做出取舍，将低位的一些信息裁剪掉，以保证后级电路获得的是 16bits 宽度的数据，这里默认是将低位舍弃。

### 15.4.3 FIFO 的启动与 DMA 搬运

AWPWM 的 TX FIFO 数据可以通过 DMA 进行搬运。

用户可以通过 PDM\_TX\_FIFO\_STATUS 寄存器实时获得目前 FIFO 有效数据的数量。

通过配置 FIFO\_CTRL[15:14] 来选择发起 DMA request 的 FIFO count 阈值，是 8/16/32，或者是由 FIFO\_CTRL[22:16] 配置来决定。

当 count 的值大于设定阈值，并且 PDM\_TX\_FIFO\_CTRL[12:8] 对应通路的 FIFO 被使能，则会发起一次 DMA 搬运。

注意，启动 TX FIFO 时，如果 TX FIFO 里面并没有有效的数据，则会触发 tx underrun 错误。因此要注意软件配置顺序。

### 15.4.4 音频声道选择器

Audio PWM 仅支持一路音频数据的播放，如果需要播放立体声音频需要使用音频声道选择器模块。此模块的典型应用是将存储器中的音频文件通过 DMA 搬运到 TX FIFO 中实现 Audio PWM 播放立体声音频。此时用户仅需要使用 DMA 将立体声数据依次顺序搬运进入 TX FIFO，并且通过 awpwm\_1 的 dac\_mix\_sel 寄存器可以选择仅播放左声道、仅播放右声道、两声道数据之和或两声道数据平均数。此功能与 awpwm\_fifo\_ctrl 的 tx\_ch\_en 同时使用，具体过程见下图所示。

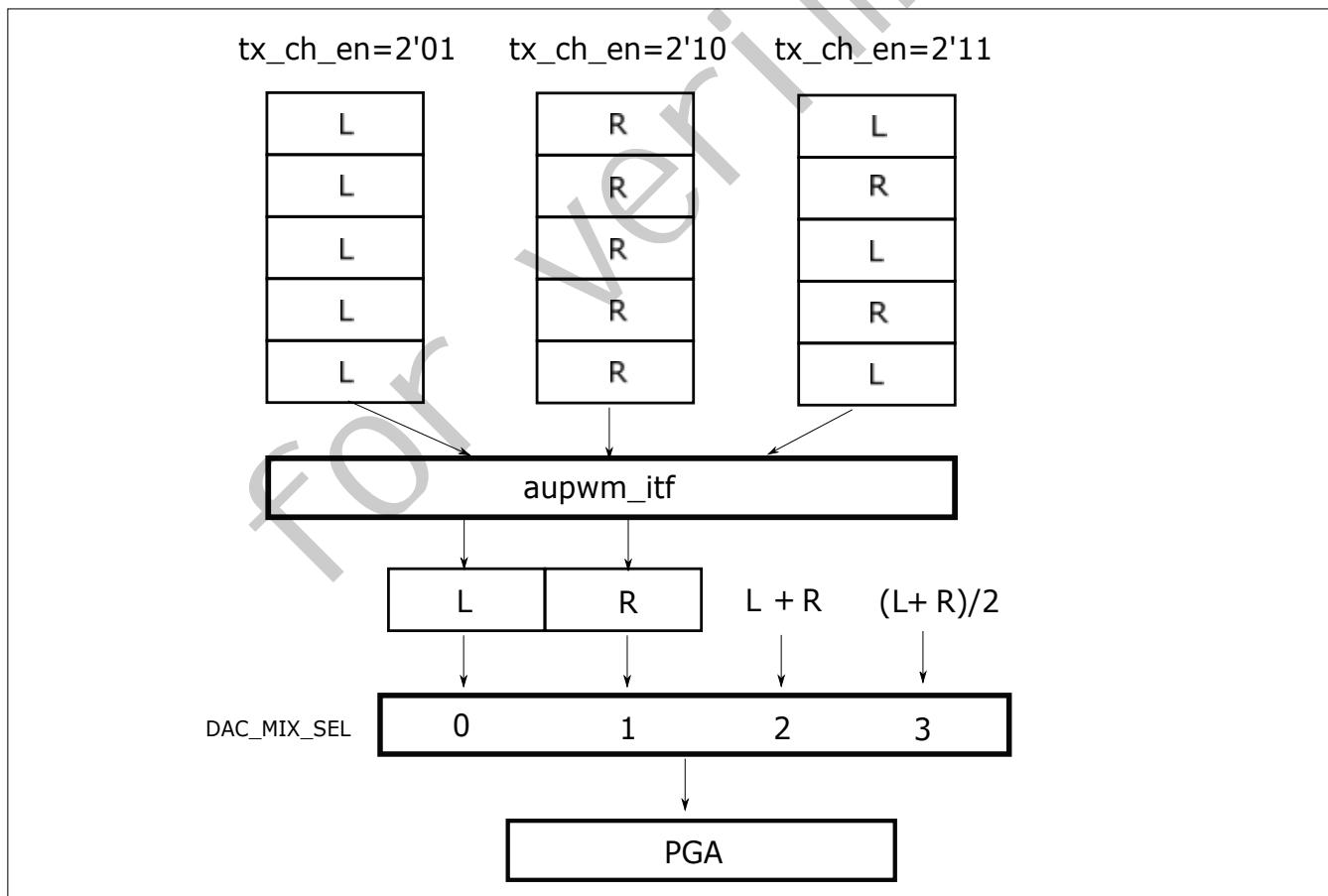


图 15.3: Mixer 示意图

如上图所示，通过 tx\_ch\_en 来控制 DMA 顺序搬入的数据如何填入 L 和 R 两个缓存。当播放源为单声道的时候，选择 tx\_ch\_en=2'01 或 tx\_ch\_en=2'10，此时数据会按顺序依次填入 R/L。另外一个 L/R 不会被更新到数据。此时通过 Mixer 选择器选择对应的声音进入下一级调制电路。

当播放源为立体声的时候，选择 tx\_ch\_en=2'11，此时 DMA 顺序搬来的数据会依次填入 L,R 缓存中，这样保证 L 和 R 都会被更新到，立体声音频数据源的保存格式也是 L,R,L,R 依次排列的形式，因此 L 和 R 会被正确更新到对应音频的数据。此时可以通过 Mixer 选择器选择需要哪一路声道或者将两个声道求和或取平均进入下一级调制电路。

注意 tx\_ch\_en 和 Mixer 选择。如果通过 tc\_ch\_en 将单声道数据放入左声道，但是 mixer 却是选择右声道，那么则会播放零数据，造成错误。

#### 15.4.5 音量控制

用户可以通过 awpwm\_s0 的 dac\_s0\_volume 寄存器配置音量，增益范围为 -95.5dB-18dB，寄存器值 \*0.5 为真实的增益值。配置完成之后对 dac\_s0\_volume\_update 寄存器写 1 更新操作。

用户可以通过 dac\_s0\_mute\_softmode, dac\_s0\_ctrl\_mode 选择音量调节模式，0 代表立即调节不做渐进处理，1 代表使用过零方式渐进调节 2 代表使用 ramp 方式渐进调节。dac\_s0\_mute\_rmpup\_rate、dac\_s0\_ctrl\_zcd\_rate 等寄存器调节 ramp 或 corss zero 的斜率。

#### 15.4.6 ZeroDetect

AudioPWM 提供 ZeroDetect 功能，当打开此功能 (aupwm\_zd\_0[16]=1) 时，TX FIFO 中的数据一直保持 0，则会关闭数字通道，保持输出为 0。这样做的目的是减少播 0 数据的时候带来的噪声，使得音频系统更加稳定。

### 15.5 配置流程

1. 根据待播的音频采样率，通过 aupwm\_0[31:28] 选择对应采样率
2. 根据待播的音频声道数决定 Mixer 寄存器的配置
3. 配置 DMA 将数据搬运到 AudioPWM TX FIFO
4. 通过 aupwm\_fifo\_ctrl 的 tx\_ch\_en 使能状态机，开始播放
5. 在播放的过程中调整音量（可选）

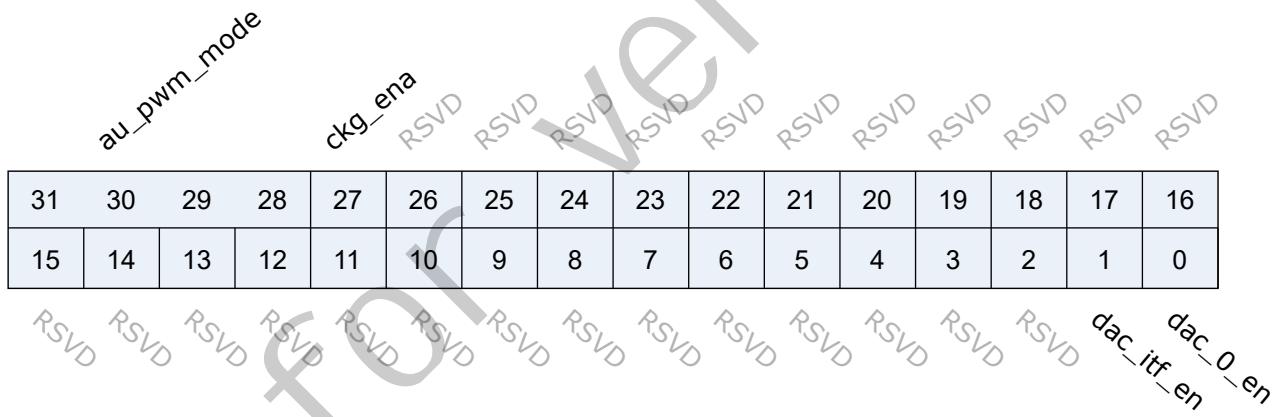
### 15.6 寄存器描述

名称	描述
aupwm_0	
aupwm_status	
aupwm_s0	
aupwm_s0_misc	

名称	描述
aupwm_zd_0	
aupwm_1	
aupwm_rsvd	
aupwm_test_0	
aupwm_test_1	
aupwm_test_2	
aupwm_test_3	
aupwm_fifo_ctrl	
aupwm_fifo_status	
aupwm_fifo_data	

### 15.6.1 aupwm\_0

地址: 0x20055000



位	名称	权限	复位值	描述
31:28	au_pwm_mode	r/w	4'd0	0:pwm 8k, 1:pwm 16k, 2:pwm 32k, 3:pwm 24k, 4:pwm 48k, 5:pwm 22.05k, 6:pwm 44.1k, 7:pwm 44.1k, 8:gpdac 8k, 9:gpdac 16k, 10:gpdac 32k, 11:gpdac 24k, 12:gpdac 48k, 13:gpdac 22.05k, 14:gpdac 44.1k, 15:gpdac 44.1k
27	ckg_ena	r/w	1	1:clock gen enable
26:2	RSVD			
1	dac_itf_en	r/w	0	1:enable dac to audio dma interface

位	名称	权限	复位值	描述
0	dac_0_en	r/w	0	1:enable dac ch0

## 15.6.2 aupwm\_status

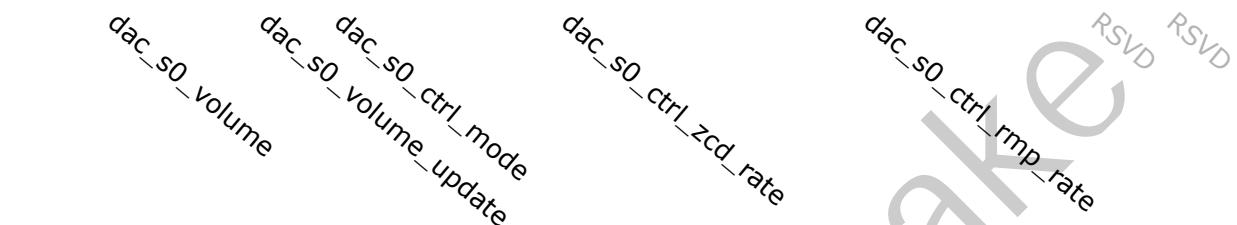
地址: 0x20055004

位	名称	权限	复位值	描述
31:25	RSVD			
24	audio_int_all	r	0	1:mute signal to analog
23	zd_amute	r	0	1:mute signal to analog
22:18	RSVD			
17	dac_s0_int_clr	r/w	0	1: clear interrupt
16	dac_s0_int	r	0	interrupt value
15:14	RSVD			
13	dac_h0_mute_done	r	1	1:dvga ch0 mute done
12	dac_h0_busy	r	0	1:dvga ch0 busy
11:0	RSVD			

### 15.6.3 aupwm\_s0

地址: 0x20055008

dac_s0_volume															
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



位	名称	权限	复位值	描述
31	dac_s0_mute	r/w	0	dac dpga ch0 sw volume control 1:mute
30	dac_s0_mute_softmode	r/w	1	0:mute directly, 1:mute with ramp down
29:26	dac_s0_mute_rmpdn_rate	r/w	4'd6	mute ramp down rate: 0:2 fs sample 1:4 fs sample 2:8 fs sample 3:16 fs sample 4:32 fs sample 5:64 fs sample 6:128 fs sample 7:256 fs sample 8:512 fs sample 9:1024 fs sample 8:2048 fs sample

位	名称	权限	复位值	描述
25:22	dac_s0_mute_rmpup_rate	r/w	4'd0	mute ramp up rate 0:2 fs sample 1:4 fs sample 2:8 fs sample 3:16 fs sample 4:32 fs sample 5:64 fs sample 6:128 fs sample 7:256 fs sample 8:512 fs sample 9:1024 fs sample 8:2048 fs sample
21:13	dac_s0_volume	r/w	9'd0	volume s9.1, -95.5dB +18dB in 0.5dB step
12	dac_s0_volume_update	r/w	0	1:volume update
11:10	dac_s0_ctrl_mode	r/w	2'd2	0:direct force volume, 1:update volume at zero crossing, 2:update volume with ramp
9:6	dac_s0_ctrl_zcd_rate	r/w	4'd2	zero crossing rate 0:2 fs sample 1:4 fs sample 2:8 fs sample 3:16 fs sample 4:32 fs sample 5:64 fs sample 6:128 fs sample 7:256 fs sample 8:512 fs sample 9:1024 fs sample 8:2048 fs sample
5:2	dac_s0_ctrl_rmp_rate	r/w	4'd6	ramp rate 0:2 fs sample 1:4 fs sample 2:8 fs sample 3:16 fs sample 4:32 fs sample 5:64 fs sample 6:128 fs sample 7:256 fs sample 8:512 fs sample 9:1024 fs sample 8:2048 fs sample

位	名称	权限	复位值	描述
1:0	RSVD			

#### 15.6.4 aupwm\_s0\_misc

地址: 0x2005500c

dac_s0_ctrl_zcd_timeout															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	RSVD														
14	RSVD														
13	RSVD														
12	RSVD														
11	RSVD														
10	RSVD														
9	RSVD														
8	RSVD														
7	RSVD														
6	RSVD														
5	RSVD														
4	RSVD														
3	RSVD														
2	RSVD														
1	RSVD														
0	RSVD														

位	名称	权限	复位值	描述
31:28	dac_s0_ctrl_zcd_timeout	r/w	4'd4	zero crossing time out period 0:2 fs sample 1:4 fs sample 2:8 fs sample 3:16 fs sample 4:32 fs sample 5:64 fs sample 6:128 fs sample 7:256 fs sample 8:512 fs sample 9:1024 fs sample 8:2048 fs sample
27:0	RSVD			

### 15.6.5 aupwm\_zd\_0

地址: 0x20055010

RSVD	zd_en															
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	

zd\_time

位	名称	权限	复位值	描述
31:17	RSVD			
16	zd_en	r/w	0	two channels, 0: zd disabled, 1: zd on channel0, 2: zd on channel1, 3: zd on both channels
15	RSVD			
14:0	zd_time	r/w	15'd512	number of zeros

### 15.6.6 aupwm\_1

地址: 0x20055014

RSVD	dac_dsm_dither_prbs_mode															
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	

dac_dsm_dither_en	dac_dsm_dither_amp	dac_dsm_scaling_en	dac_dsm_scaling_mode	dac_dsm_order	dac_dsm_out_fmt	dac_mix_sel

位	名称	权限	复位值	描述
31:17	RSVD			
16:15	dac_dsm_dither_prbs_mode	r/w	0	dac dsm dither Ifsr mode:0:LFSR32, 1:LFSR24, 2:LFSR16, 3:LFSR12
14	dac_dsm_dither_en	r/w	1	1:enable dac dsm dither
13:11	dac_dsm_dither_amp	r/w	0	dac dsm dither amplitue
10	dac_dsm_scaling_en	r/w	1	1:enable dac dsm scaling
9	RSVD			
8:7	dac_dsm_scaling_mode	r/w	0	dac dsm scaling value; u4.4
6:5	dac_dsm_order	r/w	0	0: 2-order, 1: 3-order
4	dac_dsm_out_fmt	r/w	0	0:offset binary 1:2's complement
3:2	RSVD			
1:0	dac_mix_sel	r/w	0	0: L channel, 1:R channel, 2: L+R, 3: (L+R)/2

### 15.6.7 aupwm\_rsrd

地址: 0x20055018

au_pwm_reserved															
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

### au\_pwm\_reserved

位	名称	权限	复位值	描述
31:0	au_pwm_reserved	r/w	32'hffff0000	

### 15.6.8 aupwm\_test\_0

地址: 0x2005501c

### dac\_dpga\_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

### dac\_in\_0

位	名称	权限	复位值	描述
31:16	dac_dpga_0	r	0	
15:0	dac_in_0	r	0	

## 15.6.9 aupwm\_test\_1

地址: 0x20055020

RSVD	dac_fir_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			

位	名称	权限	复位值	描述
31:17	RSVD			
16:0	dac_fir_0	r	0	

## 15.6.10 aupwm\_test\_2

地址: 0x20055024

| 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    |      |      |

dac\_sinc\_0

位	名称	权限	复位值	描述
31:16	RSVD			
15:0	dac_sinc_0	r	0	

### 15.6.11 aupwm\_test\_3

地址: 0x20055028

au\_pwm\_test\_read

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

au\_pwm\_test\_read

位	名称	权限	复位值	描述
31:0	au_pwm_test_read	r	0	

### 15.6.12 aupwm\_fifo\_ctrl

地址: 0x2005508c

RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	tx_data_mode	RSVD	RSVD	RSVD	tx_trg_level					
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

*tx\_dra\_cnt*      RSVD      RSVD      RSVD      RSVD      *tx\_ch\_en*      RSVD      RSVD      RSVD      *tx\_dra\_en*      *txa\_int\_en*      *txu\_int\_en*      *txo\_int\_en*      *tx\_fifo\_flush*

位	名称	权限	复位值	描述
31:26	RSVD			
25:24	tx_data_mode	r/w	2'b0	TX_FIFO_DATIN_MODE. TX FIFO DATA Input Mode (Mode 0, 1, 2, 3) Mode 0: Valid data's MSB is at [31] of TX_FIFO register Mode 1: Valid data's MSB is at [23] of TX_FIFO register Mode 2: Valid data's MSB is at [19] of TX_FIFO register Mode 3: Valid data's MSB is at [15] of TX_FIFO register For 16-bits transmitted audio sample: Mode 0: FIFO_I[15:0] = TXDATA[31:16] Mode 1: FIFO_I[15:0] = TXDATA[23:8] Mode 2: FIFO_I[15:0] = TXDATA[19:4] Mode 3: FIFO_I[15:0] = TXDATA[15:0]

位	名称	权限	复位值	描述
23:21	RSVD			
20:16	tx_trg_level	r/w	5'd7	<p>TX_FIFO_TRG_LEVEL.</p> <p>TX FIFO Trigger Level (TXTL[4:0])</p> <p>Interrupt and DMA request trigger level for TX FIFO room available condition</p> <p>IRQ/DRQ Generated when WLEVEL &gt; TXTL[4:0]</p> <p>Notes:</p> <p>WLEVEL represents the number of room available in the TX FIFO</p>
15:14	tx_drq_cnt	r/w	2'b0	<p>DAC_DRQ_CLR_CNT.</p> <p>When TX FIFO available room less than or equal N, DRQ Request will be de-asserted. N is defined here:</p> <ul style="list-style-type: none"> <li>00: IRQ/DRQ de-asserted when WLEVEL &lt;= TXTL[4:0]</li> <li>01: IRQ/DRQ de-asserted when WLEVEL &lt; 2</li> <li>10: IRQ/DRQ de-asserted when WLEVEL &lt; 4</li> <li>11: IRQ/DRQ de-asserted when WLEVEL &lt; 8</li> </ul> <p>WLEVEL represents the number of room available in the TX FIFO</p>
13:10	RSVD			
9:8	tx_ch_en	r/w	2'b0	<p>TX_FIFO_DATOUT_DST.</p> <p>TX FIFO Data Output Destination Select.</p> <ul style="list-style-type: none"> <li>0: Disable 1: Enable</li> <li>Bit9: DAC2 data</li> <li>Bit8: DAC1 data</li> </ul> <p>When some of the above bits set to '1', these data are always arranged in order from low-bit to high-bit.(bit8-&gt;bit9)</p>
7:5	RSVD			
4	tx_drq_en	r/w	1'b0	<p>DAC_DRQ_EN.</p> <p>DAC FIFO Room Available DRQ Enable.</p> <ul style="list-style-type: none"> <li>0: Disable</li> <li>1: Enable</li> </ul>
3	txa_int_en	r/w	1'b0	<p>DAC_IRQ_EN.</p> <p>DAC FIFO Room Available IRQ Enable.</p> <ul style="list-style-type: none"> <li>0: Disable</li> <li>1: Enable</li> </ul>
2	txu_int_en	r/w	1'b0	<p>DAC_UNDERRUN_IRQ_EN.</p> <p>DAC FIFO Under Run IRQ Enable</p> <ul style="list-style-type: none"> <li>0: Disable</li> <li>1: Enable</li> </ul>

位	名称	权限	复位值	描述
1	txo_int_en	r/w	1'b0	DAC_OVERRUN_IRQ_EN. DAC FIFO Over Run IRQ Enable 0: Disable 1: Enable
0	tx_fifo_flush	w1c	1'b0	DAC_FIFO_FLUSH. DAC FIFO Flush. Write ‘1’ to flush TX FIFO, self clear to ‘0’ .

### 15.6.13 aupwm\_fifo\_status

地址: 0x20055090

RSVD	txa	RSVD	RSVD	RSVD	txa_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 RSVD RSVD RSVD RSVD RSVD RSVD RSVD RSVD txa int RSVD txu int RSVD txo int RSVD

位	名称	权限	复位值	描述
31:25	RSVD			
24	txa	r	1'b1	TXA. TX FIFO Room Available 0: No room for new sample in TX FIFO 1: More than one room for new sample in TX FIFO (>= 1 word)
23:21	RSVD			
20:16	txa_cnt	r	5'd16	TXA_CNT. TX FIFO Available Room Word Counter
15:5	RSVD			
4	txa_int	r	1'b0	TXA_INT. TX FIFO Room Available Pending Interrupt 0: No Pending IRQ 1: Room Available Pending IRQ Automatic clear if interrupt condition fails.
3	RSVD			

位	名称	权限	复位值	描述
2	txu_int	r	1'b0	TXU_INT. TX FIFO Underrun Pending Interrupt 0: No Pending IRQ 1: FIFO Underrun Pending IRQ Write ‘1’ to clear this interrupt
1	txo_int	r	1'b0	TXO_INT. TX FIFO Overrun Pending Interrupt 0: No Pending IRQ 1: FIFO Overrun Pending IRQ Write ‘1’ to clear this interrupt
0	RSVD			

### 15.6.14 aupwm\_fifo\_data

地址: 0x20055094

tx\_data

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

tx\_data

位	名称	权限	复位值	描述
31:0	tx_data	w	32'h0	TX_DATA. Transmitting left, right channel sample data should be written this register one by one. The left channel sample data is first and then the right channel sample.

## 16.1 简介

芯片内置一个 16bit ADC，可实现音频数字 PDM 信号采样或者模拟信号采样。

## 16.2 主要特征

- 集成 1 路 16bit ADC, 可支持 1 路模拟 mic 差分/单端输入, 复用 GPIO 输入
  - 采样率: 8k~96k
  - 信噪比 (A-W): 90dB @ 6dB 增益
  - 谐波失真 + 噪声: -80dB @ 6dB 增益
  - 模拟前置增益: 6~42 dB, 3dB 一档
- 可调节的高通滤波器和独立的数字音量控制
- 支持数字 mic 接口, 复用 GPIO 输入
- 支持直流测量模式, 精度 16bit
- 独立的数字音量控制
- 32 位宽度 FIFO 深度为 32
- 支持 DMA 传输模式

## 16.3 时钟树

用户首先启动 Audio PLL 选择对应的频率值，经过一次分频后，进入模块再通过 audpdm\_top 的 adc\_rate 选择分频系数此寄存器数值与分频比如下所示

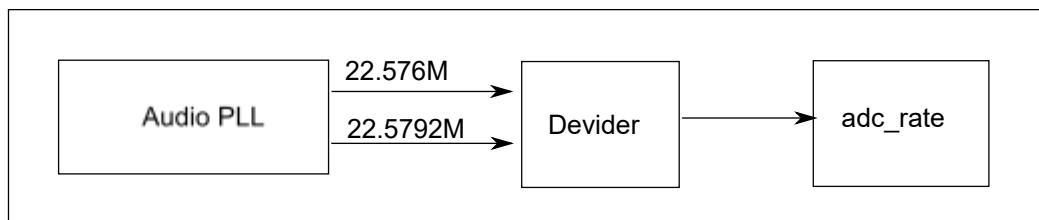


图 16.1: 时钟示意图

## 16.4 功能描述

AudioADC 模块基本框图如图所示。

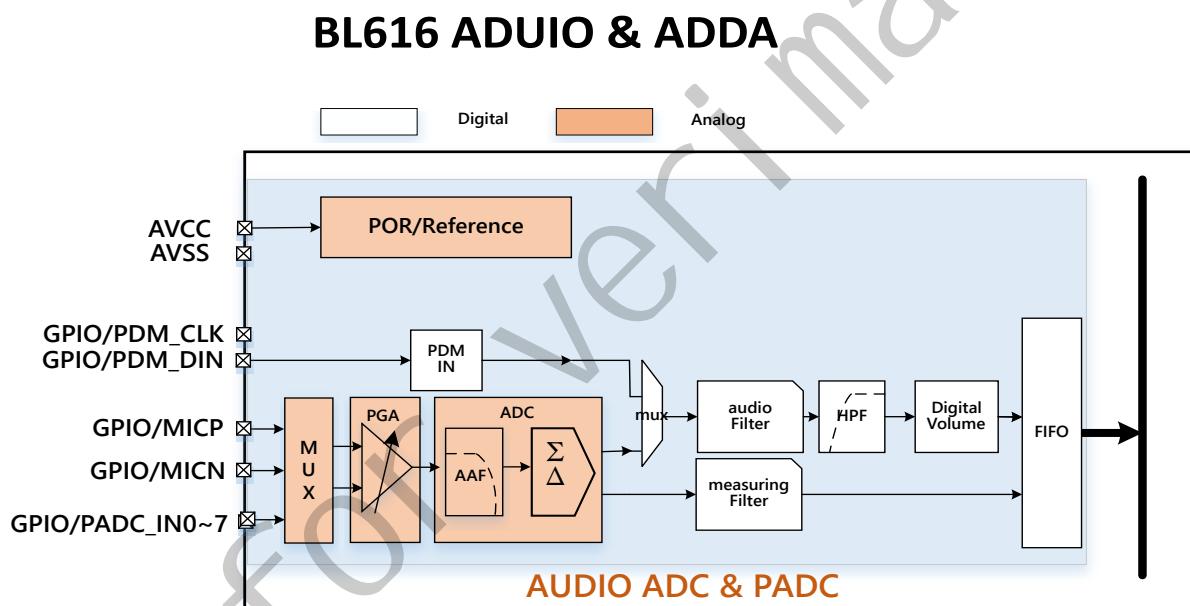


图 16.2: 模块框图

AUADC 模块输入信号既可以是数字 PDM 信号，也可以是模拟信号 (由 pdm\_dac\_0[12] 决定)。当选择为 PDM 数字信号的时候，输入会进入 PDM 解调器，然后经过滤波器，音量控制器进入 RX FIFO。当选择为数字信号时，会经过 PGA 通过 ADC 转换器以及对应滤波器，最终进入 RX FIFO。

### 16.4.1 PDM 左右声道选择

当输入信号为数字 PDM 格式时，可以通过 PDM\_PDM\_0 的 adc\_0\_pdm\_sel 功能选择是录制 PDM 的左声道或是右声道。

### 16.4.2 AUADC 中断

AUADC 模块有着丰富的中断控制，包括以下几种中断模式：

- RX FIFO 请求中断
- RX FIFO underrun 中断
- RX FIFO overrun 中断

当 RX\_FIFO\_CTRL 中 RX\_DRQ\_CNT 大于 RX\_TRG\_LEVEL 时，产生 RX FIFO 请求中断。当条件不满足时该中断标志会自动清除。

当 RX FIFO 中并没有数据，但是用户却通过 RX\_FIFO\_CTRL 中的 RX\_CH\_EN 使能了 RX FIFO 调制，则会进入 RX FIFO underrun 中断。

当用户填入超过 RX FIFO 最大深度的数据的时候，会导致 RX FIFO 溢出，从而产生 RX FIFO overrun 中断。

### 16.4.3 FIFO 格式控制

AUADC\_RX\_FIFO\_CTRL 寄存器可以控制音频数据储存在 FIFO 的格式。

FIFO 控制器支持如下四种数据存储格式，由 FIFO\_CTRL[25:24] 来决定。

- Mode 0:

$$\text{DATA}[31:0] = \{\text{FIFO}[15:0], 16'h0\}$$

- Mode 1:

$$\text{DATA}[31:0] = \{8\{\text{FIFO}[15]\}, \text{FIFO}[15:0], 8'h0\}$$

- Mode 2:

$$\text{DATA}[31:0] = \{12\{\text{FIFO}[15]\}, \text{FIFO}[15:0], 4'h0\}$$

- Mode 3:

$$\text{DATA}[31:0] = \{16\{\text{FIFO}[15]\}, \text{FIFO}[15:0]\}$$

最高有效位的分布

- Mode 0:

有效数据的最高位在 31 bits

- Mode 1:

有效数据的最高位在 23 bits

- Mode 2:

有效数据的最高位在 19 bits

- Mode 3:

有效数据的最高位在 15 bits

如果对储存格式没有特殊要求，一般来说选择 Mode3 即可，因为 ADC 的分辨率最大为 16bit，使用 16bit 宽度的 ram 去储存音频是效率最高的选择。其他格式会将有效的 16bit 数据，放在 32bits 宽度的不同位置，低位使用 0 去补齐，高位使用符号位进行补齐。

#### 16.4.4 FIFO 的启动与 DMA 搬运

PDM 的 FIFO 数据可以通过 DMA 进行搬运。

用户可以通过 PDM\_RX\_FIFO\_STATUS 寄存器实时获得目前 FIFO 有效数据的数量。

通过配置 FIFO\_CTRL[15:14] 来选择发起 DMA request 的 FIFO count 阈值，是 8/16/32，或者是由 FIFO\_CTRL[22:16] 配置来决定。

当 count 的值大于设定阈值，并且 PDM\_RX\_FIFO\_CTRL[12:8] 对应通路的 FIFO 被使能，则会发起一次 DMA 搬运。

注意，启动 TX FIFO 时，如果 TX FIFO 里面并没有有效的数据，则会触发 tx underrun 错误。因此要注意软件配置顺序。

### 16.5 配置流程

1. 选择录制音频采样率，通过 audpdm\_top[31:28] 选择对应采样率
2. 根据录制的数据源是否是 PDM 数字信号或者是模拟信号配置 pdm\_dac\_0 的 adc\_0\_src 寄存器
3. 如果是 pdm 格式，通过 pdm\_pdm\_0 的 adc\_0\_pdm\_sel 选择 pdm 的声道
4. 配置 DMA 将 Audio 的 RX FIFO 数据实时搬运到指定区域
5. 通过 audadc\_rx\_fifo\_ctrl 的 rx\_ch\_en 打开状态机开始录音
6. 在录制的过程中调整音量（可选）

## 16.6 寄存器描述

名称	描述
audpdm_top	
audpdm_if	
pdm_adc_0	
pdm_adc_1	
pdm_dac_0	
pdm_pdm_0	
pdm_dbg_0	
pdm_dbg_1	
pdm_dbg_2	
pdm_adc_s0	
audadc_ana_cfg1	
audadc_ana_cfg2	
audadc_cmd	
audadc_data	
audadc_rx_fifo_ctrl	
audadc_rx_fifo_status	
audadc_rx_fifo_data	

### 16.6.1 audpdm\_top

地址: 0x2000ac00

adc_rate																
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	RSVD	RSVD	RSVD	RSVD	RSVD	pdm_itf_inv_sel	adc_itf_inv_sel	RSVD	audio_ckg_en

位	名称	权限	复位值	描述
31:28	adc_rate	r/w	4'd1	adc fs 0:8kHz, 1:16kHz, 2:24kHz(22.05k), 3:32kHz, 4:48kHz(44.1k), 5:96kHz, 6:reserved, 7:manual, 8:padc 128kHz, 9:padc 256kHz, 10:padc 512kHz
27:4	RSVD			
3	pdm_itf_inv_sel	r/w	1'd0	1:invert clk_pdm_inf
2	adc_itf_inv_sel	r/w	1'd0	1:invert clk_adc_itf
1	RSVD			
0	audio_ckg_en	r/w	1'd0	1:enable audio clock generator

## 16.6.2 audpdm\_itf

地址: 0x2000ac04

adc_itf_en															
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	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD

位	名称	权限	复位值	描述
31	RSVD			
30	adc_itf_en	r/w	1'd0	1:enable adc to audio dma interface
29:1	RSVD			

位	名称	权限	复位值	描述
0	adc_0_en	r/w	1'd0	1:enable adc ch0

### 16.6.3 pdm\_adc\_0

地址: 0x2000ac08

## 16.6.4 pdm\_adc\_1

地址: 0x2000ac0c

位	名称	权限	复位值	描述
8:5	adc_0_k2	r/w	4'd13	adc ch0 hpf parameter k2
4	RSVD			
3:0	adc_0_k1	r/w	4'd8	adc ch0 hpf parameter k1

### 16.6.5 pdm\_dac\_0

地址: 0x2000ac10

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

位	名称	权限	复位值	描述
31:13	RSVD			
12	adc_0_src	r/w	1'd0	0:adc, 1:pdm
11:10	RSVD			
9:6	adc_pdm_l	r/w	4'b1010	pdm low value
5:4	RSVD			
3:0	adc_pdm_h	r/w	4'b0110	pdm high value

### 16.6.6 pdm\_pdm\_0

地址: 0x2000ac1c

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

位	名称	权限	复位值	描述
31:6	RSVD			
5:3	adc_0_pdm_sel	r/w	3'd0	adc ch0 source select: 0:pdm_0_l, 1:pdm_0_r, 2:pdm_1_l, 3:pdm_1_r, 4:pdm_2_l, 5:pdm_2_r
2:1	RSVD			
0	pdm_0_en	r/w	1'd0	1:enable pdm_0

## 16.6.7 pdm\_dbg\_0

地址: 0x2000ac24

位	名称	权限	复位值	描述
31:30	RSVD			
29:24	aud_test_read_sel	r/w	6'd0	select aud_test_read(0x28) test point
23	adc_test_din_en	r/w	1'd0	1:enable adc test data input from GPIO
22	RSVD			
21	adc_test_clkin_en	r/w	1'd0	1:enable adc test clkok input from GPIO
20:0	RSVD			

## 16.6.8 pdm\_dbg\_1

地址: 0x2000ac28

## aud\_test\_read

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

aud test read

位	名称	权限	复位值	描述
31:0	aud_test_read	r	32'd0	audio test read value

## 16.6.9 pdm\_dbg\_2

地址: 0x2000ac2c

位	名称	权限	复位值	描述
31:21	RSVD			
20	adc_0_fir_4s_en	r/w	1'd0	1:force adc ch0 fir output
19:0	adc_fir_4s_val	r/w	20'd0	force value of adc fir output

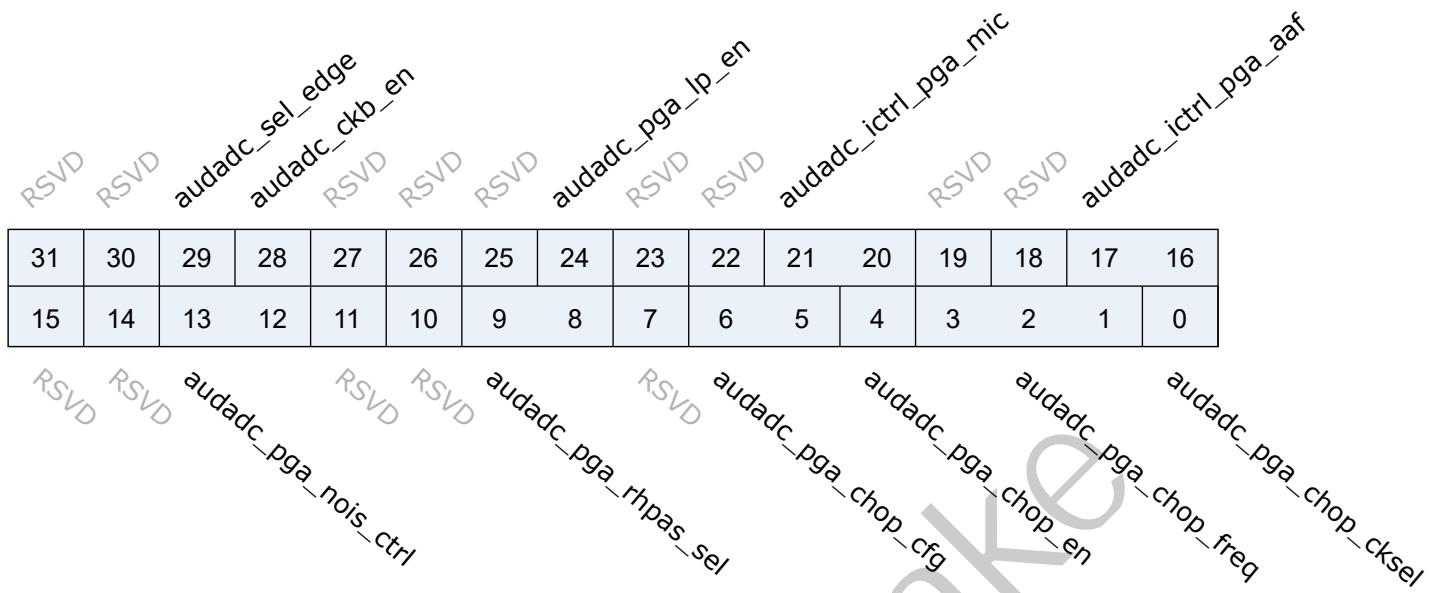
## 16.6.10 pdm\_adc\_s0

地址: 0x2000ac38

位	名称	权限	复位值	描述
31:9	RSVD			
8:0	adc_s0_volume	r/w	9'd0	volume s9.1, -95.5dB +18dB in 0.5dB step

### 16.6.11 audadc\_ana\_cfg1

地址: 0x2000ac60



位	名称	权限	复位值	描述
31:30	RSVD			
29	audadc_sel_edge	r/w	1'h0	ADC output data clock edge 0 = falling edge sent, rising edge receive 1 = rising edge sent, falling edge receive
28	audadc_ckb_en	r/w	1'h0	AUDADC clock phase control 0 = 0° 1 = 180°
27:25	RSVD			
24	audadc_pga_lp_en	r/w	1'h0	PGA lowpower function reversed, not realized in circuit
23:22	RSVD			
21:20	audadc_ictrl_pga_mic	r/w	2'h1	PGA_OPMIC bias current control 00 = 4uA 01 = 5uA 10 = 6uA 11 = 7uA
19:18	RSVD			
17:16	audadc_ictrl_pga_aaf	r/w	2'h1	PGA_OPAAF bias current control 00 = 4uA 01 = 5uA 10 = 6uA 11 = 7uA
15:14	RSVD			
13:12	audadc_pga_nois_ctrl	r/w	2'h0	PGA noise control when configured to single-ended not used

位	名称	权限	复位值	描述
11:10	RSVD			
9:8	audadc_pga_rhpas_sel	r/w	2'h0	PGA high pass filter R control when configured to AC-coupled mode 00 = 480kΩ 01 = 320kΩ 10 = 160kΩ 11 = 4kΩ, fast startup
7	RSVD			
6:5	audadc_pga_chop_cfg	r/w	2'h3	control chopper for opmic&opaaf 00 = opmic off & opaaf off 01 = opmic off & opaaf on 10 = opmic on & opaaf off 11 = opmic on & opaaf on
4	audadc_pga_chop_en	r/w	1'h1	PGA chopper control 0 = disable 1 = enable
3:1	audadc_pga_chop_freq	r/w	3'h4	PGA chopper frequency control @Fs=2048k 000 = 8k 001 = 16k 010 = 32k 011 = 64k 100 = 128k 101 = 256k 110 = 512k 111 = 1024k
0	audadc_pga_chop_cksel	r/w	1'h0	PGA chopper clock source selection 0 = adc clock 1 = synchronized clock from SDM not used

### 16.6.12 audadc\_ana\_cfg2

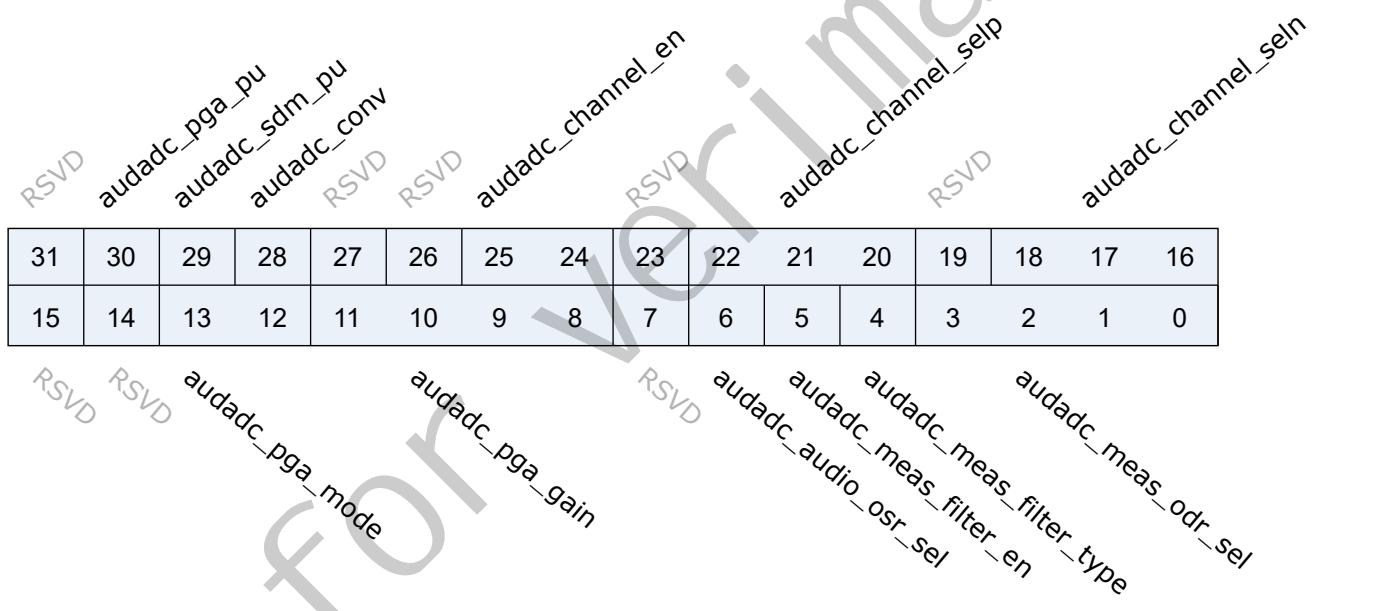
地址: 0x2000ac64

位	名称	权限	复位值	描述
31:30	RSVD			
29:28	audadc_reserved	r/w	2'h0	AUDADC reserved register
27:25	RSVD			
24	audadc_sdm_lp_en	r/w	1'h0	SDM lowpower funciton 0 = disable 1 = enable, 0.6 of disable
23:22	RSVD			
21:20	audadc_ictrl_adc	r/w	2'h1	SDM bias current control 00 = 4uA 01 = 5uA 10 = 6uA 11 = 7uA
19	RSVD			
18:16	audadc_nctrl_adc1	r/w	3'h3	op number control for first integrator in SDM 000 = 1(12uA) 001 = 2(24uA) 010 = 3(36uA) 011 = 4(48uA) 100 = 5(60uA) 101 = 5(60uA) 110 = 5(60uA) 111 = 5(60uA)
15:14	RSVD			
13:12	audadc_nctrl_adc2	r/w	2'h1	op number control for second integrator in SDM 00 = 1(12uA) 01 = 2(24uA) 10 = 3(36uA) 11 = 3(36uA)
11:9	RSVD			
8	audadc_dem_en	r/w	1'h1	dem function control 0 = disable 1 = enable
7:6	RSVD			

位	名称	权限	复位值	描述
5:4	audadc_quan_gain	r/w	2'h1	quantizer gain control for SDM 00 = Vref/14 01 = Vref/12 10 = Vref/10 11 = Vref/8
3	audadc_dither_ena	r/w	1'h1	dither control 0 = disable 1 = enable
2:1	audadc_dither_sel	r/w	2'h2	dither level control for SDM 00 = 0 01 = LSB*1/15 10 = LSB*2/15 11 = LSB*3/15
0	audadc_dither_order	r/w	1'h0	dither order control for SDM 0 = 0 order 1 = 1 order

### 16.6.13 audadc\_cmd

地址: 0x2000ac68



位	名称	权限	复位值	描述
31	RSVD			
30	audadc_pga_pu	r/w	1'h0	PGA related circuit enable 0 = disable 1 = enable
29	audadc_sdm_pu	r/w	1'h0	SDM related circuit enable 0 = disable 1 = enable
28	audadc_conv	RSVD		
27	audadc_channel_en	RSVD		
26	audadc_channel_sel	RSVD		
25	audadc_channel_sel	RSVD		
24	audadc_channel_sel	RSVD		
23	audadc_channel_sel	RSVD		
22	audadc_meas_filter_en	RSVD		
21	audadc_meas_filter_type	RSVD		
20	audadc_meas_odr_sel	RSVD		
19	audadc_audio_osr_sel	RSVD		
18	audadc_pga_mode	RSVD		
17	audadc_pga_gain	RSVD		
16	audadc_meas_filter_en	RSVD		
15	audadc_meas_odr_sel	RSVD		
14	audadc_audio_osr_sel	RSVD		
13	audadc_pga_mode	RSVD		
12	audadc_pga_gain	RSVD		

位	名称	权限	复位值	描述
28	audadc_conv	r/w	1'h0	SDM conversion start singal 0 = remain resetting status 1 = start conversion both analog intetrator and measuring digital decimation filter will be reset when audadc_conv configured to low. Measuring digital decimation filter need to reset to same initial condition because it's feedback configuration for FIR. Audio filter dont need this resetting.
27:26	RSVD			
25:24	audadc_channel_en	r/w	2'h0	channel mux switch enable or disable MSB controls Positive channel, LSB controls Negative channel 0 = disable, look into each channel will see high impedance 1 = enable, one of eight channel will be choose
23	RSVD			
22:20	audadc_channel_selp	r/w	3'h0	Positive channel selection, connected to PGA positive terminal 000 = AIN0 001 = AIN1 010 = AIN2 011 = AIN3 100 = AIN4 101 = AIN5 110 = AIN6 111 = AIN7
19	RSVD			
18:16	audadc_channel_seln	r/w	3'h0	Negative channel selection, connected to PGA negative terminal 000 = AIN0 001 = AIN1 010 = AIN2 011 = AIN3 100 = AIN4 101 = AIN5 110 = AIN6 111 = AIN7
15:14	RSVD			
13:12	audadc_pga_mode	r/w	2'h0	PGA mode configuration 00: AC-Coupled & differential-ended, Audio application 01: AC-Coupled & single-ended, Audio application 10: DC-Coupled & differential-ended, Measuring application 11: DC-Coupled & single-ended, Measuring application(may not used)

位	名称	权限	复位值	描述
11:8	audadc_pga_gain	r/w	4'h0	PGA Gain control 0000 = 6dB 0001 = 6dB 0010 = 6dB 0011 = 9dB 0100 = 12dB 0101 = 15dB 0110 = 18dB 0111 = 21dB 1000 = 24dB 1001 = 27dB 1010 = 30dB 1011 = 33dB 1100 = 36dB 1101 = 39dB 1110 = 42dB 1111 = 42dB
7	RSVD			
6	audadc_audio_osr_sel	r/w	1'h0	audio osr configuration 0 = 128 1 = 64
5	audadc_meas_filter_en	r/w	1'h0	measuring mode enable, measuring filter is on when set to high 0 = disable 1 = enable
4	audadc_meas_filter_type	r/w	1'h0	digital dicimation filter selection when in measuring mode 0 = SINC3 1 = Low-Latency
3:0	audadc_meas_odr_sel	r/w	4'h3	audadc ouput data rate selection when configured to measuring mode 0000 = 2.5SPS 0001 = 5SPS 0010 = 10SPS 0011 = 20SPS 0100 = 25SPS 0101 = 50SPS 0110 = 100SPS 0111 = 200SPS 1000 = 400SPS 1001 = 800SPS 1010 = 1000SPS 1011 = 2000SPS 1100 = 4000SPS 1101 = 4000SPS 1110 = 4000SPS 1111 = 4000SPS

### 16.6.14 audadc\_data

地址: 0x2000ac6c

audadc_valid_4s_en	audadc_valid_4s_val	audadc_soft_rst	RSVD	RSVD	RSVD	RSVD	audadc_data_rdy	audadc_raw_data
31	30	29	28	27	26	25	24	23
22	21	20	19	18	17	16	15	14

## audadc raw data

位	名称	权限	复位值	描述
31	audadc_valid_4s_en	r/w	1'h0	
30	audadc_valid_4s_val	r/w	1'h0	
29	audadc_soft_rst	r/w	1'h0	
28:25	RSVD			
24	audadc_data_rdy	r	1'h0	audadc data ready indicator when measuring mode selected, auto reset to 0 after read 0 = not ready 1 = ready
23:0	audadc_raw_data	r	16'h0	audadc output 16bit data, 2's

## 16.6.15 audadc rx fifo ctrl

地址: 0x2000ac80

<i>RSVD</i>	<i>RSVD</i>	<i>RSVD</i>	<i>RSVD</i>	<i>RSVD</i>	<i>RSVD</i>	<i>rx_data_mode</i>	<i>RSVD</i>	<i>RSVD</i>	<i>RSVD</i>	<i>RSVD</i>	<i>rx_trg_level</i>				
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

位	名称	权限	复位值	描述
31:26	RSVD			

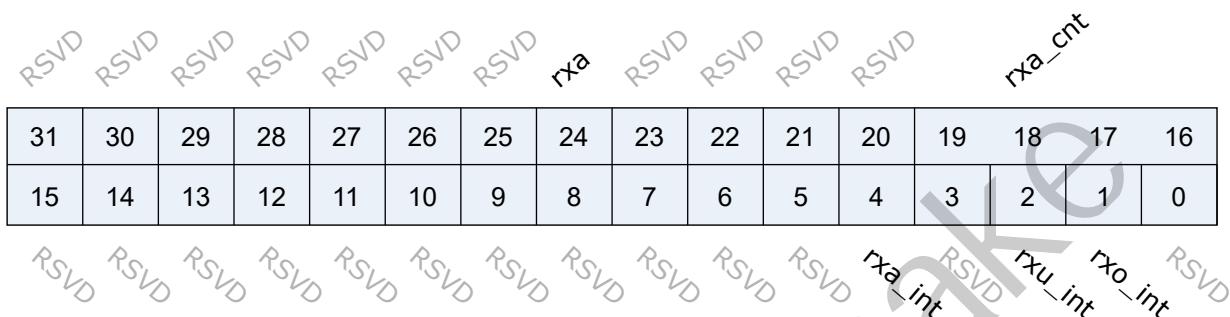
位	名称	权限	复位值	描述
25:24	rx_data_mode	r/w	2'b0	<p>RX_FIFO_DATOUT_MODE.</p> <p>RX FIFO DATA Output Mode (Mode 0, 1, 2, 3)</p> <p>Mode 0: Valid data's MSB is at [31] of RX_FIFO register</p> <p>Mode 1: Valid data's MSB is at [23] of RX_FIFO register</p> <p>Mode 2: Valid data's MSB is at [19] of RX_FIFO register</p> <p>Mode 3: Valid data's MSB is at [15] of RX_FIFO register</p> <p>Note: Expanding ‘0’ at LSB of RX FIFO register (data invalid region)</p> <p>Expanding sign bit at MSB of RX FIFO register (data invalid region)</p> <p>For 24-bit received audio sample resolution:</p> <ul style="list-style-type: none"> <li>Mode 0: RXDATA[31:0] = FIFO_O[23:0], 8' h0</li> <li>Mode 1: RXDATA[31:0] = 8FIFO_O[23], FIFO_O[23:0]</li> <li>Mode 2: RXDATA[31:0] = 12FIFO_O[23], FIFO_O[23:4]</li> <li>Mode 3: RXDATA[31:0] = 16FIFO_O[23], FIFO_O[23:8]</li> </ul> <p>For 20-bit received audio sample resolution:</p> <ul style="list-style-type: none"> <li>Mode 0: RXDATA[31:0] = FIFO_O[23:4], 12' h0</li> <li>Mode 1: RXDATA[31:0] = 8FIFO_O[23], FIFO_O[23:4], 4' h0</li> <li>Mode 2: RXDATA[31:0] = 12FIFO_O[23], FIFO_O[23:4]</li> <li>Mode 3: RXDATA[31:0] = 16FIFO_O[23], FIFO_O[23:8]</li> </ul> <p>For 16-bit received audio sample resolution:</p> <ul style="list-style-type: none"> <li>Mode 0: RXDATA[31:0] = FIFO_O[23:8], 16' h0</li> <li>Mode 1: RXDATA[31:0] = 8FIFO_O[23], FIFO_O[23:8], 8' h0</li> <li>Mode 2: RXDATA[31:0] = 12FIFO_O[23], FIFO_O[23:8], 4'h0</li> <li>Mode 3: RXDATA[31:0] = 16FIFO_O[23], FIFO_O[23:8]</li> </ul>
23:20	RSVD			
19:16	rx_trg_level	r/w	4'd3	<p>RX_FIFO_TRG_LEVEL.</p> <p>RX FIFO Trigger Level (RXTL[3:0])</p> <p>Interrupt and DMA request trigger level for RX FIFO Data Available condition</p> <p>IRQ/DRQ Generated when WLEVEL &gt; RXTL[3:0]</p> <p>Notes:</p> <p>WLEVEL represents the number of valid samples in the RX FIFO</p>

位	名称	权限	复位值	描述
15:14	rx_drq_cnt	r/w	2'b0	<p>RX_DRQ_CLR_CNT.</p> <p>When RX FIFO available data less than or equal N, DRQ Request will be de-asserted. N is defined here:</p> <ul style="list-style-type: none"> <li>00: IRQ/DRQ de-asserted when WLEVEL &lt;= RXTL[3:0]</li> <li>01: IRQ/DRQ de-asserted when WLEVEL &lt; 1</li> <li>10: IRQ/DRQ de-asserted when WLEVEL &lt; 2</li> <li>11: IRQ/DRQ de-asserted when WLEVEL &lt; 4</li> </ul> <p>WLEVEL represents the number of valid samples in the RX FIFO</p>
13:9	RSVD			
8	rx_ch_en	r/w	1'b0	<p>RX_FIFO_DATIN_SRC.</p> <p>RX FIFO Data Input Source Select.</p> <p>0: Disable 1: Enable</p> <p>Bit8: ADC1 data</p>
7	RSVD			
6:5	rx_data_res	r/w	2'd0	<p>RX_SAMPLE_BITS.</p> <p>Receiving Audio Sample Resolution</p> <p>0: 16 bits</p> <p>1: 20 bits</p> <p>2: 24 bits</p> <p>3: Reserved</p>
4	rx_drq_en	r/w	1'b0	<p>ADC_DRQ_EN.</p> <p>ADC FIFO Data Available DRQ Enable.</p> <p>0: Disable</p> <p>1: Enable</p>
3	rxu_int_en	r/w	1'b0	<p>ADC_IRQ_EN.</p> <p>ADC FIFO Data Available IRQ Enable.</p> <p>0: Disable</p> <p>1: Enable</p>
2	rxo_int_en	r/w	1'b0	<p>ADC_UNDERRUN_IRQ_EN.</p> <p>ADC FIFO Under Run IRQ Enable</p> <p>0: Disable</p> <p>1: Enable</p>
1	rxo_int_en	r/w	1'b0	<p>ADC_OVERRUN_IRQ_EN.</p> <p>ADC FIFO Over Run IRQ Enable</p> <p>0: Disable</p> <p>1: Enable</p>

位	名称	权限	复位值	描述
0	rx_fifo_flush	w1c	1'b0	ADC_FIFO_FLUSH. ADC FIFO Flush. Write ‘1’ to flush TX FIFO, self clear to ‘0’.

### 16.6.16 audadc\_rx\_fifo\_status

地址: 0x2000ac84



位	名称	权限	复位值	描述
31:25	RSVD			
24	rxa	r	1'b0	RXA. RX FIFO Available 0: No available data in RX FIFO 1: More than one sample in RX FIFO (>= 1 word)
23:20	RSVD			
19:16	rxa_cnt	r	4'h0	RXA_CNT. RX FIFO Available Sample Word Counter
15:5	RSVD			
4	rxa_int	r	1'b0	RXA_INT. RX FIFO Data Available Pending Interrupt 0: No Pending IRQ 1: Data Available Pending IRQ Automatic clear if interrupt condition fails.
3	RSVD			
2	rxu_int	r	1'b0	RXU_INT. RX FIFO Underrun Pending Interrupt 0: No Pending IRQ 1: FIFO Underrun Pending IRQ Write ‘1’ to clear this interrupt

位	名称	权限	复位值	描述
1	rxo_int	r	1'b0	RXO_INT. RX FIFO Overrun Pending Interrupt 0: No Pending IRQ 1: FIFO Overrun Pending IRQ Write ‘1’ to clear this interrupt
0	RSVD			

### 16.6.17 audadc\_rx\_fifo\_data

地址: 0x2000ac88

rx\_data

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\_data

位	名称	权限	复位值	描述
31:0	rx_data	r	32'h0	RX_DATA. RX Sample Host can get one sample by reading this register. The left channel sample data is first and then the right channel sample.

## 17.1 简介

EMAC 模块是一个兼容 IEEE 802.3 的 10/100Mbps 以太网 MAC(Ethernet Media Access Controller)。其包含状态及控制寄存器组，收发模块，收发缓冲描述符组，主机接口，MDIO 接口，物理层芯片 (PHY) 接口。

状态及控制寄存器组包含了 EMAC 的状态位及控制位，是与用户程序的接口，负责控制数据收发，并汇报状态。

收发模块负责根据收发描述符内的控制字，从指定内存位置取得数据帧，添加前导，CRC，并扩充短的帧后通过 PHY 发出；或是从 PHY 接收数据，并根据收发缓冲描述符，将数据放入指定内存。收发完成后设置相关的事件标志。如果使能了事件中断，将通过中断请求到主机进行处理。

MDIO 及 MII/RMII 接口负责与 PHY 进行通信，包括读写 PHY 的寄存器，以及数据包的收发。

## 17.2 主要特征

- 兼容 IEEE 802.3 定义的 MAC 层功能
- 支持 IEEE 802.3 定义的 MII/RMII 接口的 PHY
- 通过 MDIO 接口与 PHY 交互
- 支持 10Mbps 与 100Mbps 以太网
- 支持半双工与全双工
- 在全双工模式下，支持自动流控及生成控制帧
- 在半双工模式下，支持碰撞检测及重传
- 支持 CRC 的生成及校验
- 数据帧前导生成及移除
- 发送时，自动扩展短的数据帧
- 检测过长或过短的数据帧 (长度限制)

- 可传输长数据帧 (> 标准以太帧长度)
- 自动丢弃重发次数超限或帧间隙过小的数据包
- 广播包过滤
- 用于保存多达 128 个 BD(Buffer Descriptor) 的内部 RAM
- 在发送时，支持将一个数据包分拆配置到多个连续的 BD
- 发送/接收的各种事件标志
- 在事件发生时产生对应中断

## 17.3 功能描述

EMAC 模块的组成如下图。

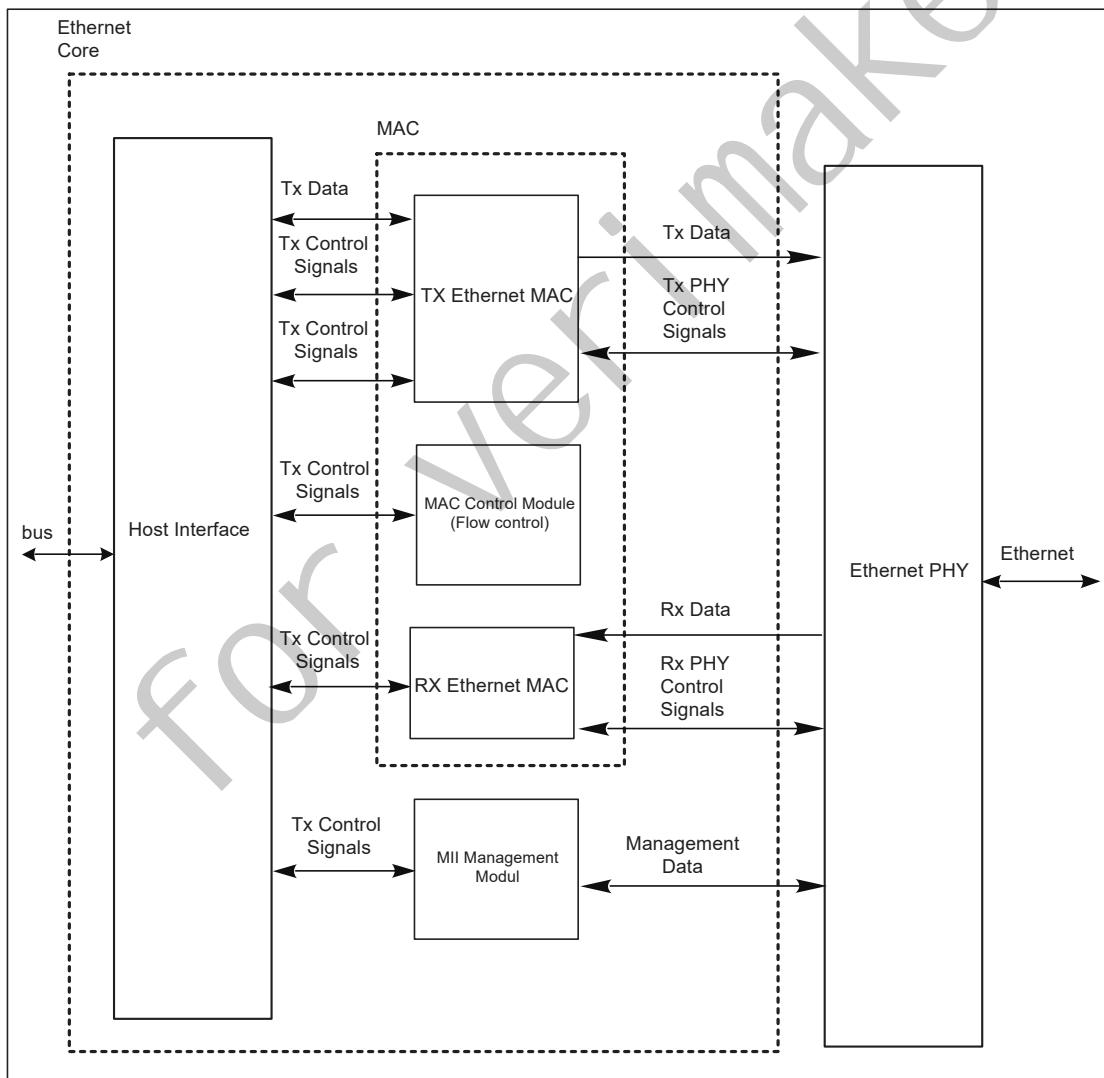


图 17.1: EMAC 框图

模块的控制寄存器通过 MDIO 接口，可以读写 PHY 的寄存器，从而实现配置、选择模式(半/全双工)、发起协商等操作。接收模块过滤并检查收到的数据帧：是否有合法的前导，FCS，长度等，并根据缓冲描述符(BD)，将数据存放到指定内存地址。发送模块根据数据缓冲描述符，从内存中取得数据，添加前导，FCS，pad 等，然后根据 CSMA/CD 协议，将数据发出。如果检测到 CRS，将会延迟重试。收发缓冲描述符组连接到系统 RAM，此 RAM 用于保存发送和接收的以太网数据帧。每个描述符包含相应的控制状态字以及对应的缓冲内存地址。描述符一共有 128 组，用于发送或者接收，可以灵活分配。

## 17.4 时钟

EMAC 模块需要一路时钟用于同步收发(100Mbps 时，25MHz(MII) 或 50MHz(RMII); 10Mbps 时，2.5MHz)。此时钟必须在 EMAC 与 PHY 之间同步。

## 17.5 收发缓冲描述符(BD, Buffer Descriptor)

收发缓冲描述符，用于提供 EAMC 与数据帧缓存地址信息之间的关联，对收发数据帧进行控制，以及提供收发状态提示。每个描述符由两个连续的 word(32bit) 构成，低地址的 word0 提供了本 buffer 包含的数据帧的长度，控制及状态位；高地址的 word1 是对应的数据帧缓存的 32 位内存地址(指针)。

发送 BD 中的 word0 具体描述：

[31:16]: 发送数据包长度(LEN)。

[15]: 发送 BD 准备好标志(RD)。软件写 1 告知 EMAC 此 BD 包含需要发送的数据，硬件写 0 说明此 BD 数据已发送完成或有错误发生。

[14]: 中断使能标志位(IRQ)。置位则此 BD 可以申请 TXE 或 TXB 中断。

[13]: 回绕标记(WR)。置位标志着此 BD 是最后一个发送 BD，硬件将从起始 BD 开始重新循环发送。

[12]: 填充标志(PAD)。若置位，则在 EMAC 中设置了填充允许的话，发送 BD 将会自动填充过短的数据包。

[11]: 校验标志(CRC)。若置位，则 EMAC 会自动计算发送数据包的 CRC，并附在数据包中。

[10]: 帧结束标志(EoF)。如果一帧数据占用多个 BD，则此位标志着这帧数据的结束。

[8]: underfun 标志(UR)。置位则说明 BD 的发送过程中发生了 FIFO underrun 的错误。

[7:4]: 重发次数计数器(RTRY)。记录了重发次数。

[3]: 重发次数超限标志(RL)。置位则表明重发次数超过了 COLLCOMF 中配置的最大重发次数 MAXRET。

[2]: Late Collision 标志(LC)。置位则表明在发送此 BD 时发生了 Late Collision。

[1]: Defer Indication 标志(DF)。置位则表明此包被延迟发送。

[0]: 载波检测失效(CS)。如果在发送过程中检测不到载波，则置位。

接收 BD 中的 word0 具体描述：

[31:16]: 发送数据包长度(LEN)。

[15]: 接收 BD 空标志 (RD)。置位说明此 BD 为空 (没保存收到的数据), 清零说明此 BD 有接收到的数据, 或接收过程中有错误发生。

[14]: 中断使能标志位 (IRQ)。置位则此 BD 可以申请 RXE 或 RXB 中断。

[13]: 回绕标记 (WR)。置位标志着此 BD 是最后一个接收 BD, 硬件将从接收起始 BD 开始重新循环发送。

[8]: 控制帧标志 (CF)。置位则说明此 BD 接收到了一个 Control Frame。

[7]: miss 标志 (M)。若在混杂模式收到了数据包, 但被内部地址逻辑标记为 miss, 则 EMAC 设置此标志位。

[6]: overrun 标志 (OR)。置位则说明接收过程中发生了 FIFO overrun 错误。

[5]: 接收错误标志 (RE)。置位则说明接收过程中收到了 PHY 发出的 RX ERR 信号。

[4]: Dribble Nibble 标志 (DN)。置位则说明接收到了奇数个 nibbles。

[3]: 数据包过长标志 (TL)。置位则说明接收的数据包太长, 超过了设置。

[2]: 数据表过短标志 (SF)。置位则表明接收到的数据包长度小于最小的允许值。

[1]: CRC 错误标志 (CRC)。置位则表明接收到的数据包 CRC 校验失败。

[0]: Late Collision 标志 (LC)。置位则表明在接收数据到此 BD 时发生了 Late Collision。

需要注意的是: 对于 BD, 需要按 word 写入。

EMAC 模块支持 128 个 BD, 由发送/接收逻辑共享, 可自由组合。但发送 BD 总是占据前面的连续区域 (个数由 MAC\_TX\_BD\_NUM 寄存器中的 TXBDNUM 域来指定)。

EMAC 按照 BD 的顺序, 循环处理发送/接收 BD, 直到遇到标记为 WR 的 BD 就回绕到发送/接收各自的首个 BD。

## 17.6 PHY 交互

PHY 交互寄存器组提供了与 PHY 交互需要的命令及数据通信的方式。EMAC 通过 MDIO 接口控制 PHY 的工作模式, 并保证两者的工作模式匹配 (速率, 全/半双工等)。

数据包通过 MII/RMII 接口在 EMAC 与 PHY 之间交互, 可以通过 EMAC 的模式寄存器 (EMAC\_MODE) 中的 RMII\_EN 位选择: 当此 bit 为 1, 则选择 RMII 模式, 否则就是 MII 模式。

MII 及 RMII 模式均支持 IEEE 802.3u 标准中指定的 10Mbps 与 100Mbps 的传输速率。

MII 及 RMII 的传输信号描述与下表。

表 17.1: 传输信号

名称	MII	RMII
EXTCK_EREFCK	ETXCK: 发送时钟信号	EREFCK: 参考时钟
ECRS	ECRS: 载波探测	-
ECOL	ECOL: 碰撞检测	-

表 17.1: 传输信号 (continued)

名称	MII	RMII
ERXDV	ERXDV: 数据 valid	ECRSDV: 载波检测/数据 valid
ERX0-ERX3	ERX0-ERX3: 4-bit 接收数据	ERX0-ERX1: 2-bit 接收数据
ERXER	ERXER: 接收错误指示	ERXER: 接收错误指示
ERXCK	ERXCK: 接收时钟信号	-
ETXEN	ETXEN: 发送使能	ETXEN: 发送使能
ETX0-ETX3	ETX0-ETX3: 4-bit 发送数据	ETX0-ETX1: 2-bit 发送数据
ETXER	ETXER: 发送错误指示	-
EMDC	MDIO Clock	MDIO Clock
EMDIO	MDIO Data Input Output	MDIO Data Input Output

RMII 接口引脚较少，使用 2-bit 数据线用于收发，在 100Mbps 速率时，需要提供 50MHz 的参考时钟。

## 17.7 编程流程

### 17.7.1 PHY 初始化

- 根据 PHY 类型，设置 EMAC\_MODE 寄存器中的 RMII\_EN 位来选择合适的连接方式
- 设置 EMAC 的 MAC 地址到 EMAC\_MAC\_ADDR0 与 EMAC\_MAC\_ADDR1 中
- 通过编程 EMAC\_MIIMODE 寄存器中的域 CLKDIV，为 MDIO 部分设置合适的时钟
- 设置对应 PHY 的地址到寄存器 EMAC\_MIIADDRESS 的域 FIAD 中
- 根据 PHY 的手册，通过 EMAC\_MIICOMMAND 与 EMAC\_MIITX\_DATA 寄存器发送命令
- 读取 PHY 的数据会保存在 EMAC\_MIIRX\_DATA 寄存器中
- 通过 EMAC\_MIISTATUS 寄存器可以查询与 PHY 命令交互的状态

基础的交互完成后，应当使 PHY 进入自动协商状态。协商完成之后，根据协商结果编程模式到 EMAC\_MODE 寄存器中的 FULLD 位。

## 17.7.2 发送数据帧

- 配置 EMAC\_MODE 寄存器中数据帧格式、间隔等位域
- 通过配置 EMAC\_TX\_BD\_NUM 寄存器中的 TXBDNUM 域来指定发送所使用的 BD 的个数，那么剩余的就是 RX 的 BD
- 在内存中准备好需要发送的数据帧
- 将数据帧的地址填写到对应发送 BD 的数据指针域 (word 1) 中
- 清空对应发送 BD 的控制与状态域 (wrod 0) 中的状态标记，并设置控制域 (CRC 使能, PAD 使能, 中断使能等)
- 写入数据帧长度，并设置好 RD 域，告知 EMAC 此 BD 数据需要发送；如需要，设置上 IRQ 位，以使能中断
- 特别的，如果是最后一个发送的 BD，需要设置上 WR 位，EMAC 会在处理完这个 BD 之后“回绕”到第一个发送 BD 进行处理
- 如果有多个 BD 需要发送，则重复设置 BD 的步骤以填充所有的发送 BD
- 如果一个数据包只包含在一个 BD 中，那么需要设置其 EOF 位为 1
- 如果一个数据包分在多个 BD 里进行发送，那么只需要将其占用的最后一个 BD 标记为数据包结束 (设置 EOF 位)
- 如果需要使能发送中断，还需要配置 EMAC\_INT\_MASK 寄存器中的 TX 相关位
- 配置 EMAC\_MODE 寄存器中的 TXEN 位，以使能发送
- 如果使能了中断，在发送的中断中，可用通过 EMAC\_TX\_BD\_NUM 寄存器中的 TXBDNUM 域获取当前的 BD
- 根据当前 BD 的状态字进行相应的处理
- 数据已被发送出去的 BD，其控制域中的 RD 位会被硬件清零，且不会被再次发送；需要填充新数据后，置位 RD，此 BD 即可再次用于发送

## 17.7.3 接收数据帧

- 配置 EMAC\_MODE 寄存器中数据帧格式、间隔等位域
- 通过配置 EMAC\_TX\_BD\_NUM 寄存器中的 TXBDNUM 域来指定发送所使用的 BD 的个数，那么剩余的就是 RX 的 BD
- 在内存中准备好接收数据的区域
- 将数据帧的地址填写到对应接收 BD 的数据指针域 (word 1) 中
- 清空对应发送 BD 的控制与状态域 (wrod 0) 中的状态标记，并设置控制域 (中断使能等)
- 写入可接收的数据帧长度，并设置好 E 位域，告知 EMAC 此 BD 空闲，可以用于数据接收；如需要，设置上 IRQ 位，以使能中断
- 特别的，如果是最后一个有效接收 BD，需要设置上 WR 位，EMAC 会在处理完这个 BD 之后“回绕”到第一个接收

## BD 进行处理

- 如果有多个 BD 可供接收数据，则重复设置 BD 的步骤以填充所有的 BD
- 如果需要使能接收中断，还需要配置 EMAC\_INT\_MASK 寄存器中的 RX 相关位
- 配置 EMAC\_MODE 寄存器中的 RXEN 位，以使能接收
- 如果使能了中断，在接收的中断中，可用通过 EMAC\_TX\_BD\_NUM 寄存器中的 RXBDNUM 域获取当前的 BD
- 根据当前 BD 的状态字进行相应的处理
- 接收完成的 BD，其控制域中的 E 位会被硬件清零，且不会被再次用于接收；需要取走数据，置位 E，此 BD 即可再次用于接收

## 18.1 简介

USB(Universal Serial Bus)通用串行总线,是一个外部总线标准,用于规范电脑与外部设备的连接和通讯。BL616/BL618 支持 USB2.0 (HighSpeed + FullSpeed), 可作为主机控制器 (Host) 或者外围控制器 (Device)。作为主机控制器时, 它包含一个支持所有速度事务的 USB 主机控制器。在没有软件干预的情况下, 主机控制器可以自行处理基于事务的数据结构以减轻 CPU 的负载并自动在 USB 总线上发送和接收数据。当作为外围控制器时, 除端点 0 外的每个端点都支持 USB 规范的传输类型, 以适配各种应用类型。此外, BL616/BL618 的 USB 控制器符合 OTG 标准, 支持会话请求协议 (SRP) 和主机协商协议 (HNP)。

## 18.2 主要特征

- 兼容 USB2.0 (HighSpeed + FullSpeed)
- 兼容 OTG revision1.3
- 支持 UTMI+ Level 3
- 支持 OTG SRP 和 HNP 协议
- 支持 Host、OTG、Device 模式
- 支持 LPM
- 兼容 EHCI 数据结构 (不支持 FSTN 和 SITD)
- 支持 9 个端点 (EP0: control endpoint EP1-EP8: interrupt/isolation/bulk endpoint)
- 支持 1 个控制传输专用 FIFO 和 4 个一般 FIFO (控制 FIFO: 64 字节一般 FIFO: 512 字节)
- 支持双 FIFO、三 FIFO 模式工作
- 支持 DMA
- 支持 VDMA

## 18.3 功能描述

### 18.3.1 USB 使用步骤

1. 配置内部 transceiver
2. 配置 usb controller
3. 配置 usb 中断
4. 配置 usb dma/vdma (不支持 cpu 直接读写 fifo)
5. 完成

### 18.3.2 部分寄存器配置及功能描述

### 18.3.3 USB 枚举阶段中断处理流程

## 19.1 简介

ISO11898 是由以研发和生产汽车电子产品著称的德国 BOSCH 公司开发的，是国际上应用最广泛的现场总线之一。

## 19.2 主要特征

- 支持 1Mbps 和自定义位速率
- 支持 ISO11898 协议 2.0A 和 2.0B
- 支持自测模式（自发自收）
- 支持帧过滤
- 支持静默模式（不应答，没有有效的错误标志）
- 支持不重发的单次传输
- 支持查询仲裁失败的 bit 位
- 任何总线错误都可以触发中断

## 19.3 功能介绍

### 19.3.1 发送缓冲区 (TXB)

发送缓冲区是 CPU 和位流处理器之间的接口，能够存储完整的消息，以便通过 ISO11898 网络传输。缓冲区长度为 13 字节，由 CPU 写入，由位流处理器读取。

### 19.3.2 接收缓冲区（RXB, RXFIFO）

接收缓冲区是接收滤波器和 CPU 之间的接口，用于储存从 ISO11898 总线接收并且经过滤波的消息。接收缓冲区（RXB）表示接收 FIFO（RXFIFO）中可由 CPU 访问的 13 字节窗口，其总长度为 64 字节。接收 FIFO 使得 CPU 可以在处理一帧消息的同时接收其他消息。

### 19.3.3 接收过滤器（ACF）

接收过滤器将收到的标识符和接收过滤寄存器中的内容进行比较，并决定是否应该接受此消息。如果接受此消息，则将完整的消息储存在 RXFIFO 中。

### 19.3.4 位流处理器（BSP）

位流处理器是一个序列发生器，用于处理发送缓冲区、接收缓冲区和 ISO11898 总线之间的数据。它还在 ISO11898 总线上执行错误侦测、仲裁、位填充和错误处理。

### 19.3.5 位时序逻辑（BTL）

位时序逻辑监控串行 ISO11898 总线并处理与总线相关的位时序。它在消息开始时总线由“隐形到显性”的转换时进行位同步（硬同步），也在后续的消息接收期间再次进行位同步（软同步）。BTL 还提供可编程的时间段用来补偿传播延时与相移（例如由于振荡器漂移），并可定义采样点和在一个位时间内的采样次数。

### 19.3.6 错误管理逻辑（EML）

错误管理逻辑负责传输层模块的错误界定。它从 BSP 接收错误声明然后将错误的统计信息通知给 BSP 和 IML（中断管理逻辑）

## 19.4 功能描述

### 19.4.1 模式

#### 19.4.1.1 自测模式

通过对 MOD 寄存器的 STM 位置‘1’来选择自测模式。在这种模式下，可以使用接收请求命令在总线上无其他活跃节点的情况下进行全节点测试，并且即使没有收到应答 ISO11898 控制器也将执行成功的传输。

### 19.4.1.2 静默模式

通过对 MOD 寄存器的 LOM 位置'1' 来选择进入静默模式。在这种模式下，ISO11898 控制器即使成功收到消息也不会对 ISO11898 总线做出应答，并且错误计数器也会停留在当前值。这种操作模式会强制 ISO11898 控制器成为被动错误，此时也不可以传输消息。在软件驱动的位速率检测和热插拔等场景中可以使用这种静默模式，其他所有功能都可以和正常模式一样使用。

### 19.4.1.3 复位模式

MOD 寄存器中的 RM 位一旦由'0' 变成'1'，将导致当前的发送和接收消息都被终止并进入复位模式。当 RM 位从'1' 到'0' 转变时，ISO11898 控制器将返回到操作模式。

在不同模式下的不同操作含义如下表所示。

ADDRESS OFFSET	OPERATING MODE			RESET MODE	
	READ	WRITE		READ	WRITE
0x00	mode	mode		mode	mode
0x04	(00H)	command		(00H)	command
0x08	status	reserved		status	reserved
0x0C	interrupt	reserved		interrupt	reserved
0x10	interrupt enable	interrupt enable		interrupt enable	interrupt enable
0x14	reserved	reserved		reserved	reserved
0x18	bus timing 0	reserved		bus timing 0	bus timing 0
0x1C	bus timing 1	reserved		bus timing 1	bus timing 1
0x20	reserved	reserved		reserved	reserved
0x24	reserved	reserved		reserved	reserved
0x28	reserved	reserved		reserved	reserved
0x2C	arbitration lost capture	reserved		arbitration lost capture	reserved
0x30	error code capture	reserved		error code capture	reserved
0x34	error warning limit	reserved		error warning limit	error warning limit
0x38	RX error counter	reserved		RX error counter	RX error counter
0x3C	TX error counter	reserved		TX error counter	TX error counter
0x40	SFF RX frame information	EFF RX frame information	SFF TX frame information	EFF TX frame information	acceptance code 0
0x44	RX identifier 1	RX identifier 1	TX identifier 1	TX identifier 1	acceptance code 1

0x48	RX identifier 2	RX identifier 2	TX identifier 2	TX identifier 2	acceptance code 2	acceptance code 2
0x4C	RX data 1	RX identifier 3	TX data 1	TX identifier 3	acceptance code 3	acceptance code 3
0x50	RX data 2	RX identifier 4	TX data 2	TX identifier 4	acceptance mask 0	acceptance mask 0
0x54	RX data 3	RX data 1	TX data 3	TX data 1	acceptance mask 1	acceptance mask 1
0x58	RX data 4	RX data 2	TX data 4	TX data 2	acceptance mask 2	acceptance mask 2
0x5C	RX data 5	RX data 3	TX data 5	TX data 3	acceptance mask 3	acceptance mask 3
0x60	RX data 6	RX data 4	TX data 6	TX data 4	reserved	reserved
0x64	RX data 7	RX data 5	TX data 7	TX data 5	reserved	reserved
0x68	RX data 8	RX data 6	TX data 8	TX data 6	reserved	reserved
0x6C	(FIFO RAM)	RX data 7	reserved	TX data 7	reserved	reserved
0x70	(FIFO RAM)	RX data 8	reserved	TX data 8	reserved	reserved
0x74	RX message counter		reserved		RX message counter	reserved
0x78	RX buffer start address		reserved		RX buffer start address	RX buffer start address
0x7C	clock divider		clock divider		clock divider	clock divider

## 19.4.2 发送处理

### 19.4.2.1 发送流程

1. 检查 SR 寄存器的 TBS 位来确保发送缓冲区是空的。
2. 配置帧信息、ID 号和数据。
3. 通过置位 CMR 寄存器中的 TR 位来请求发送。

### 19.4.2.2 终止发送

当 CPU 要求暂停先前的发送时，可以使用终止发送的功能，例如需要先发送更紧急的消息。已经在发送过程中的消息不受此功能影响不会停止。为了查看之前的消息是否发送成功，应该检查 SR 寄存器中的发送完成标志位 (TCS)。应用软件可以通过对 CMR 寄存器中的 AT 位置'1' 来使用该功能，这应该在 SR 寄存器中的 TBS 位为'1' 或者发送中断产生后执行。

有一点需要注意的是，即使消息被终止，也会产生发送中断，因为发送缓冲区的状态位已经指示为“已释放”状态。

### 19.4.2.3 自发自收

应用软件可以通过置位 CMR 寄存器中的 SRR 位实现自发自收，此时发送和接收是同步进行的。其他操作与普通发送流程一样。

### 19.4.2.4 注意点

1. 如果同时置位 CMA 寄存器中的 TR 和 AT 位，消息将会只发送一次。此时即使出现错误事件或者仲裁失败也不会再次发送。
2. 如果同时置位 CMA 寄存器中的 SRR 和 AT 位，消息将使用自发自收的方式只发送一次。此时即使出现错误事件或者仲裁失败也不会再次发送。
3. 如果同时置位 CMA 寄存器中的 SRR、TR 和 AT 位，消息将以同时置位 TR 和 AT 位的方式发送。
4. 一旦状态寄存器中的发送状态位被置位，内部的发送请求位就被自动清零。
5. 如果 CMA 寄存器中的 TR 和 SRR 被同时置位，SRR 位将被忽略。

## 19.4.3 接收处理

### 19.4.3.1 接收流程

接收到的消息被储存在 64 字节深度的内部 FIFO 中，FIFO 完全由硬件管理，从而节省了 CPU 的处理负荷，简化了软件并保证了数据的一致性。应用程序可以通过 FIFO 的输出接口来读取收到的消息。当 SR 寄存器中的 RBS 置位时，RXFIFO 中则有一帧或多帧消息可读，软件获取消息后，将 CMR 寄存器中的 RRB 置位可释放当前消息占用的 RXFIFO。

### 19.4.3.2 消息数量

RMC 寄存器表示 RXFIFO 中的可读消息的数量，该值随每一次的接收事件递增，并随每一次的释放缓冲区递减。复位后该值是 0。

### 19.4.3.3 接收缓冲区

RBSA 寄存器表示当前内部 RAM 中储存的接收到的消息的第一个字节的地址，其映射到接收缓冲区窗口。借助该信息可以解读内部 RAM 中的内容。这部分的内部 RAM 区域可以由 CPU 进行读取和写入（仅在复位模式下可写入）。

示例：如果 RBSA 的值为 18H，则接收缓冲区窗口（偏移地址为 10H 到 12H）的当前可读消息也被存储在 RAM 地址从 18H 开始的位置。由于 RAM 地址是直接映射到 ISO11898 偏移地址 20H（对应 RAM 地址 0H）开始的位置，所以消息也可以从 ISO11898 偏移地址 38H 和后面的字节中读出。（ISO11898 地址 = RBSA + 20H = 18H + 20H = 38H）。如果消息地址超过 RAM 地址 3FH 则它将从 RAM 地址 0 继续。

当 FIFO 中至少还有一条消息时，应该发出释放接收缓冲区的命令，此时 RBSA 就被更新到下一条消息的开始位置。

在硬件复位时，RBSA 寄存器的值被初始化为 ‘00H’，在软件复位（设置为复位模式）时该寄存器值不会变化，但 FIFO 被清除，这意味着 RAM 内容不会改变，但是下一个接收（或发送）的消息将覆盖接收缓冲区窗口中的可见消息。

### 19.4.4 标识符过滤

在接收过滤器的协助下，只有当接收到的消息的标识符位与接收过滤器寄存器中的预定义位相同时，ISO11898 控制器才能允许将接收到的消息传递到 RXFIFO。接收过滤器由接收码寄存器（ACRn）和接收屏蔽寄存器（AMRn）组成。可被接收的消息中的匹配位的值由接收码寄存器设定，哪些位可以屏蔽由接收屏蔽寄存器设定。

有两种不同的滤波模式可以选择（MOD 寄存器中的 AFM 位设定）：

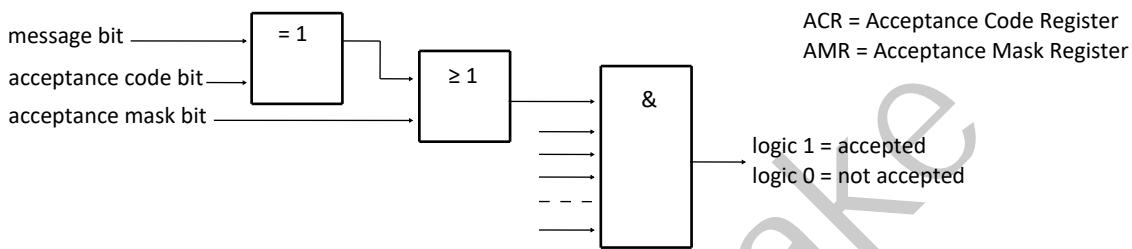
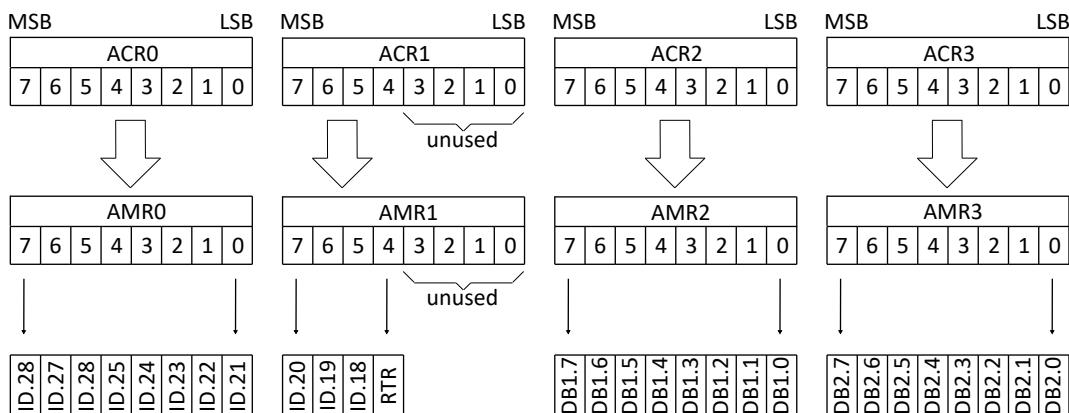
- 单滤波器模式 (AFM = 1)。
- 双滤波器模式 (AFM = 0)。

#### 19.4.4.1 单滤波器配置

在这种配置中，可以定义一个长达 4 字节的滤波器。过滤字节和消息字节之间的位对应关系取决于当前接收的帧格式。

标准帧：如果接收到标准帧格式的消息，包括 RTR 位和前两个数据字节在内的完整标识符用于接受过滤。如果由于设置了 RTR 位而没有数据字节存在，或者由于设定了相应的数据长度而没有数据字节或只有一个数据字节，也可以接收到消息。

所有滤波位之间是逻辑与的关系，必须所有位都通过滤波器，一条消息才能被接收到。请注意 AMR1 和 ACR1 的低 4 位没有使用，为了与将来产品兼容这几位应该被设置为屏蔽位，即 AMR1 的 3~0 位都为 1。

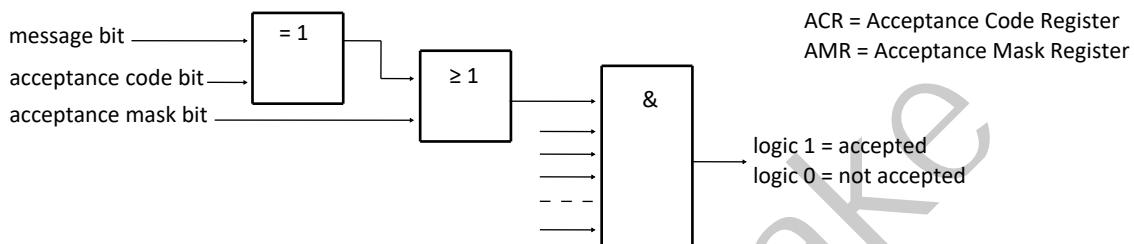
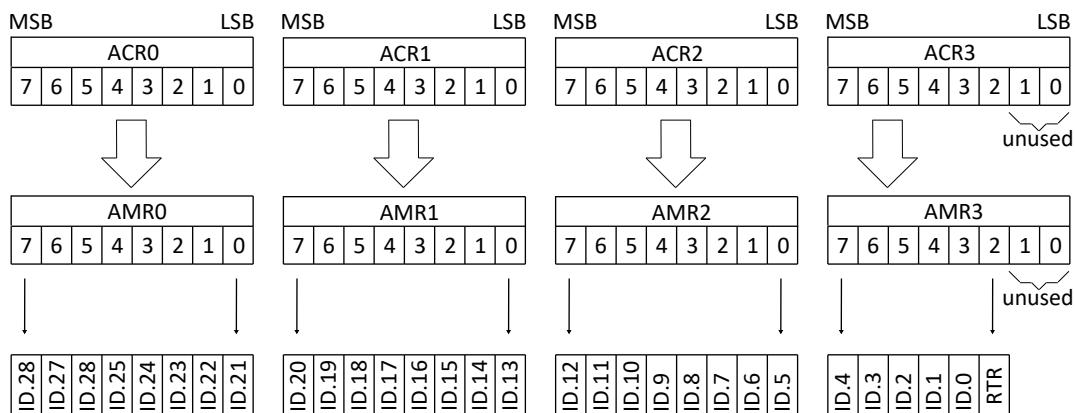


DBX.Y means data byte X, bit Y.

Single filter configuration, receiving standard frame messages

扩展帧: 如果接收到扩展帧格式的消息, 包括 RTR 位在内的完整标识符用于接受过滤。

所有滤波位之间是逻辑与的关系, 必须所有位都通过滤波器, 一条消息才能被接收到。请注意 AMR3 和 ACR3 的低 2 位没有使用, 为了与将来产品兼容这几位应该被设置为屏蔽位, 即 AMR3 的 1~0 位都为 1。



Single filter configuration, receiving extended frame messages

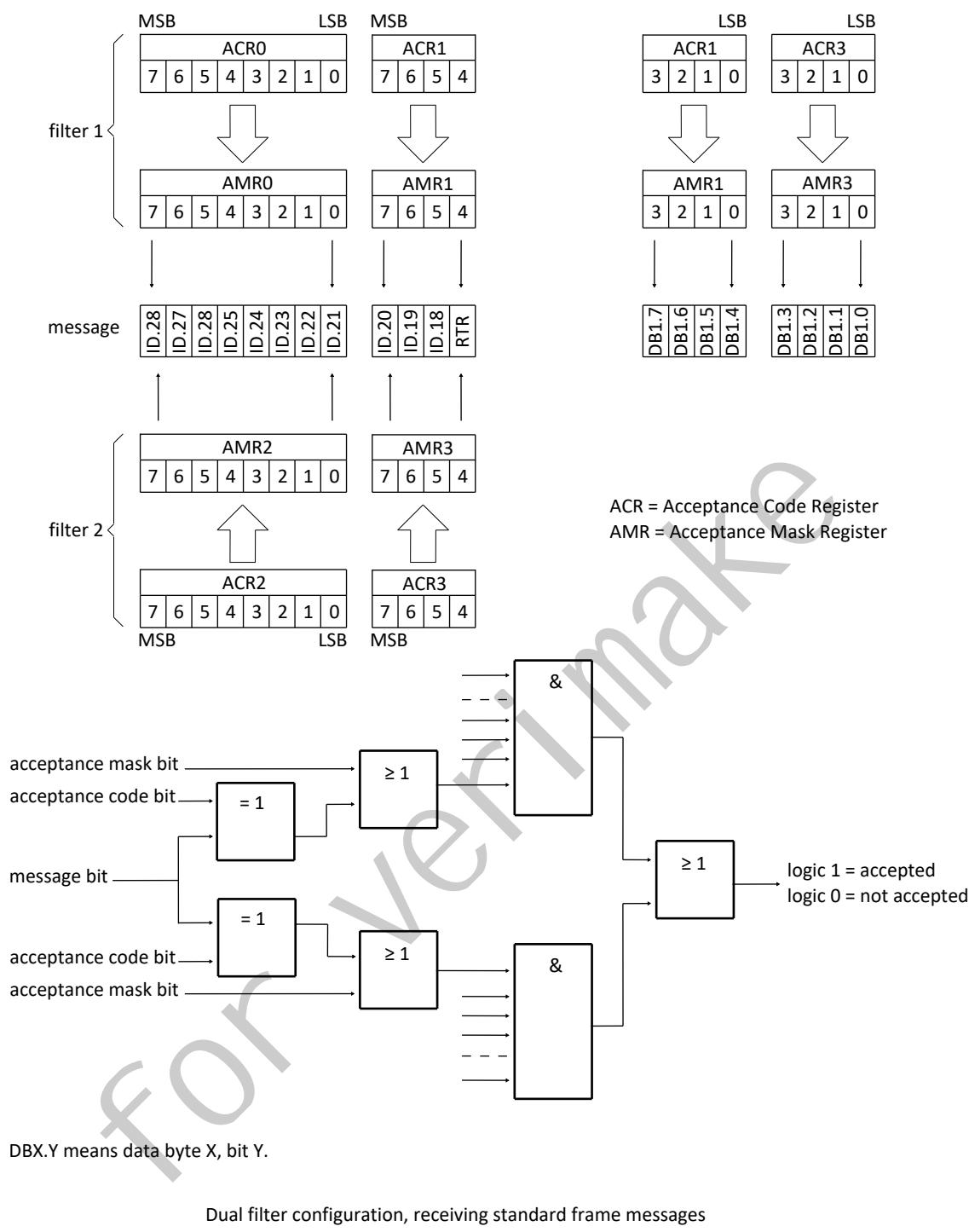
#### 19.4.4.2 双滤波器配置

可以在这种配置中定义两个短滤波器，收到的消息会与这两个滤波器都进行比较，以决定是否将该消息复制到接收缓冲区。只要有一个滤波器接收该消息，则收到的消息就是有效的。过滤字节和消息字节之间的位对应关系取决于当前接收的帧格式。

**标准帧:** 如果接收到标准帧格式的消息，则定义的这两个滤波器看起来有点不一样，第一个滤波器比较包括 RTR 和第一个数据字节在内的完整的标识符，第二个滤波器则只比较包括 RTR 位在内的标准标识符。

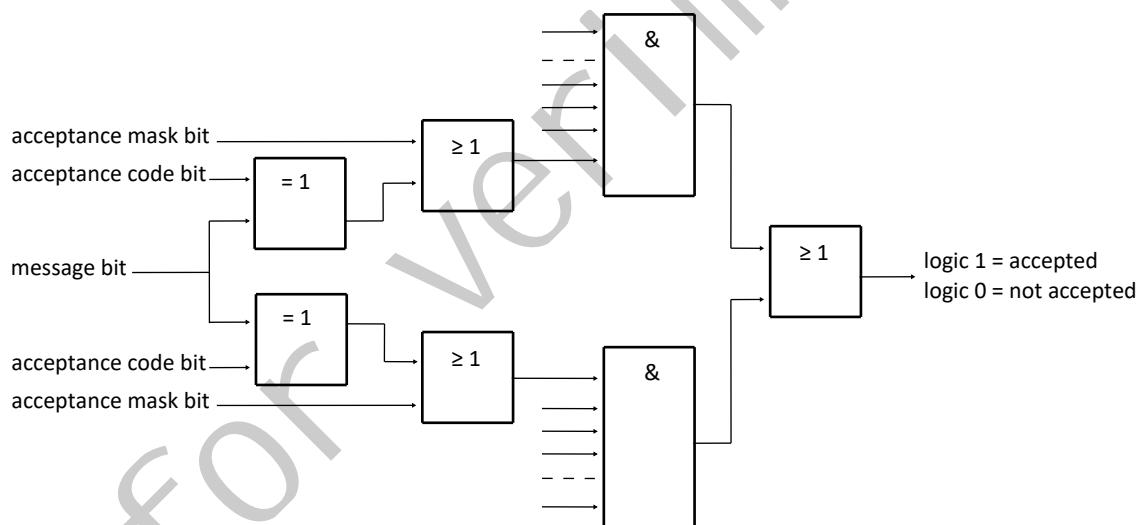
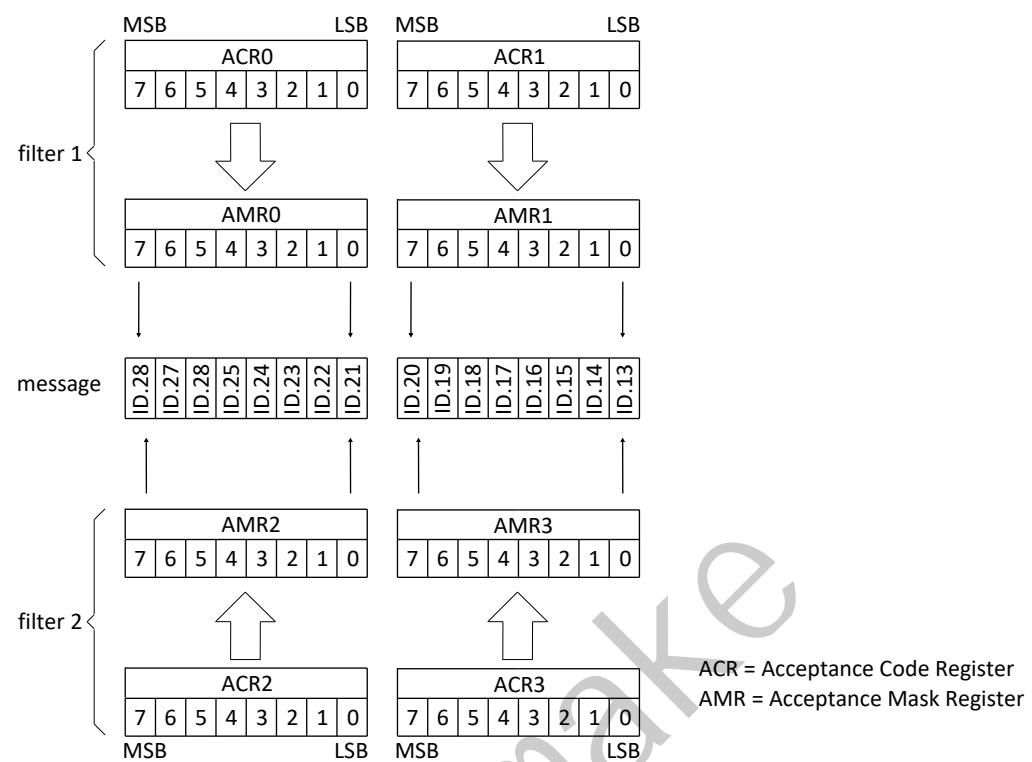
为了成功接收消息，至少一个完整过滤器的所有单个位比较都表示接受。在 RTR 被置位或者数据长度为 0 的情况下是没有数据的。然而，如果直到 RTR 位之前的第一部分都表示接受，则消息也是可以通过滤波器 1 的。

如果第一个滤波器不需要过滤数据字节，则 AMR1 和 AMR3 的低 4 位必须设置为逻辑 1（无关紧要），然后这两个滤波器使用包括 RTR 在内的标准标识符一样地运行。



**扩展帧:** 如果接收到扩展帧格式的消息，则定义的这两个滤波器看起来是一样的。这两个滤波器都只比较扩展标识符的前两个字节。

至少一个完整的滤波器的所有单个位比较都表明接受，消息才能被成功接收。

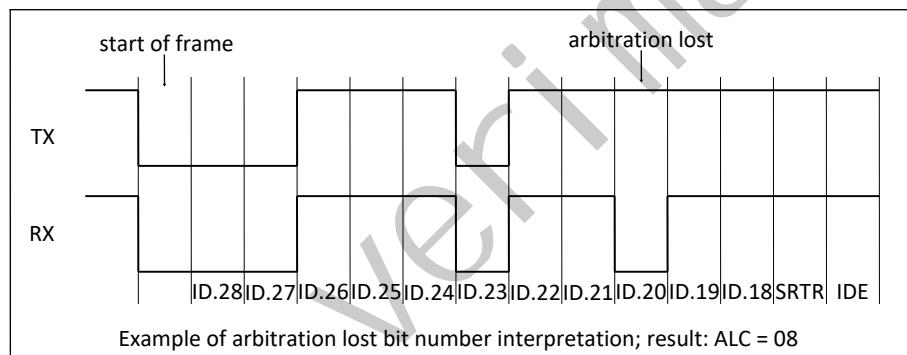
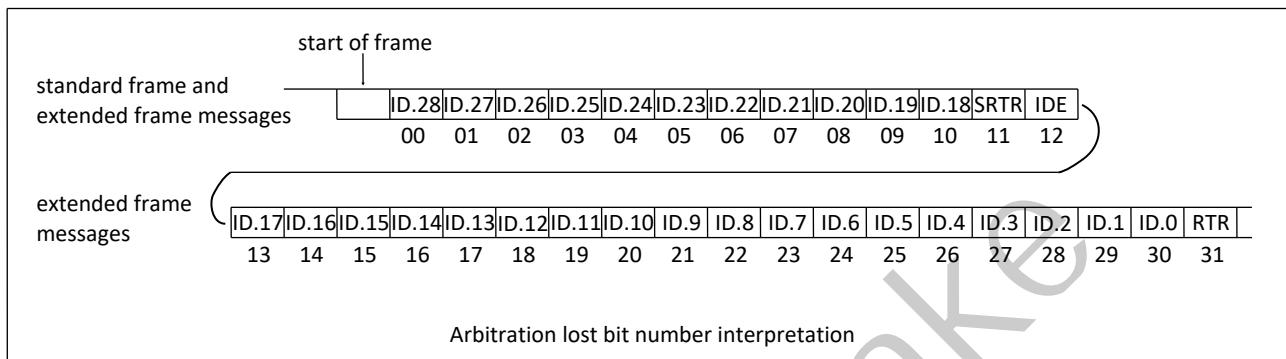


Dual filter configuration, receiving extended frame messages

## 19.4.5 出错管理

### 19.4.5.1 仲裁失败

仲裁失败捕获寄存器（ALC）包含仲裁失败的位置，该寄存器只能被 CPU 读取不能写入。如果使能了仲裁失败的中断，则一旦仲裁失败将产生中断。同时，位流处理器中的当前位的位置将被捕获到 ALC 中。在用户软件读取 ALC 的内容之前，该寄存器的值将一直固定不变。读取该寄存器值后，捕获机制就会再次激活。在中断寄存器被读出的时候，相应的中断标志也是被清除的，在中断失败寄存器被读出之前是不会再次产生仲裁失败中断的。



BITS					DECIMAL VALUE	FUNCTION
ALC.4	ALC.3	ALC.2	ALC.1	ALC.0		
0	0	0	0	0	00	arbitration lost in bit 1 of identifier
0	0	0	0	1	01	arbitration lost in bit 2 of identifier
0	0	0	1	0	02	arbitration lost in bit 3 of identifier
0	0	0	1	1	03	arbitration lost in bit 4 of identifier
0	0	1	0	0	04	arbitration lost in bit 5 of identifier
0	0	1	0	1	05	arbitration lost in bit 6 of identifier

0	0	1	1	0	06	arbitration lost in bit 7 of identifier
0	0	1	1	1	07	arbitration lost in bit 8 of identifier
0	1	0	0	0	08	arbitration lost in bit 9 of identifier
0	1	0	0	1	09	arbitration lost in bit 10 of identifier
0	1	0	1	0	10	arbitration lost in bit 11 of identifier
0	1	0	1	1	11	arbitration lost in bit SRTR
0	1	1	0	0	12	arbitration lost in bit IDE
0	1	1	0	1	13	arbitration lost in bit 12 of identifier
0	1	1	1	0	14	arbitration lost in bit 13 of identifier
0	1	1	1	1	15	arbitration lost in bit 14 of identifier
1	0	0	0	0	16	arbitration lost in bit 15 of identifier
1	0	0	0	1	17	arbitration lost in bit 16 of identifier
1	0	0	1	0	18	arbitration lost in bit 17 of identifier
1	0	0	1	1	19	arbitration lost in bit 18 of identifier
1	0	1	0	0	20	arbitration lost in bit 19 of identifier
1	0	1	0	1	21	arbitration lost in bit 20 of identifier
1	0	1	1	0	22	arbitration lost in bit 21 of identifier
1	0	1	1	1	23	arbitration lost in bit 22 of identifier
1	1	0	0	0	24	arbitration lost in bit 23 of identifier
1	1	0	0	1	25	arbitration lost in bit 24 of identifier
1	1	0	1	0	26	arbitration lost in bit 25 of identifier

1	1	0	1	1	27	arbitration lost in bit 26 of identifier
0	1	1	0	0	28	arbitration lost in bit 27 of identifier
1	1	1	0	1	29	arbitration lost in bit 28 of identifier
1	1	1	1	0	30	arbitration lost in bit 29 of identifier
1	1	1	1	1	31	arbitration lost in bit RTR

#### 19.4.5.2 错误捕获

错误捕获寄存器（ECC）包含总线错误的类型和位置，该寄存器只能被 CPU 读取不能写入。如果是能了总线错误中断，则一旦总线发生错误将产生总线错误中断。同时，位流处理器中的当前位的位置将被捕获到 ECC 中。在用户软件读取 ECC 的内容之前，该寄存器的值将一直固定不变。读取该寄存器值后，捕获机制就会再次激活。对中断寄存器中相应位的读取操作会清除该位，在读取捕获寄存器之前，不会再次产生总线错误中断。

错误捕获寄存器中值代表的错误类型和种类如下所示。

BIT ECC.7	BIT ECC.6	FUNCTION
0	0	bit error
0	1	form error
1	0	stuff error
1	1	other type of error

BIT ECC.4	BIT ECC.3	BIT ECC.2	BIT ECC.1	BIT ECC.0	FUNCTION
0	0	0	1	1	start of frame
0	0	0	1	0	ID.28 to ID.21
0	0	1	1	0	ID.20 to ID.18
0	0	1	0	0	bit SRTR
0	0	1	0	1	bit IDE
0	0	1	1	1	ID.17 to ID.13
0	1	1	1	1	ID.12 to ID.5
0	1	1	1	0	ID.4 to ID.0
0	1	1	0	0	bit RTR

0	1	1	0	1	reserved bit 1
0	1	0	0	1	reserved bit 0
0	1	0	1	1	data length code
0	1	0	1	0	data field
0	1	0	0	0	CRC sequence
1	1	0	0	0	CRC delimiter
1	1	0	0	1	acknowledge slot
1	1	0	1	1	acknowledge delimiter
1	1	0	1	0	end of frame
1	0	0	1	0	intermission
1	0	0	0	1	active error flag
1	0	1	1	0	passive error flag
1	0	0	1	1	tolerate dominant bits
1	0	1	1	1	error delimiter
1	1	1	0	0	overload flag

#### 19.4.5.3 接收错误计数器 (RXERR)

接收错误计数寄存器的值代表当前接收错误的数量，在硬件复位后该寄存器被初始化为逻辑 0。在操作模式下该寄存器只能被 CPU 执行读取操作，对该寄存器的写操作只能在复位模式下执行。如果发生总线关闭事件，RXERR 被设置为逻辑 0。此时，总线处于关闭状态，对该寄存器的写操作不起作用。

需要注意的是，只有先进入复位模式，CPU 才可以对 RXERR 的值进行修改，在这样的情况下，错误状态可能会改变，错误警告中断和错误被动中断不会发生，除非再取消复位模式。

#### 19.4.5.4 发送错误计数器 (TXERR)

发送错误计数寄存器的值代表当前发送错误的数量，在操作模式下该寄存器只能被 CPU 执行读取操作，对该寄存器的写操作只能在复位模式下执行。在硬件复位后该寄存器的值被初始化为逻辑 0。如果发生总线关闭事件，TXERR 值就被设定为 127，这样就可以计算协议定义的最短时间（出现 128 次总线空闲信号）。在此期间读取该寄存器的值可以获取总线关闭恢复的状态信息。如果总线处于关闭状态，则对 TXERR 的范围从 0 到 254 的写操作会清除总线关闭状态标志，并且控制器将在清除复位模式后等待 11 个连续隐形位（总线空闲）出现一次。

通过 CPU 将 255 写入 TXERR 将产生总线关闭事件，需要注意的是只有先进入复位模式才可以进行 CPU 强制修改该寄存器值的操作，这样的情况下，错误状态或者总线状态将可能改变，错误警告中断或错误被动中断不会受新值影响，除非再退出复位模式。退出复位模式后，TXERR 的值还好像和发生总线错误导致总线关闭那样一样的机制运行，这意味着会再次进入复位模式，TXERR 的值又被初始化为 127，RXERR 的值被初始化为 0，并且相关状态和中断寄存

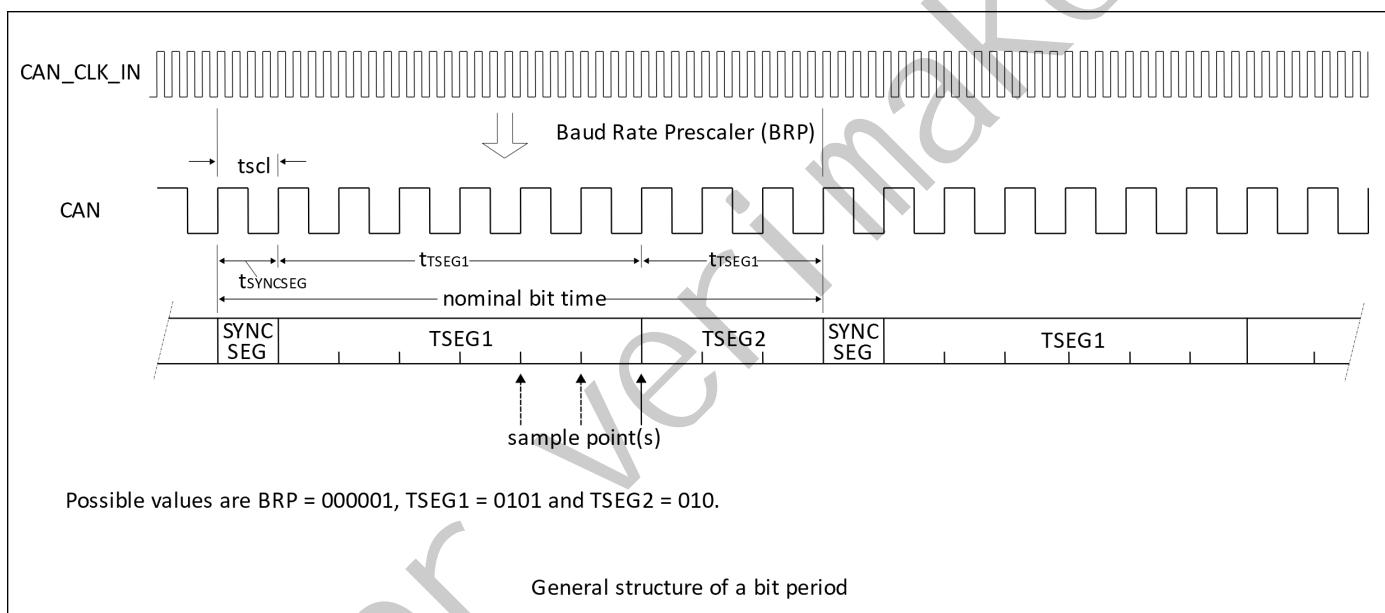
器都被重新设置。此时退出复位模式将执行协议定义的总线关闭恢复流程（等待 128 个总线空闲信号的发生）。如果在总线关闭并恢复前（TXERR>0）再次进入复位模式，总线将继续保持关闭状态并且 TXERR 的值被冻结。

#### 19.4.5.5 错误限值设定

错误警告限制可以由 EWLR 寄存器设定，该寄存器默认值（硬件复位后）是 96。在复位模式下，这个寄存器可被 CPU 读取或写入，在操作模式下，该寄存器只能被读取。当 RXERR 和 TXERR 两个错误计数值至少一个大于等于 EWLR 寄存器设定的值时，SR 寄存器中的 ES 位将被置位，否者被清零，此时如果 IER 寄存器中的 EIE 位被置位，则将产生错误警告中断。需要注意的是该寄存器只有在先进入复位模式后才能操作。对该寄存器的操作可能会引起错误状态的改变，并且不会让错误警告中断产生，除非再退出复位模式。

#### 19.4.6 位时序

时序图如下：



#### 19.4.6.1 波特率分频器 (BRP)

ISO11898 系统时钟 tscl 的周期是可以设定的，并且这确定了各个位的时序。ISO11898 系统时钟的计算公式如下：

$$tscl = 2 * tCLK * (32 * BRP.5 + 16 * BRP.4 + 8 * BRP.3 + 4 * BRP.2 + 2 * BRP.1 + BRP.0 + 1)$$

#### 19.4.6.2 同步跳转宽度 (SJW)

为了补偿不同总线控制器的时钟振荡器之间的相移，任何总线控制器都必须在当前传输的任何相关信号边缘重新同步。同步跳转宽度定义了一个位周期可以通过一次重新同步缩短或延长的最大时钟周期数：

$$t_{SJW} = tscl * (2 * SJW.1 + SJW.0 + 1)$$

#### 19.4.6.3 采样 (SAM)

当 BTR1 寄存器中的 SAM 位为 1 时，总线将采样三次，这种模式推荐在中低速总线中使用，此时总线中的滤波器将有好处。如果 SAM 位为 0，则总线只采样一次，这种模式推荐在高速模式中使用。

#### 19.4.6.4 时间段 (TSEG)

TSEG 包含 BTR1 寄存器中 TSEG1 和 TSEG2 两部分，它决定了每一个位的时钟数和采样点位置，计算公式如下：

$$t_{SYNCSEG} = 1 * tscl$$

$$t_{TSEG1} = tscl * (8 * TSEG1.3 + 4 * TSEG1.2 + 2 * TSEG1.1 + TSEG1.0 + 1)$$

$$t_{TSEG2} = tscl * (4 * TSEG2.2 + 2 * TSEG2.1 + TSEG2.0 + 1)$$

## 20.1 简介

CAM(Camera) 模块负责将并行接口 (DVP) 转换成一般总线接口 (AHB)，把图像传感器生成的像素数据写入系统内存中，作为后续影像传输或压缩使用。CAM 模块拥有灵活的输出格式配置，可以满足多种多样的图像处理需求。

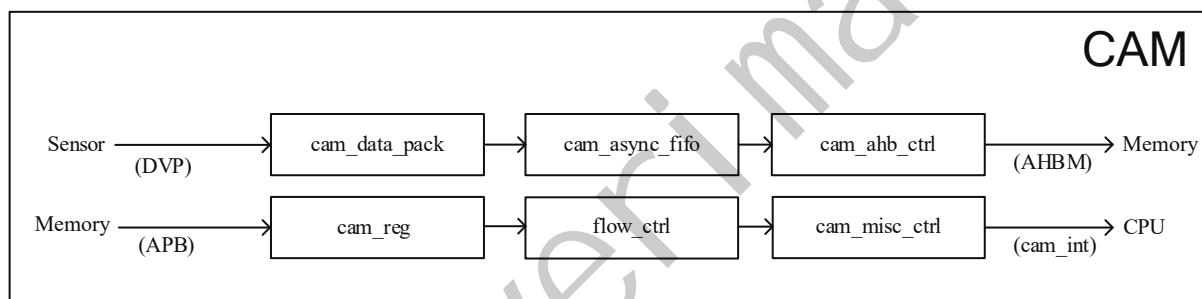


图 20.1: Cam 框图

## 20.2 主要特征

- 并行接口 8-bitDVP 信号，高速数据传输 (80M)，可配置 DVP 信号有效水平与逻辑组合
- 支持 8-bit/16-bit/24-bit 输入像素位宽
- 支持将 RGB888 输入格式转为 RGB565 或 RGBA8888 输出
- 支持影片模式和照片模式
- 可配置丢弃模式，包括：
  - 丢弃奇数位数据
  - 丢弃偶数位数据
  - 丢弃奇数行的奇数位数据
  - 丢弃奇数行的偶数位数据

- 可配置的图像传感器行帧同步信号选择和极性选择
- 支持图像矩形裁剪
- 以 1~32 为周期的帧取舍功能
- 支持行帧同步信号的完整性检测
- AHB 总线通信接口
- 512 字节缓存 FIFO 以应对总线偶而忙碌的状态
- 连续缓存多达 4 组图像信息
- 多种应用中断，有利于弹性使用与出错提示

## 20.3 功能描述

### 20.3.1 DVP(Digital Video Port) 信号与配置

DVP (Digital Video Port) 是并行接口，主要有时钟，帧同步，行同步与 8-bit 数据管脚，时钟的极限限制在 80MHz，所以一般用于 5MP 以下分辨率 Sensor。芯片内可独立配置帧同步与行同步的有效电平，并在有效数据上提供四种模式：

- A. 帧同步与行同步同时有效 (“&” 逻辑)
- B. 帧同步或行同步择一有效 (“|” 逻辑)
- C. 帧同步有效
- D. 行同步有效

### 20.3.2 YCbCr 格式

亮度信号被称为 Y，色度信号是由两个互相独立的信号组成。视颜色系统和格式不同，两种色度信号经常被称为 U、V 或 Pb、Pr 或 Cb、Cr。这是由不同的编码格式产生的，但实际上它们的概念基本相同。

由于人类视网膜上识别亮度的视网膜杆细胞要多于识别色度的视网膜锥细胞，人眼对亮度的敏感程度要高于对色度的敏感程度。所以可以将色度信息丢弃一部分而不被人眼所察觉。

色度信号分辨率最高的格式是 4:4:4，即每 4 点 Y 采样对应了 4 点 Cb 和 4 点 Cr 采样。而 4:2:2 是每 4 点 Y 采样对应了 2 点 Cb 和 2 点 Cr 采样，在这种格式中，色度信号的扫描线数量和亮度信号一样多，但是每条扫描线上的色度采样点却只有亮度信号的一半。与上面提到的格式不同，4:2:0 并不是每 4 点 Y 采样对应 2 点 Cb 和 0 点 Cr 采样，而是每 4 点 Y 采样对应 1 点 Cb 和 1 点 Cr 采样。4:0:0 是丢弃所有的色度信息，即灰度图。

### 20.3.3 影片模式/照片模式

照片模式下当软件给的固定大小的存储器被写满时，CAM 会停止，需要软件进行 POP 操作将空间空出后才会继续写入。

影片模式下会在软件给的固定大小的存储器上不停地复写，也就是将内存当作 ring buffer 的概念，无需软件进行 POP 操作，使用上要确保图像被实时取出或是跟着 MJPEG 模块做连动。

RGB888 转 RGB565/RGBA8888 输出对于输入格式为 RGB888 的图像数据，可以选择转为 RGB565 或 RGBA8888 写入到内存中。如果转为 RGB565 格式，则 R、G、B 的排列顺序可以通过寄存器 MISC 的位 FORMAT\_565 进行控制，不同的值对应的排列顺序如下所示：

- 0: byte2[7:3], byte1[7:2], byte0[7:3]
- 1: byte1[7:3], byte2[7:2], byte0[7:3]
- 2: byte2[7:3], byte0[7:2], byte1[7:3]
- 3: byte0[7:3], byte2[7:2], byte1[7:3]
- 4: byte1[7:3], byte0[7:2], byte2[7:3]
- 5: byte0[7:3], byte1[7:2], byte2[7:3]

如果转为 RGBA8888 格式，则 A 的值与寄存器 MISC 的位 ALPHA 中填入的值一致。

### 20.3.4 图像矩形裁剪

通过寄存器 HSYNC\_CONTROL 和 VSYNC\_CONTROL 的高 16 位和低 16 位分别设置行同步信号和帧同步信号裁剪的起始和结束位置，就可以将指定位置和大小的矩形窗口内的图像裁剪下来，超出矩形部分的数据会被丢弃。其中行同步信号起始和结束设置的是像素序号，帧同步信号起始和结束设置的是行号，裁剪后的图像包含起始点而不包含结束点。

### 20.3.5 帧取舍功能

可以通过寄存器 FRAME\_PERIOD 设置一个帧周期 n，n 的取值范围是 0~31，对应的实际值是 n+1，然后通过寄存器 FRAME\_VLD 设置在一个帧周期中保留哪几帧图像。需要保留的在对应的 bit 位置写 1，需要舍弃的在对应的 bit 位置写 0。比如 n 设为 5，FRAME\_VLD 的值设为 0x13，则每 6 帧图像中，第 1、2、5 帧图像会写入内存中，而第 3、4、6 帧图像会被舍弃，以 6 帧为周期进行循环。

### 20.3.6 行帧同步信号完整性检测

通过寄存器 FRAME\_SIZE\_CONTROL 的低 16 位和高 16 位可以分别设置行同步信号比较值和帧同步信号比较值，可以对信号的完整性进行检测。其中行同步信号设置的是每行的总像素数，帧同步信号设置的是总行数。当一帧图像的行或帧同步信号计数值与比较值不相等时，会有对应的中断产生。

### 20.3.7 缓存图像信息

模块内部包含 4 组 FIFO 记录图像地址和图像 ID。每当此模块完整写入一帧到内存，便会将此帧图像的起始地址和图像 ID 纪录于此 FIFO 中，但要注意的是当发生内存剩余不足，或是 4 组 FIFO 满存的状况时，模块会自动丢掉接下来图像的讯息，在图像信息取出的部分，可通过 pop 操作将最旧的图像信息空出，此时 FIFO 会自动推进，保证 FIFO 内部图像信息的时序，如下图：

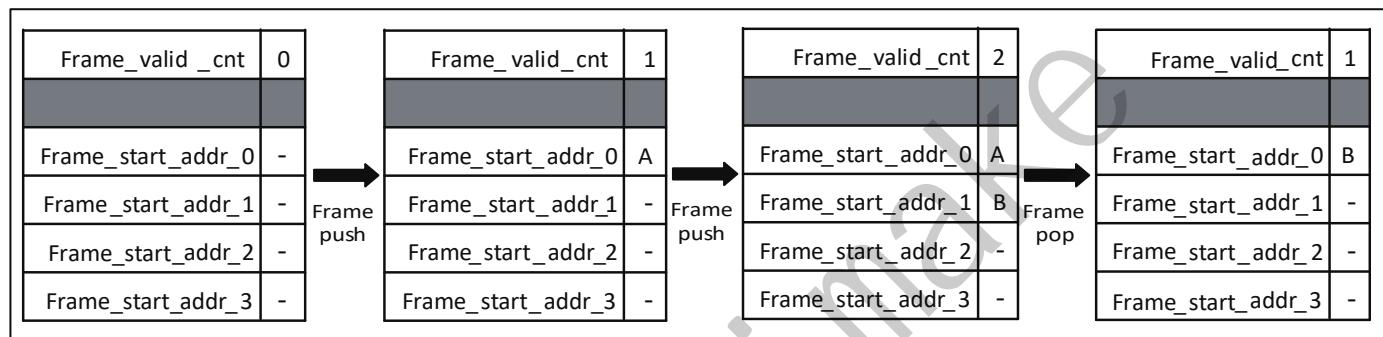


图 20.2: FIFO 框架

### 20.3.8 支持多种中断信息 (可独立开关配置)

A、Normal 中断-可设定一个计数值 n，每当写入 n 张图像后就会发出一次中断

B、Memory 中断-当内存剩余空间不足一帧大小，已经使用了的内存被复写时，发出中断，如下图所示：

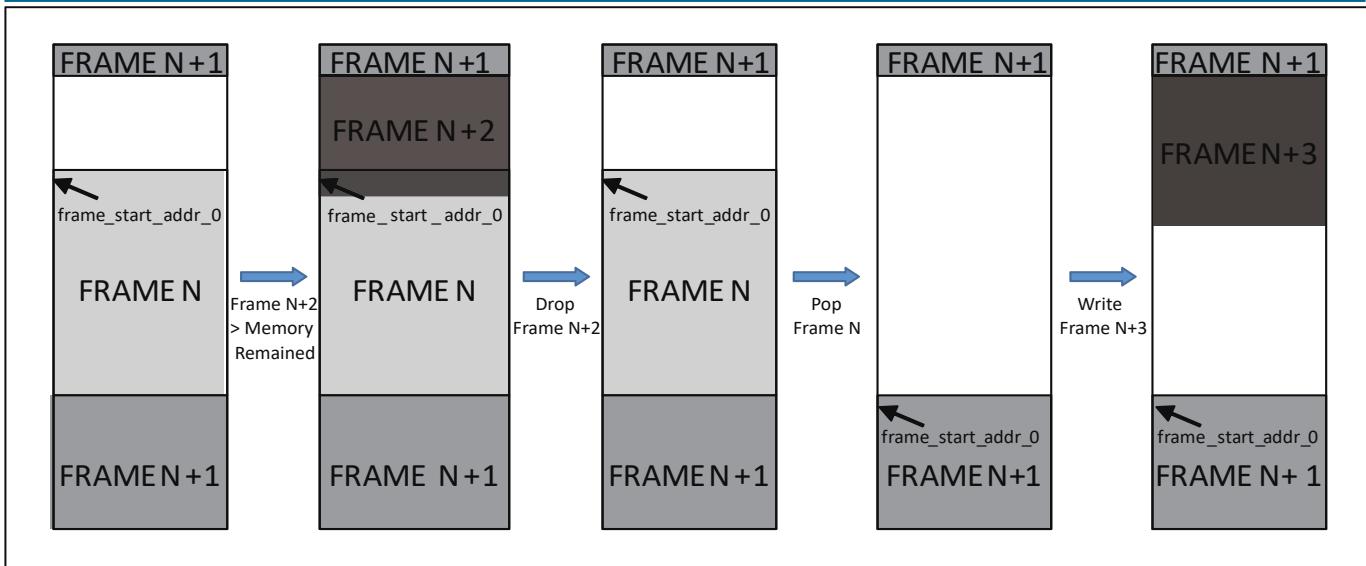


图 20.3: 内存

- C、Frame 中断-当未处理的图像超过 4 组，无法再存储更多的图像信息时，发出中断
- D、FIFO 中断-当总线来不及写入内存，导致 FIFO 溢出时，发出中断
- E、Hsync 中断 - 当一帧图像中某行的像素点数与设置值不相等（行同步信号完整性检测不通过）时，发出中断
- F、Vsync 中断 - 当一帧图像的总行数与设置值不相等（帧同步信号完整性检测不通过）时，发出中断

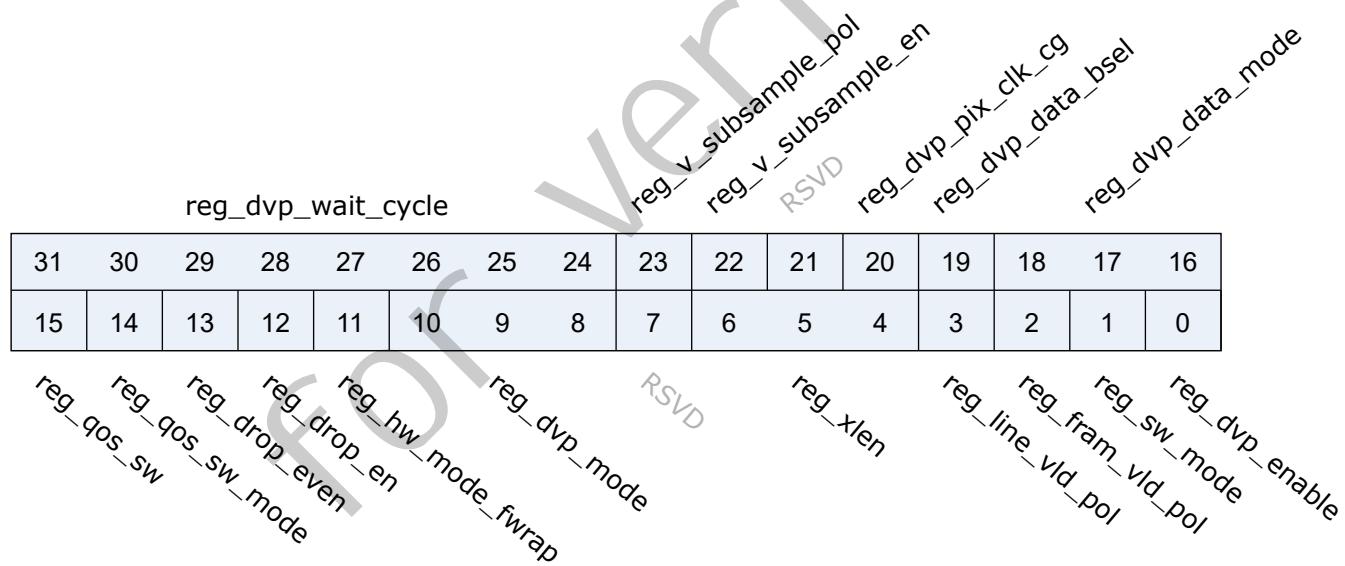
## 20.4 寄存器描述

名称	描述
dvp2axi_configue	
dvp2axi_addr_start	
dvp2axi_mem_bcnt	
dvp_status_and_error	
dvp2axi_frame_bcnt	
dvp_frame_fifo_pop	
dvp2axi_frame_vld	
dvp2axi_frame_period	
dvp2axi_misc	
dvp2axi_hsync_crop	
dvp2axi_vsync_crop	

名称	描述
dvp2axi_fram_exm	
frame_start_addr0	
frame_start_addr1	
frame_start_addr2	
frame_start_addr3	
frame_id_sts01	
frame_id_sts23	
dvp_debug	
dvp_dummy_reg	

#### 20.4.1 dvp2axi\_configue

地址: 0x30015000



位	名称	权限	复位值	描述
31:24	reg_dvp_wait_cycle	r/w	8'h40	Cycles in FSM Wait mode
23	reg_v_subsample_pol	r/w	1'b0	DVP2BUS vertical sub-sampling polarity 1'b0: Odd lines are masked 1'b1: Even lines are masked
22	reg_v_subsample_en	r/w	1'b0	DVP2BUS vertical sub-sampling enable

位	名称	权限	复位值	描述
21	RSVD			
20	reg_dvp_pix_clk_cg	r/w	1'b0	DVP pix clk gate
19	reg_dvp_data_bsel	r/w	1'b0	Byte select signal for DVP 8-bit mode, don't care if reg_dvp_data_8bit is disabled 1'b0: Select the lower byte of pix_data 1'b1: Select the upper byte of pix_data
18:16	reg_dvp_data_mode	r/w	3'b0	DVP 8-bit mode enable 3'd0: DVP pix_data is 16-bit wide 3'd1: DVP pix_data is 24-bit mode 3'd2: DVP pix_data is 24-comp-16-bit mode 3'd3: DVP pix_data is 24-exp-32-bit mode 3'd4: DVP pix_data is 8-bit wide Others - Reserved
15	reg_qos_sw	r/w	1'b0	AXI Qos software mode value
14	reg_qos_sw_mode	r/w	1'b0	AXI QoS software mode enable
13	reg_drop_even	r/w	1'b0	Only effect when reg_drop_en=1 : 1'b1 : Drop all even bytes 1'b0 : Drop all odd bytes
12	reg_drop_en	r/w	1'b0	Drop mode Enable
11	reg_hw_mode_fwrap	r/w	1'b1	DVP2BUS HW mode with frame start address wrap to reg_addr_start
10:8	reg_dvp_mode	r/w	3'd0	Image sensor mode selection: 3'd0-Vsync&Hsync 3'd1-Vsync Hsync 3'd2-Vsync 3'd3-Hsync
7	RSVD			
6:4	reg_xlen	r/w	3'd3	burst length setting 3'd0 - Single / 3'd1 - INCR4 / 3'd2 - INCR8 3'd3 - INCR16 / 3'd5 - INCR32 / 3'd6 - INCR64
3	reg_line_vld_pol	r/w	1'b1	Image sensor line valid polarity, 1'b0 - Active low, 1'b1 - Active high
2	reg_fram_vld_pol	r/w	1'b1	Image sensor frame valid polarity, 1'b0 - Active low, 1'b1 - Active high
1	reg_sw_mode	r/w	1'b0	DVP2BUS SW manual mode (don't care if reg_swap_mode is enabled)
0	reg_dvp_enable	r/w	1'b0	module enable

#### 20.4.2 dvp2axi\_addr\_start

地址: 0x30015004

reg\_addr\_start

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\_addr\_start

位	名称	权限	复位值	描述
31:0	reg_addr_start	r/w	32'h80000000	AXI start address

#### 20.4.3 dvp2axi\_mem\_bcnt

地址: 0x30015008

reg\_mem\_burst\_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

reg\_mem\_burst\_cnt

位	名称	权限	复位值	描述
31:0	reg_mem_burst_cnt	r/w	32'hC000	AXI burst cnt before wrap to "reg_addr_start"

#### 20.4.4 dvp\_status\_and\_error

地址: 0x3001500c

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

frame\_valid\_cnt

RSVD RSVD st\_dvp\_idle axi\_idle st\_bus\_fsh st\_bus\_wait st\_bus\_func st\_bus\_idle RSVD sts\_vcnt\_int sts\_hcnt\_int

reg\_frame\_cnt\_trgr\_int

RSVD

sts\_fifo\_int sts\_frame\_int sts\_mem\_int sts\_normal\_int reg\_int\_fifo\_en reg\_int\_frame\_en reg\_int\_mem\_en reg\_int\_normal\_en reg\_int\_vcint\_en reg\_int\_hcnt\_en

位	名称	权限	复位值	描述
31:30	RSVD			
29	st_dvp_idle	r	1'b1	DVP2BUS asynchronous fifo idle status
28	axi_idle	r	1'b1	DVP2BUS AHB idle status
27	st_bus_flsh	r	1'b0	DVP in flush state
26	st_bus_wait	r	1'b0	DVP in wait state
25	st_bus_func	r	1'b0	DVP in functional state
24	st_bus_idle	r	1'b1	DVP in idle state
23	RSVD			
22	sts_vcnt_int	r	1'b0	Vsync valid line count non-match interrupt status
21	sts_hcnt_int	r	1'b0	Hsync valid pixel count non-match interrupt status
20:16	frame_valid_cnt	r	5'd0	Frame counts in memory before read out in SW mode
15	sts_fifo_int	r	1'b0	FIFO OverWrite interrupt status
14	sts_frame_int	r	1'b0	Frame OverWrite interrupt status
13	sts_mem_int	r	1'b0	Memory OverWrite interrupt status
12	sts_normal_int	r	1'b0	Normal Write interrupt status
11	reg_int_fifo_en	r/w	1'b1	FIFO OverWrite interrupt enable
10	reg_int_frame_en	r/w	1'b0	Frame OverWrite interrupt enable
9	reg_int_mem_en	r/w	1'b0	Memory OverWrite interrupt enable
8	reg_int_normal_en	r/w	1'b0	Normal Write interrupt enable
7	reg_int_vcnt_en	r/w	1'b0	Vsync valid line count match interrupt enable
6	reg_int_hcnt_en	r/w	1'b0	Hsync valid pixel count match interrupt enable
5	RSVD			
4:0	reg_frame_cnt_trgr_int	r/w	5'd0	Frame to issue interrupt at SW Mode

#### 20.4.5 dvp2axi\_frame\_bcnt

地址: 0x30015010

reg\_frame\_byte\_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

reg\_frame\_byte\_cnt

位	名称	权限	复位值	描述
31:0	reg_frame_byte_cnt	r/w	32'h7e90	Single Frame byte cnt(Need pre-calculation)

#### 20.4.6 dvp\_frame\_fifo\_pop

地址: 0x30015014

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  
 reg\_int\_vcnt\_clr reg\_int\_hcnt\_clr reg\_int\_fifo\_clr reg\_int\_frame\_clr reg\_int\_normal\_clr  
 rfifo\_pop

位	名称	权限	复位值	描述
31:10	RSVD			
9	reg_int_vcnt_clr	w1p	1'd0	Interrupt clear
8	reg_int_hcnt_clr	w1p	1'd0	Interrupt clear
7	reg_int_fifo_clr	w1p	1'd0	Interrupt clear
6	reg_int_frame_clr	w1p	1'd0	Interrupt clear
5	reg_int_mem_clr	w1p	1'd0	Interrupt clear
4	reg_int_normal_clr	w1p	1'd0	Interrupt clear
3:1	RSVD			
0	rfifo_pop	w1p	1'b0	Write this bit will trigger fifo pop

#### 20.4.7 dvp2axi\_frame\_vld

地址: 0x30015018

reg_frame_n_vld															
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\_frame\_n\_vld

位	名称	权限	复位值	描述
31:0	reg_frame_n_vld	r/w	32'hffff_-_ffff	Bitwise frame valid in period

#### 20.4.8 dvp2axi\_frame\_period

地址: 0x3001501c

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\_frame\_period

位	名称	权限	复位值	描述
31:5	RSVD			
4:0	reg_frame_period	r/w	5'h0	Frame period cnt. (EX. Set this register 0, the period is 1)

#### 20.4.9 dvp2axi\_misc

地址: 0x30015020

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\_alpha

reg\_format\_565

位	名称	权限	复位值	描述
31:11	RSVD			

位	名称	权限	复位值	描述
10:8	reg_format_565	r/w	3'd0	Only work when reg_dvp_data_mode=2 (24-comp-16-bit mode) 3'd0: B2(5)B1(6)B0(5) 3'd1: B1(5)B2(6)B0(5) 3'd2: B2(5)B0(6)B1(5) 3'd3: B0(5)B2(6)B1(5) 3'd4: B1(5)B0(6)B2(5) 3'd5: B0(5)B1(6)B2(5)
7:0	reg_alpha	r/w	8'h0	Only work when "reg_dvp_data_mode==2'd3(DVP pix - data is 24-exp-32-bit mode)" The value of [31:24]

#### 20.4.10 dvp2axi\_hsync\_crop

地址: 0x30015030

reg\_hsync\_act\_start

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\_hsync\_act\_end

位	名称	权限	复位值	描述
31:16	reg_hsync_act_start	r/w	16'h0	Valid hsync start cnt
15:0	reg_hsync_act_end	r/w	16'hFFFF	Valid hsync end cnt

#### 20.4.11 dvp2axi\_vsync\_crop

地址: 0x30015034

reg\_vsync\_act\_start

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\_vsync\_act\_end

位	名称	权限	复位值	描述
31:16	reg_vsync_act_start	r/w	16'h0	Valid vsync start cnt
15:0	reg_vsync_act_end	r/w	16'hFFFF	Valid vsync end cnt

#### 20.4.12 dvp2axi\_fram\_exm

地址: 0x30015038

reg\_total\_vcnt

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\_total\_hcnt

位	名称	权限	复位值	描述
31:16	reg_total_vcnt	r/w	16'h0	Total valid line count in a frame
15:0	reg_total_hcnt	r/w	16'h0	Total valid pix count in a line

#### 20.4.13 frame\_start\_addr0

地址: 0x30015040

frame\_start\_addr\_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

frame\_start\_addr\_0

位	名称	权限	复位值	描述
31:0	frame_start_addr_0	r	32'd0	DVP2BUS PIC 0 Start address

#### 20.4.14 frame\_start\_addr1

地址: 0x30015048

frame\_start\_addr\_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

frame\_start\_addr\_1

位	名称	权限	复位值	描述
31:0	frame_start_addr_1	r	32'd0	DVP2BUS PIC 1 Start address

#### 20.4.15 frame\_start\_addr2

地址: 0x30015050

frame\_start\_addr\_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

frame\_start\_addr\_2

位	名称	权限	复位值	描述
31:0	frame_start_addr_2	r	32'd0	DVP2BUS PIC 2 Start address

#### 20.4.16 frame\_start\_addr3

地址: 0x30015058

frame\_start\_addr\_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

frame\_start\_addr\_3

位	名称	权限	复位值	描述
31:0	frame_start_addr_3	r	32'd0	DVP2BUS PIC 3 Start address

#### 20.4.17 frame\_id\_sts01

地址: 0x30015060

frame\_id\_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

frame\_id\_0

位	名称	权限	复位值	描述
31:16	frame_id_1	r	16'd0	DVP2BUS PIC 1 ID
15:0	frame_id_0	r	16'd0	DVP2BUS PIC 0 ID

#### 20.4.18 frame\_id\_sts23

地址: 0x30015064

frame\_id\_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

frame\_id\_2

位	名称	权限	复位值	描述
31:16	frame_id_3	r	16'd0	DVP2BUS PIC 3 ID
15:0	frame_id_2	r	16'd0	DVP2BUS PIC 2 ID

#### 20.4.19 dvp\_debug

地址: 0x300150f0

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															

位	名称	权限	复位值	描述
31:12	RSVD			
11:8	reg_id_latch_line	r/w	4'd5	ID latch timing (line count)
7:4	RSVD			
3:1	reg_dvp_dbg_sel	r/w	3'd0	DVP2BUS debug flag selection

位	名称	权限	复位值	描述
0	reg_dvp_dbg_en	r/w	1'b0	DVP2BUS debug flag enable

#### 20.4.20 dvp\_dummy\_reg

地址: 0x300150fc

RESERVED

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

RESERVED

位	名称	权限	复位值	描述
31:0	RESERVED	rsvd	32'hf0f0f0f0	RESERVED

## 21.1 简介

MJPEG(Motion Joint Photographic Experts Group) 是一种视频编码格式，可精确到帧编辑和多层图像处理，把运动的视频序列作为连续的静止图像来处理，这种压缩方式单独完整地压缩每一帧。通过对 YCbCr 格式的原始数据进行压缩，可以大幅降低一帧图像所占用的内存空间。

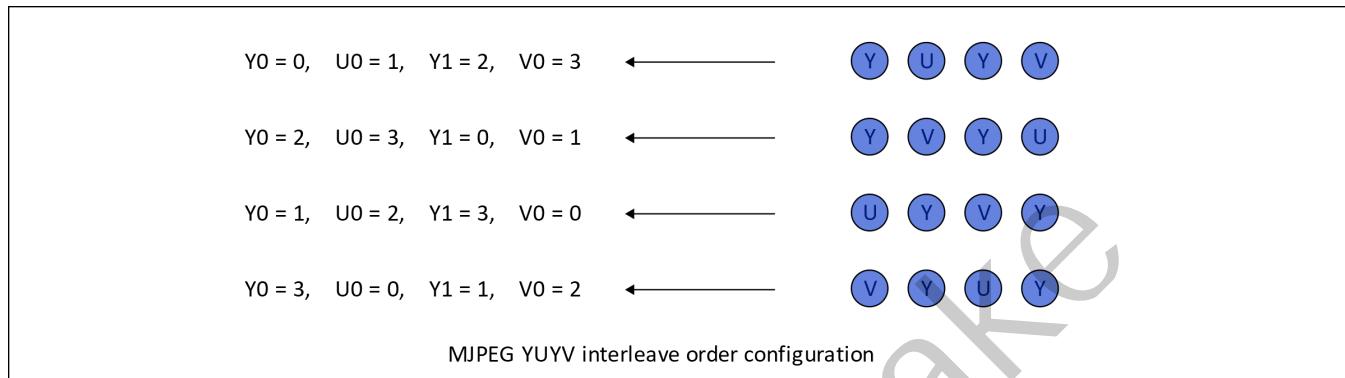
## 21.2 主要特征

- 可配置的输入格式，包括：
  - YCbCr4:2:2 平面或打包模式
  - YCbCr4:2:0 平面模式
  - YCbCr4:0:0
- 可配置的输入数据 Y、Cb、Cr 排列顺序
- 量化系数表可自由配置
- 支持软件模式和连动模式
- 支持 swap 模式
- 支持 kick 模式
- 可预留 jpg 头部空间和自动添加 jpg 尾
- 连续缓存多达 4 组图片信息
- 多种应用中断，有利于弹性使用与出错提示

## 21.3 功能描述

### 21.3.1 输入配置

通过寄存器 MJPEG\_CONTROL\_1 的位 <REG\_YUV\_MODE> 可以选择输入数据的 YCbCr 格式，包括 YCbCr4:2:2 平面或打包模式，YCbCr4:2:0 平面模式和 YCbCr4:0:0 灰度图。当选择 YCbCr4:2:2 打包模式时，通过寄存器 MJPEG\_HEADER\_BYTE 的高 8 位可以配置 Y、Cb、Cr 的排列顺序。在选择其他模式时，可以通过寄存器 MJPEG\_CONTROL\_1 的位 <REG\_ORDER\_U\_EVEN> 设定 Cb 和 Cr 的顺序。详细的配置如下图所示。



### 21.3.2 量化系数表

量化系数表可由用户自由配置，<reg\_q\_0\_00> 到 <reg\_q\_0\_3F> 表示灰度信息 Y 分量的量化表，<reg\_q\_1\_00> 到 <reg\_q\_1\_3F> 表示色度信息 Cb 和 Cr 的量化表。量化表按照从左上角向下排列的顺序，第一列结束后紧跟第二列，即 reg\_q\_0\_00 代表 Y 分量第一行第一列的量化数值，reg\_q\_0\_01 代表 Y 分量第二行第一列的量化数值，reg\_q\_0\_07 代表 Y 分量第八行第一列的量化数值，reg\_q\_0\_08 代表 Y 分量第一行第二列的量化数值，依次类推。

### 21.3.3 软件模式和连动模式

将 MJPEG\_CONTROL\_2 寄存器的位 <REG\_MJPEG\_SW\_MODE> 置 1 时即使能软件模式，在此模式下 MJPEG 会从内存中读取指定帧数的原始图片数据进行压缩，此模式下需要预先将图片数据准备好。

将 MJPEG\_CONTROL\_2 寄存器的位 <REG\_MJPEG\_SW\_MODE> 清 0 时即使能连动模式，在此模式下 MJPEG 会将 CAM 模块的输出作为其输入进行处理，MJPEG 是以 8\*8 的数据块为一个单元进行处理的，当 CAM 向内存中写完 8 行数据后，MJPEG 就会开始工作，MJPEG 处理完的内存空间会释放给 CAM 重新使用，这样 CAM 无需一整张图片的内存空间即可完成与 MJPEG 的连动。

### 21.3.4 swap 模式

使用 **swap** 模式时, MJPEG 存储空间会被平均分成两块, 当 MJPEG 写完其中一块时会产生一个中断通知软件将数据读走, 而它会将数据写入另一块中。来回交替使用, 从而用不足一帧图片的存储空间进行数据处理。

### 21.3.5 kick 模式

将 MJPEG\_CONTROL\_2 寄存器的位 <REG\_SW\_KICK\_MODE> 置 1 时即使能 kick 模式, 在此模式下每次向 MJPEG\_CONTROL\_2 寄存器的位 <REG\_SW\_KICK> 写 1 即可启动一次压缩, 压缩行数由 MJPEG\_YUV\_MEM\_SW 寄存器的位 <REG\_SW\_KICK\_HBLK> 确定。需要注意的是 kick 模式只能在软件模式下使用, 不能在连动模式下使用。

### 21.3.6 jpg 功能

jpg 功能会自动在每帧数据的开头留出一定字节数的空间并填充指定数据, 该空间的大小由寄存器 MJPEG\_HEADER\_BYTEx 的低 12 位进行设置。在偏移 0x800 处有 768 字节的内存用于存放需要填充在每帧开头的数据, 用户只需将 jpg 头部信息写入其中, 则生成的每帧数据开头都会包含这段信息。另外如果使能自动填充 jpg 尾的话还会在每帧数据的结尾补上两个字节数据, 这两个字节的值为 0xFFD9。

### 21.3.7 缓存图片信息

模块内部包含 4 组 FIFO 记录图片地址、图片大小和图片 ID。每当此模块完整写入一帧到内存, 便会将此帧图片的起始地址、图片大小和图片 ID 纪录于此 FIFO 中, 但要注意的是当发生内存剩余不足, 或是 4 组 FIFO 满存的状况时, 模块会自动丢掉接下来图片的讯息, 在图片信息取出的部分, 可通过 APB 接口做 pop 的动作, 将最旧的图片信息空出, 此时 FIFO 会自动推进, 保证 FIFO 内部图片信息的时序。

### 21.3.8 支持多种中断信息 (可独立开关配置)

- A、Normal 中断-可设定固定写入几张图片后发出中断
- B、Camera 中断-MJPEG 处理的速度跟不上 CAM 模块写入的速度导致 CAM 溢出时, 发出中断
- C、Memory 中断-当内存被复写时, 发出中断
- D、Frame 中断-当未处理图片超过 4 组时, 发出中断
- E、Swap 中断 - 当 swap 模式下一个内存块被写满时, 发出中断

## 22.1 简介

DBI 是显示总线接口 (Display Bus Interface) 的简称，也被称为 MCU 接口，是 MIPI 联盟定义的用于与显示屏通信的接口协议。DBI 协议的数据线和控制线是复用的，驱动的显示模块需要带有 GRAM。DBI 又可以细分为 MIPI DBI Type A、MIPI DBI Type B 和 MIPI DBI Type C 三种不同的模式，不同模式下的硬件接口以及数据采样都有所不同。如 MIPI DBI Type A 是下降沿采样 (基于 Motorola 6800 总线)，而 MIPI DBI Type B 是上升沿采样 (基于 Intel 8080 总线)。该模块另外还集成了 QSPI(Quad Serial Peripheral Interface) 显示接口，QSPI 有四根数据线，是对 SPI 接口的扩展，极大提高了传输效率。

## 22.2 主要特征

- 符合 MIPI® 联盟标准
- 支持 Type B 和 Type C 模式：
  - Type B 为 8 位数据接口
  - Type C 支持 3-Line 和 4-Line
- 额外支持 QSPI 模式，QSPI 模式下特征如下：
  - 一次传输包含一字节命令阶段、一到三字节可调的地址阶段、可选的数据阶段
  - 命令、地址和数据读写分别都支持 1 线/4 线模式，可以任意组合
- 除 QSPI 模式外，一次传输都支持可选的命令与数据阶段，QSPI 接口有另外的地址阶段
- 数据阶段可设置为以字节为单位的 **normal** 模式和以像素为单位的 **pixel** 模式：
  - 数据阶段 **normal** 模式用于传输配置数据，单次最多可写入 256 个字节，最多可读出 8 个字节
  - 数据阶段 **pixel** 模式用于传输像素数据，数据量以像素为单位，单次最多可以写 2 的 24 次方个像素，不可读
- 输入像素格式支持 RGB 和 YUV444，其中 RGB 支持以下八种内存排列方式：

- RGBA8888
- BGRA8888
- ARGB8888
- ABGR8888
- RGB888
- BGR888
- RGB565
- BGR565

- YUV444 支持以下六种内存排列方式：

- YUVN8888
- VUYN8888
- NYUV8888
- NVUY8888
- YUV888
- VUY888

- YUV444 格式到 RGB565/666/888 格式的硬件转码

- 输出像素格式支持 RGB565、RGB666 和 RGB888

- CS 信号拉高释放条件可配置

- 多种中断控制

- 支持 DMA 传输模式

## 22.3 功能描述

DBI 模块基本框图如图所示。

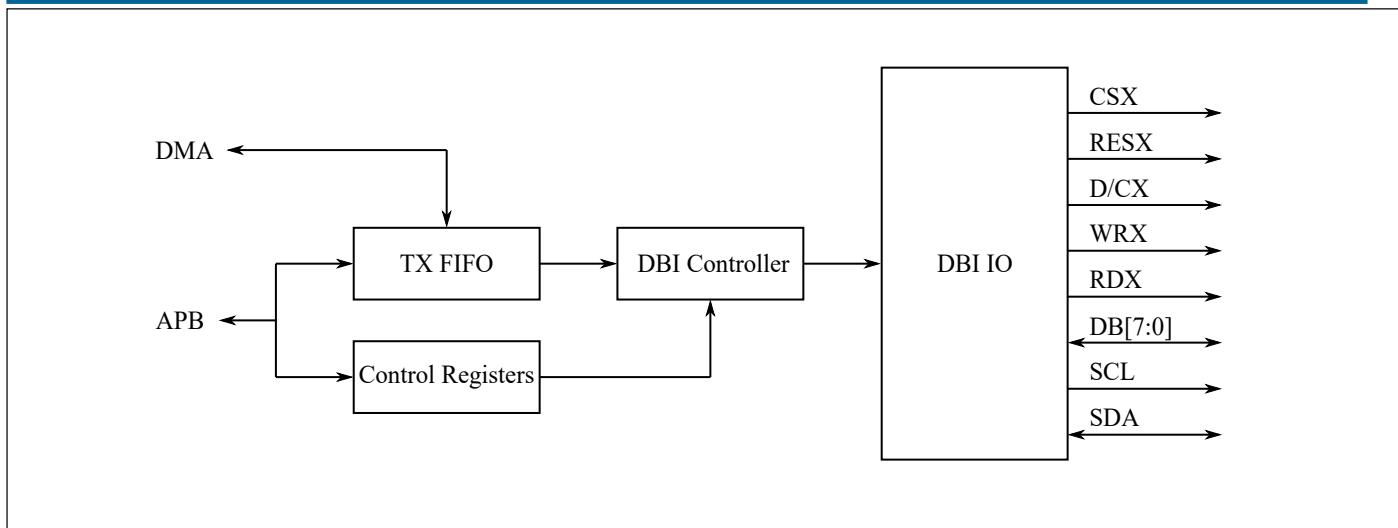


图 22.1: DBI 基本框图

### 22.3.1 DBI Type B

#### 22.3.1.1 写时序

DBI Type B 模式拥有 8 位并行数据线，MCU 向显示模块写入数据的时序如下图所示：

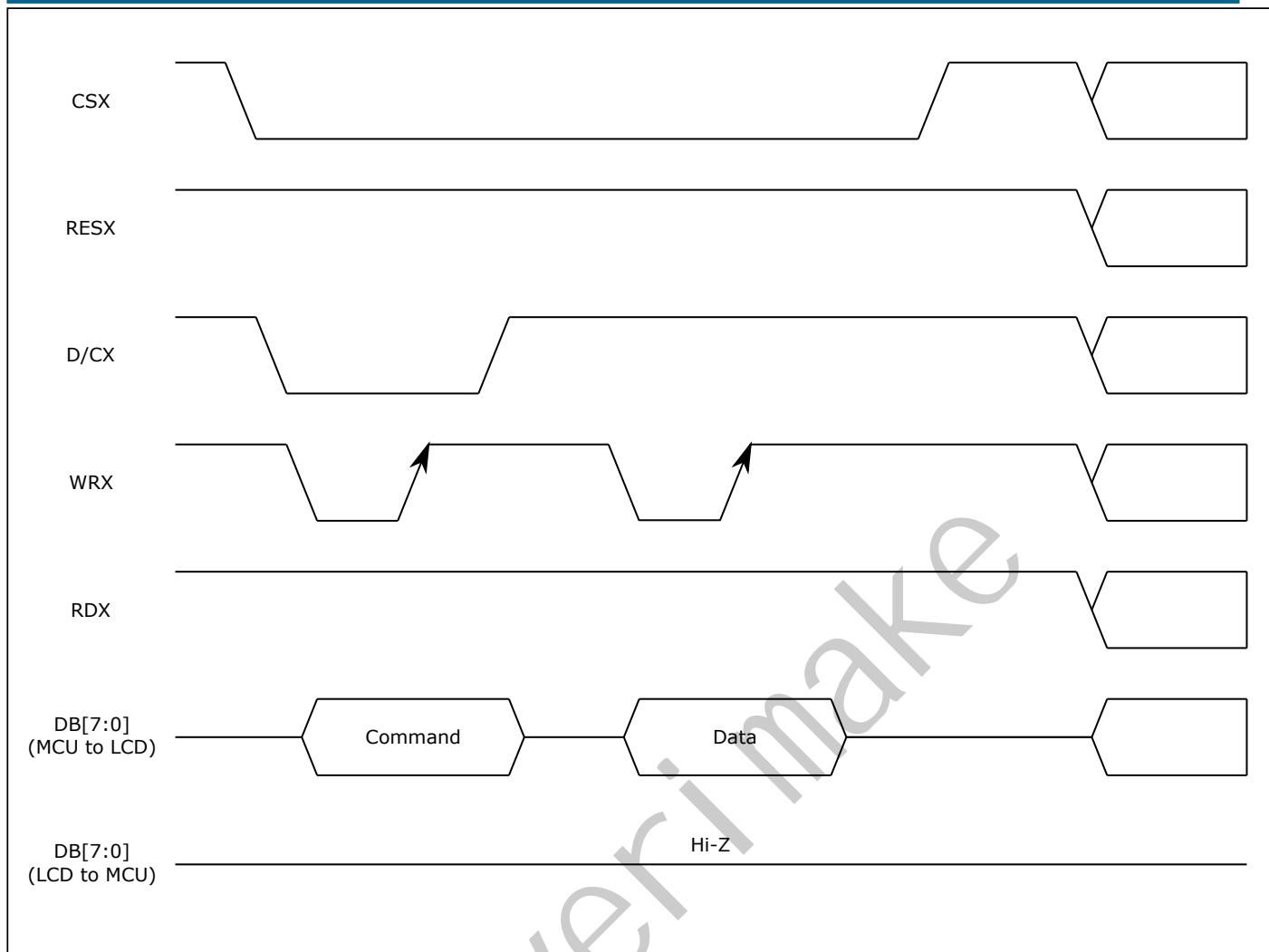


图 22.2: 写时序

其中：

- **CSX:** 片选信号。当该信号为低时显示模块被选中，为高时显示模块将忽略其他所有的接口信号；
- **RESX:** 外部复位信号。当该信号为低时显示模块被复位；
- **D/CX:** 数据/命令选择信号。当该信号为 0 时，DB[7:0] 位为命令；当该信号为 1 时，DB[7:0] 位为 RAM 数据或命令参数；
- **WRX:** 并行数据写入选通信号。该信号在写入周期内从高到低驱动，然后拉回到高。显示模块在该信号的上升沿读取 MCU 传输的信息。该信号是一种非同步信号，可在不使用时终止；
- **RDX:** 并行数据读取选通信号。该信号从高到低被驱动，然后在读取周期中被拉回到高。MCU 在该信号的上升沿读取显示模块传输的信息。该信号是一种非同步信号，可在不使用时终止；
- **DB[7:0]:** 8 位数据信号。用于传输命令、命令参数或数据。

## 22.3.1.2 读时序

MCU 从显示模块读取数据的时序如下图所示：

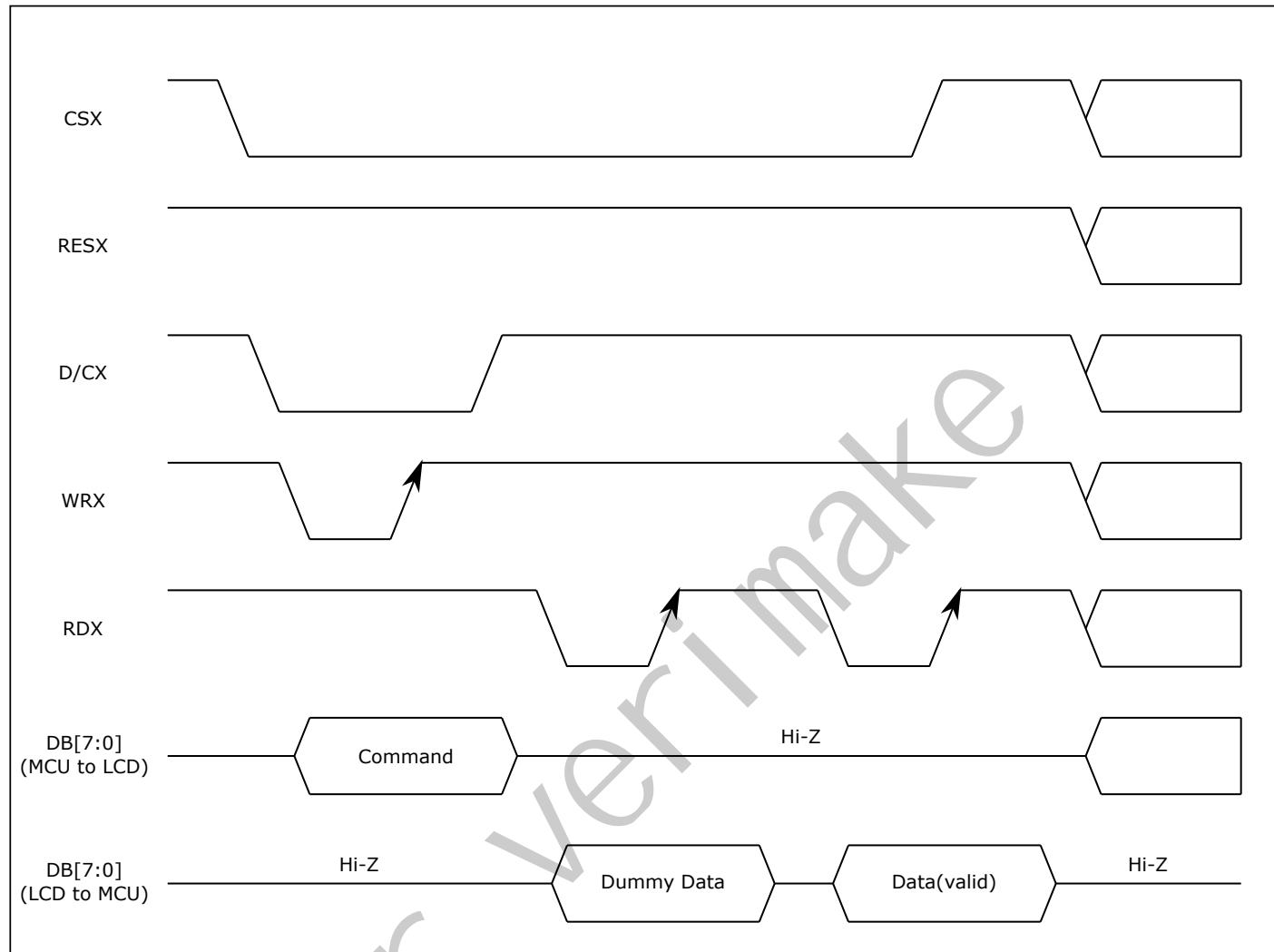


图 22.3: 读时序

## 22.3.1.3 输出 RGB565

RGB565 格式数据输出如下图所示：

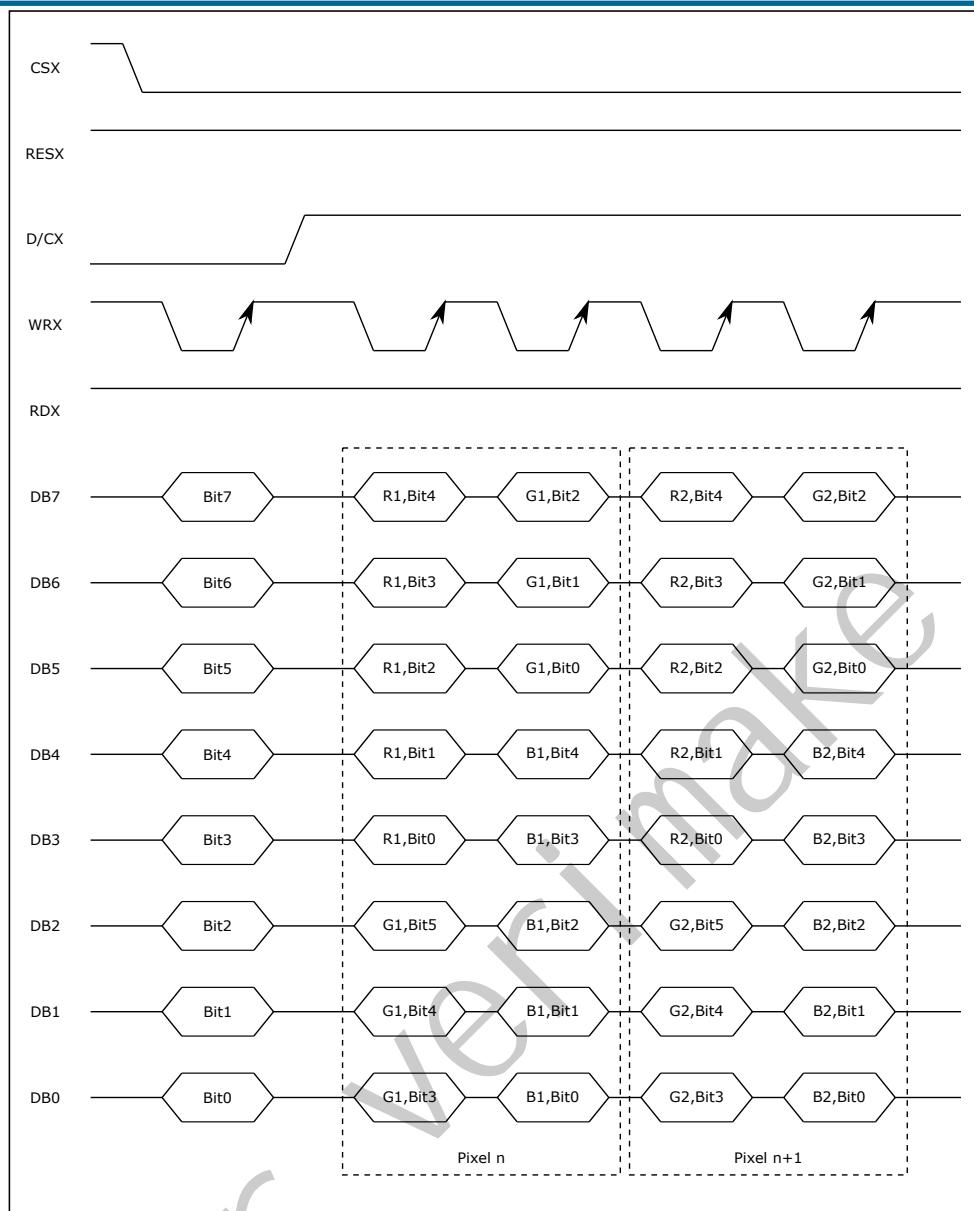


图 22.4: RGB565 输出

#### 22.3.1.4 输出 RGB666

RGB666 格式数据输出如下图所示：

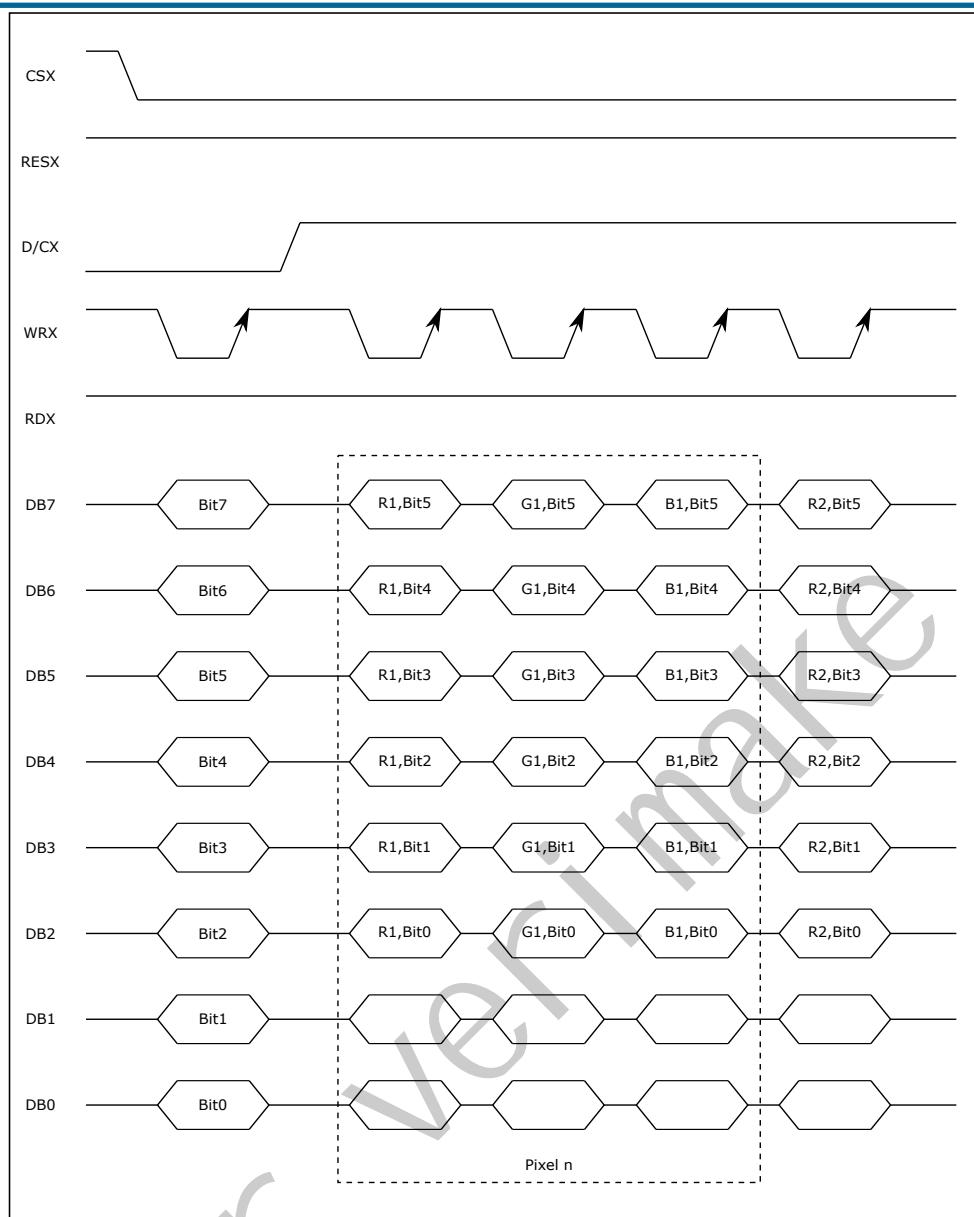


图 22.5: RGB666 输出

### 22.3.1.5 输出 RGB888

RGB888 格式数据输出如下图所示：

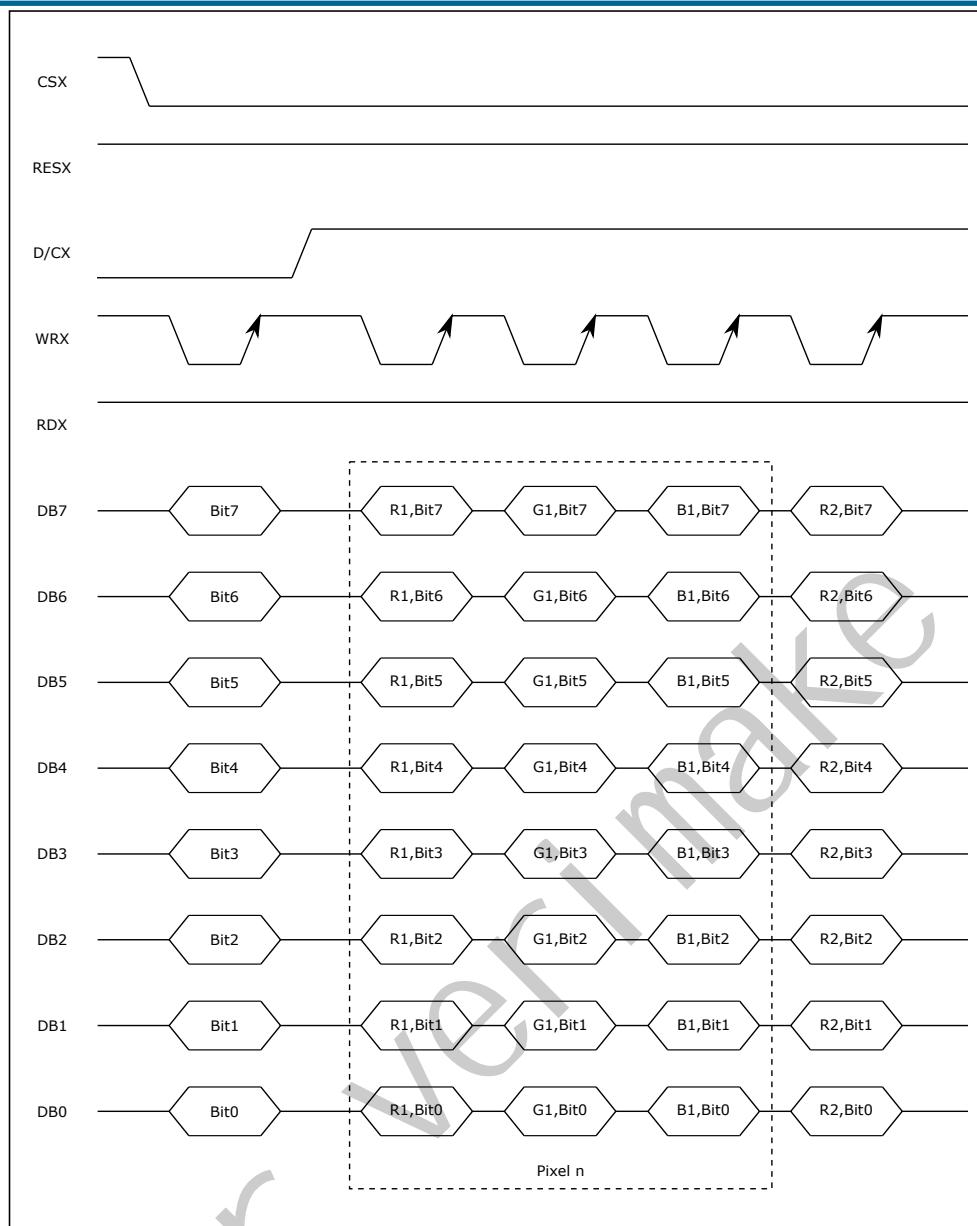


图 22.6: RGB888 输出

### 22.3.2 DBI Type C 3-Line

#### 22.3.2.1 写时序

DBI Type C 模式是 3 线 9 位串行接口，MCU 向显示模块写入数据的时序如下图所示：

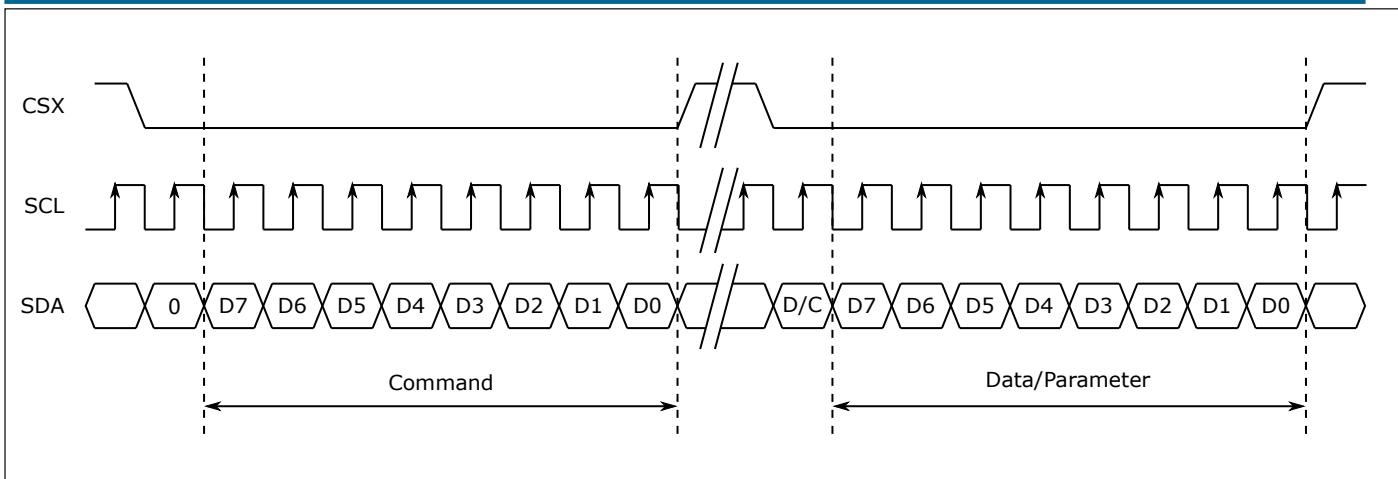


图 22.7: 写时序

其中：

- CSX：片选信号。当该信号为低时显示模块被选中，为高时显示模块将忽略其他所有的接口信号；
- SCL：串行时钟输入信号。用于为数据传输提供时钟信号。
- SDA：串行数据输入/输出信号。在写操作中，串行数据包包含一个 D/CX(数据/命令) 选择位和一个传输字节。如果 D/CX 位为低，则传输字节为命令；如果 D/CX 位为高，则传输字节为显示数据或命令参数。

### 22.3.2.2 读时序

MCU 从显示模块读取数据的时序如下图所示：

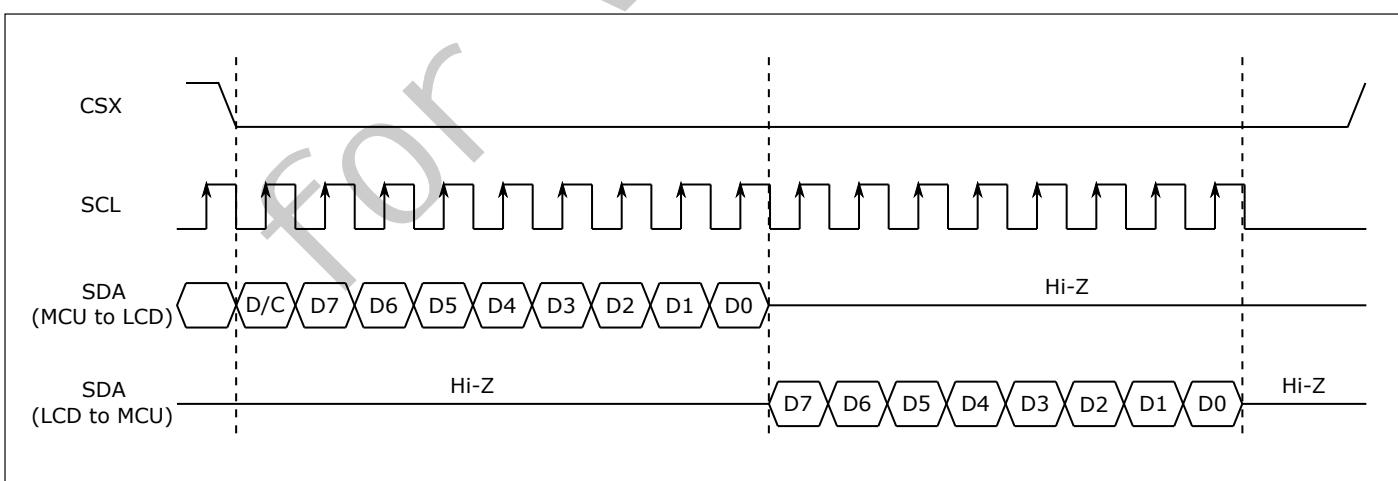


图 22.8: 读时序

### 22.3.2.3 输出 RGB565

RGB565 格式数据输出如下图所示：

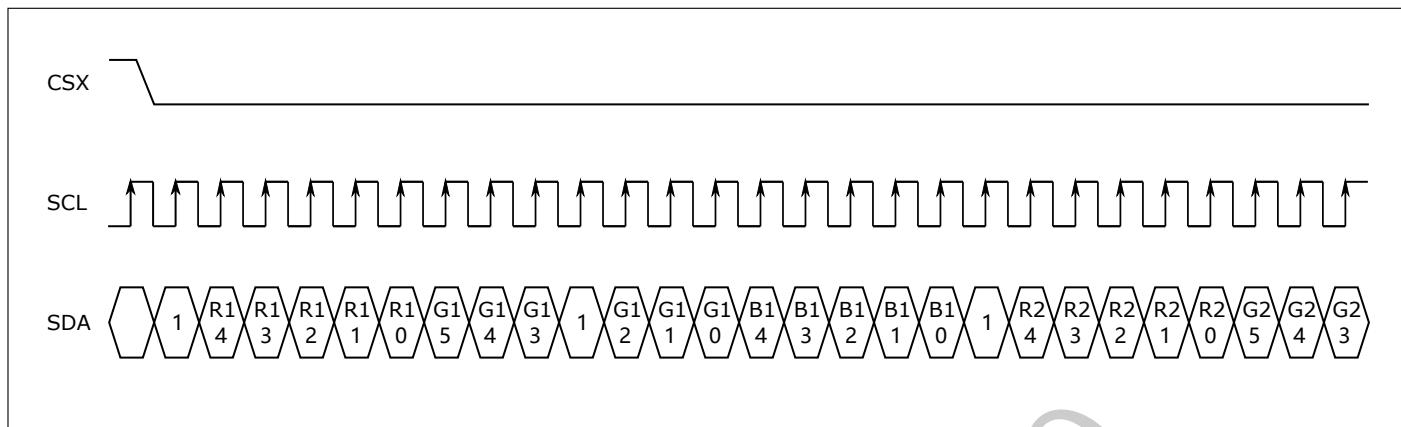


图 22.9: RGB565 输出

### 22.3.2.4 输出 RGB666

RGB666 格式数据输出如下图所示：

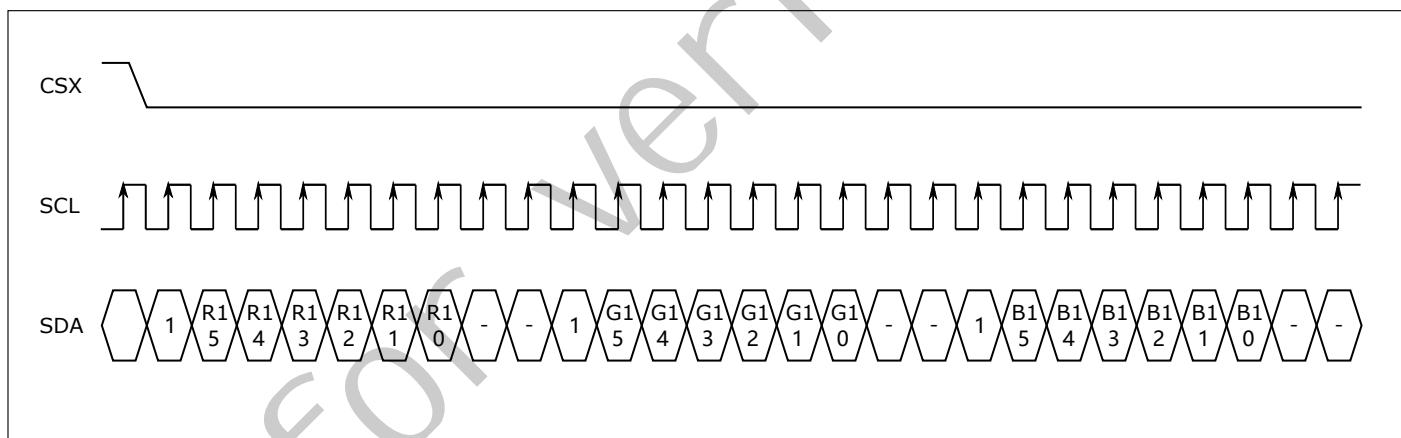


图 22.10: RGB666 输出

### 22.3.2.5 输出 RGB888

RGB888 格式数据输出如下图所示:

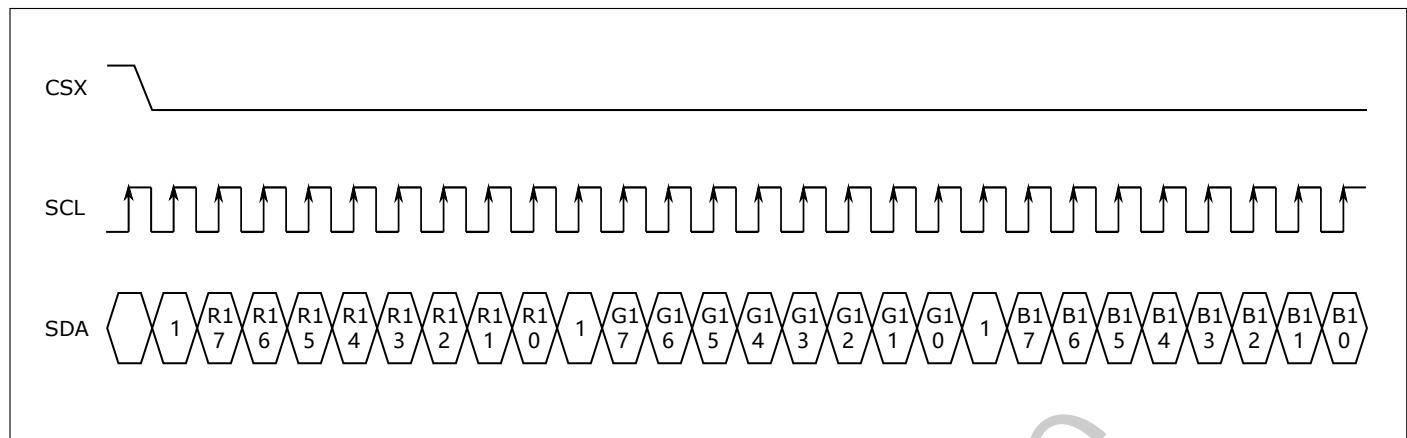


图 22.11: RGB888 输出

### 22.3.3 DBI Type C 4-Line

#### 22.3.3.1 写时序

DBI Type C 模式是 4 线 8 位串行接口，MCU 向显示模块写入数据的时序如下图所示:

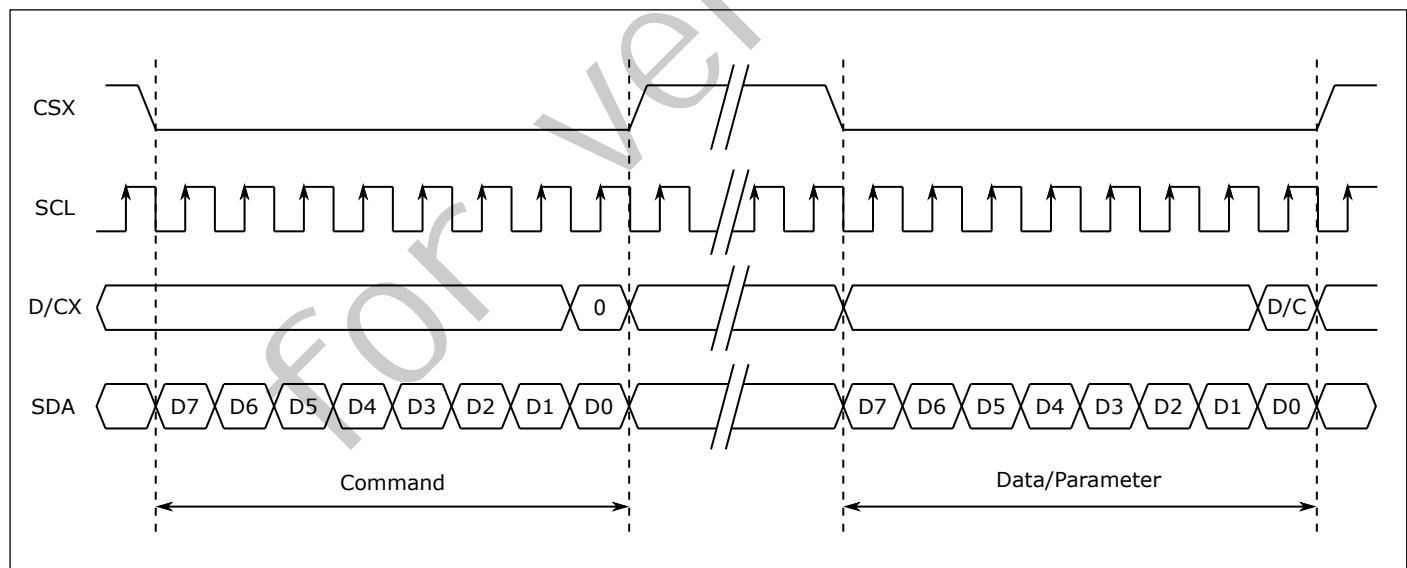


图 22.12: 写时序

其中：

- **CSX:** 片选信号。当该信号为低时显示模块被选中，为高时显示模块将忽略其他所有的接口信号；
- **D/CX:** 数据/命令选择信号。如果该信号为低，则 SDA 传输的信息为命令；如果该信号为高，则 SDA 传输的信息

为显示数据或命令参数。

- SCL: 串行时钟输入信号。用于为数据传输提供时钟信号。
- SDA: 串行数据输入/输出信号。用于传输命令、命令参数或数据。

### 22.3.3.2 读时序

MCU 从显示模块读取数据的时序如下图所示：

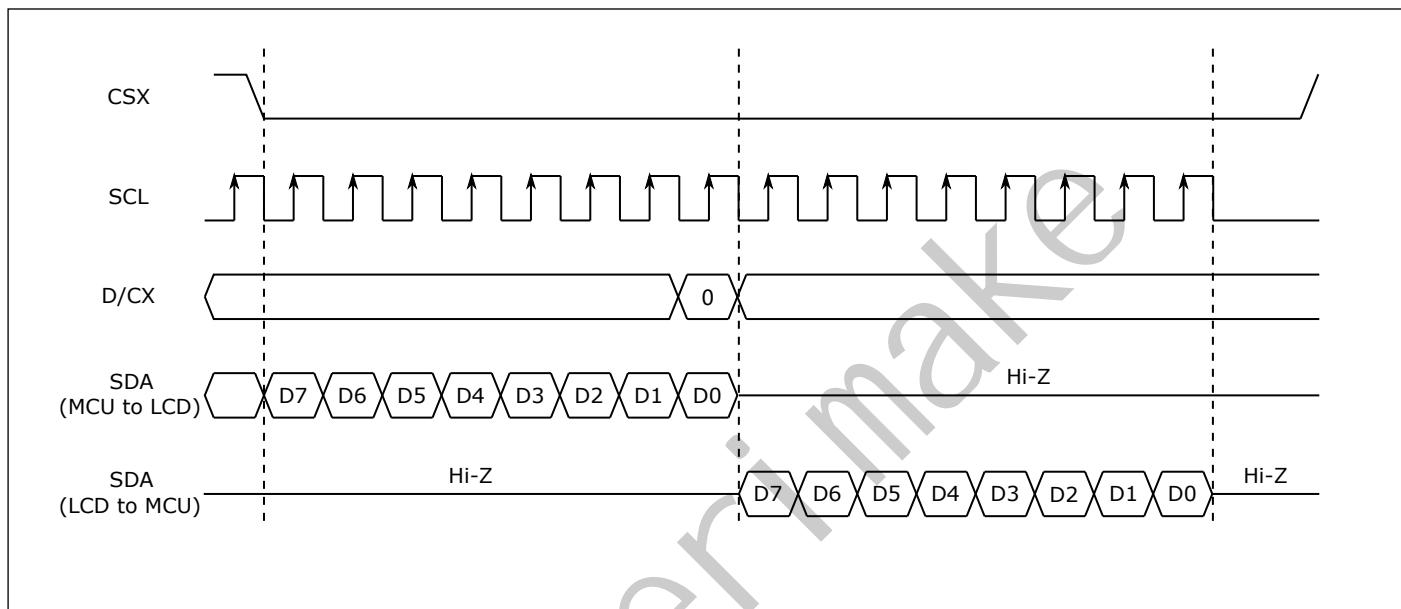


图 22.13: 读时序

### 22.3.3.3 输出 RGB565

RGB565 格式数据输出如下图所示：

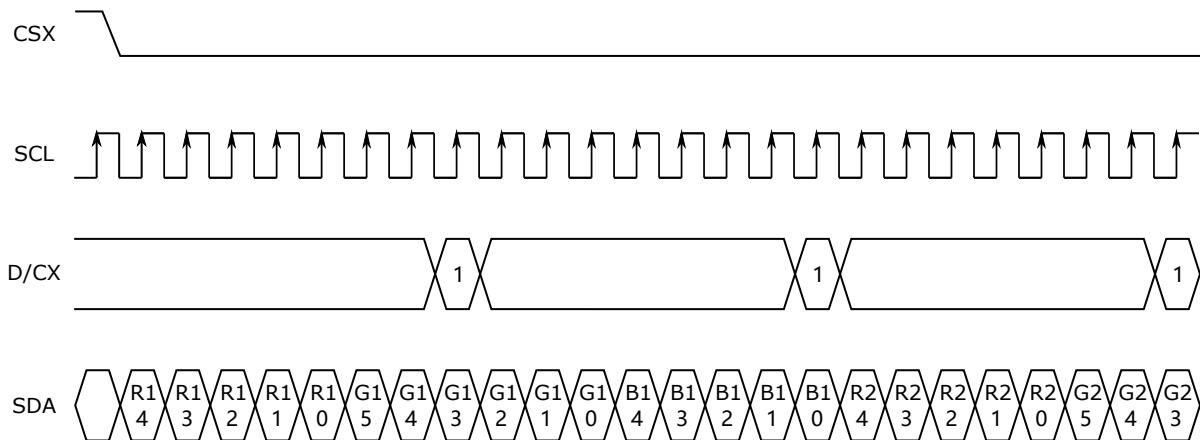


图 22.14: RGB565 输出

#### 22.3.3.4 输出 RGB666

RGB666 格式数据输出如下图所示：

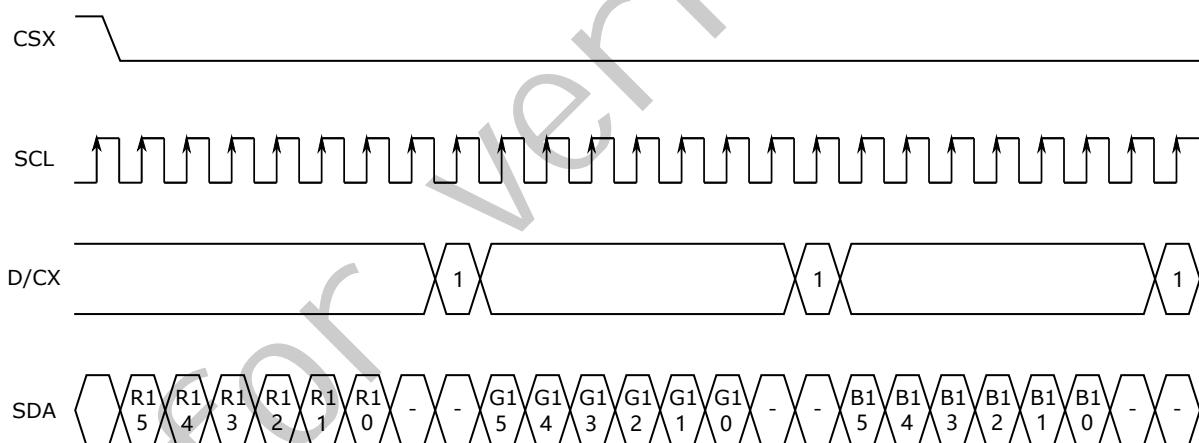


图 22.15: RGB666 输出

### 22.3.3.5 输出 RGB888

RGB888 格式数据输出如下图所示：

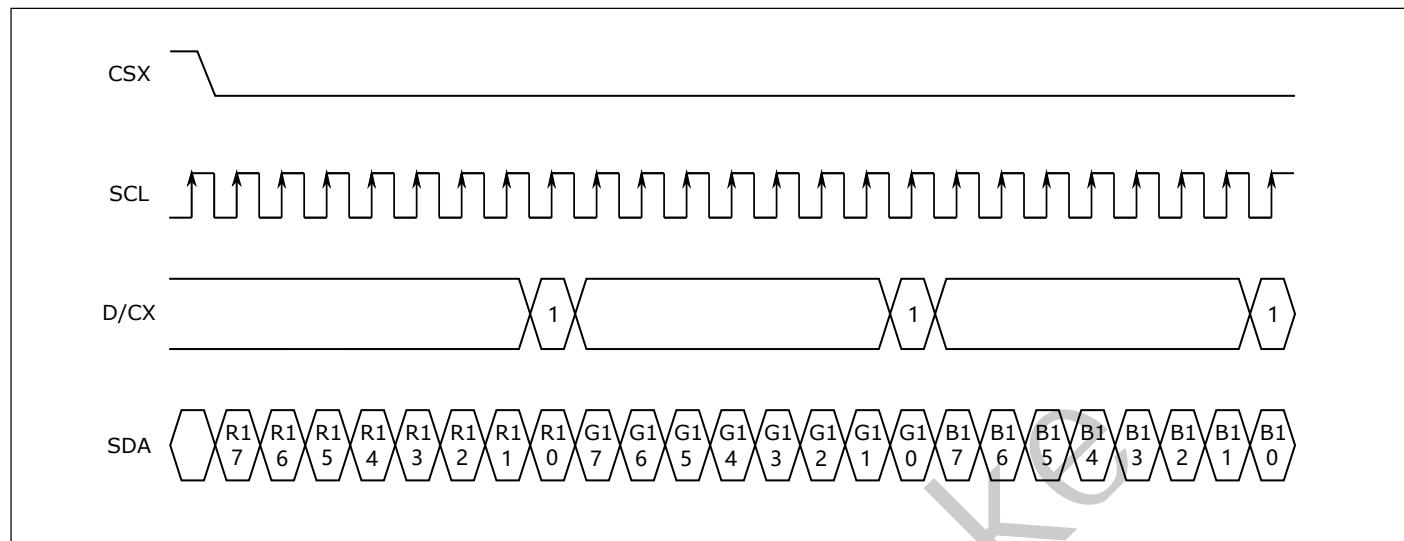


图 22.16: RGB888 输出

## 22.3.4 QSPI

### 22.3.4.1 写时序

以 1 线命令、1 线 3 字节长度地址、1 线数据模式为例，MCU 向显示模块写入数据的时序如下图所示：

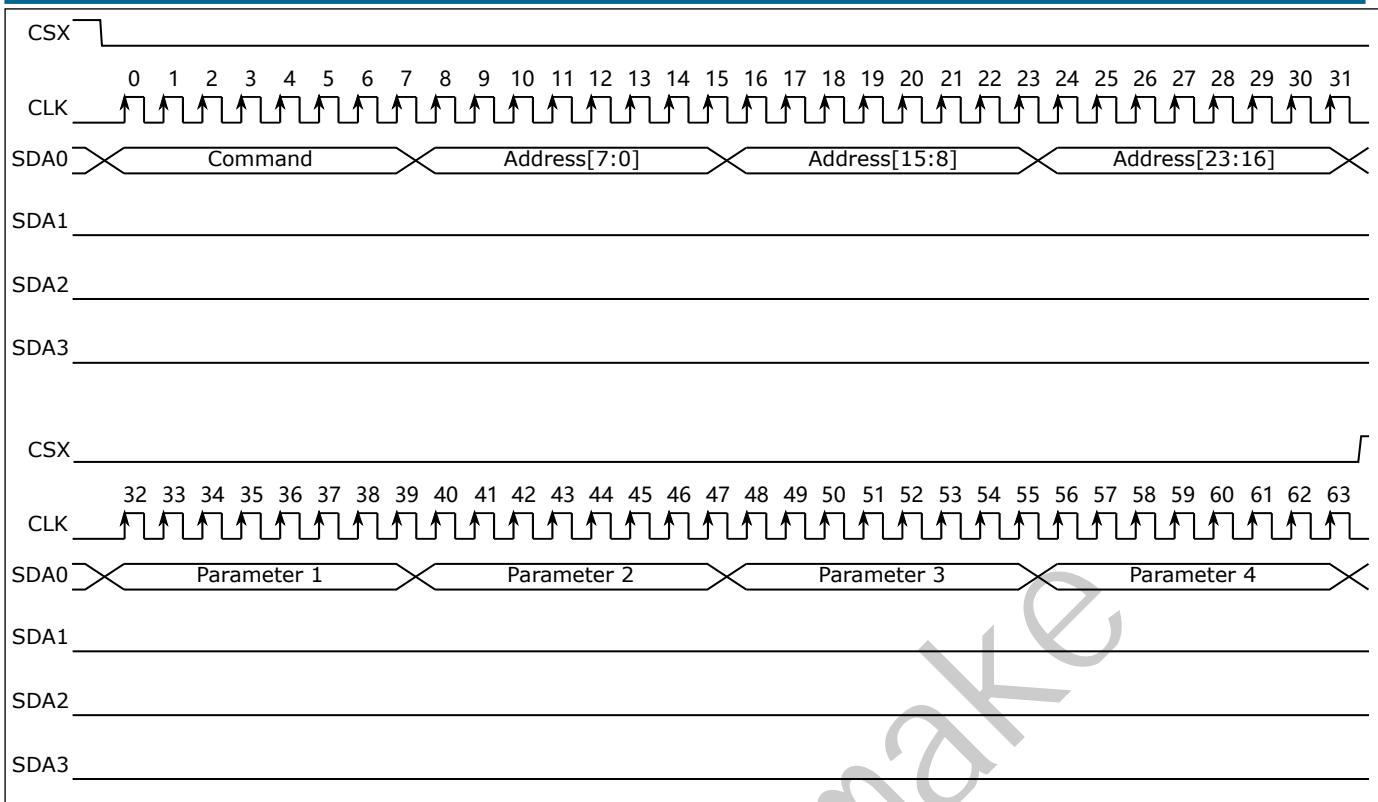


图 22.17: 写时序

其中：

- **CSX:** 片选信号。当该信号为低时显示模块被选中，为高时显示模块将忽略其他所有的接口信号；
- **CLK:** 时钟输入信号。用于为数据传输提供时钟信号。
- **SDA0:** 数据线 0，用于传输命令、地址或数据。
- **SDA1:** 数据线 1，用于传输命令、地址或数据。
- **SDA2:** 数据线 2，用于传输命令、地址或数据。
- **SDA3:** 数据线 3，用于传输命令、地址或数据。

### 22.3.4.2 读时序

以 1 线命令、1 线 3 字节长度地址、1 线数据模式为例，MCU 从显示模块读取数据的时序如下图所示：

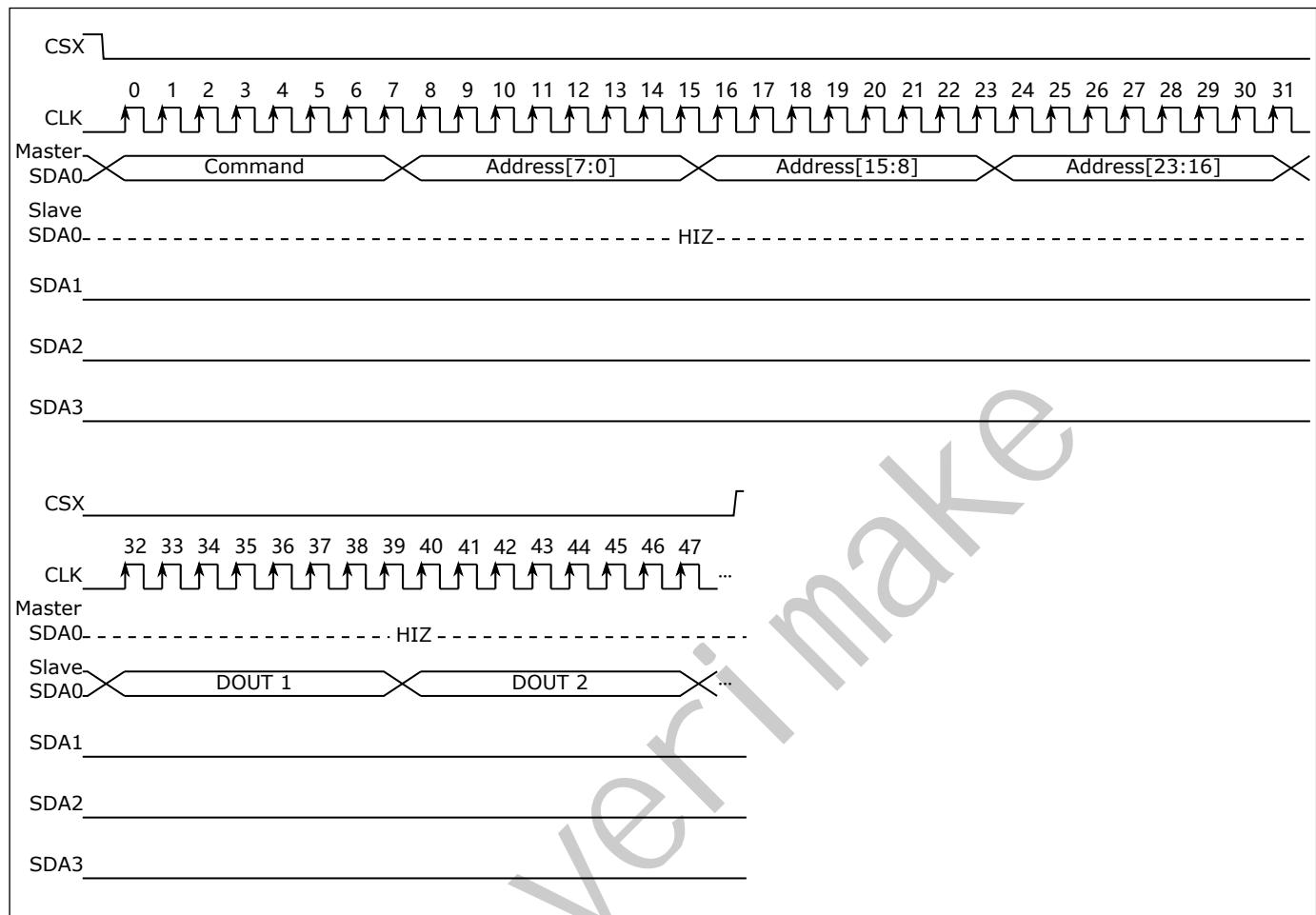


图 22.18: 读时序

### 22.3.4.3 1 线命令、1 线地址、1 线数据模式

以命令为 0x02，地址为 3 字节 0x002C00/0x003C00 为例，1 线命令、1 线地址、1 线数据模式下 RGB565 格式数据输出如下图所示：

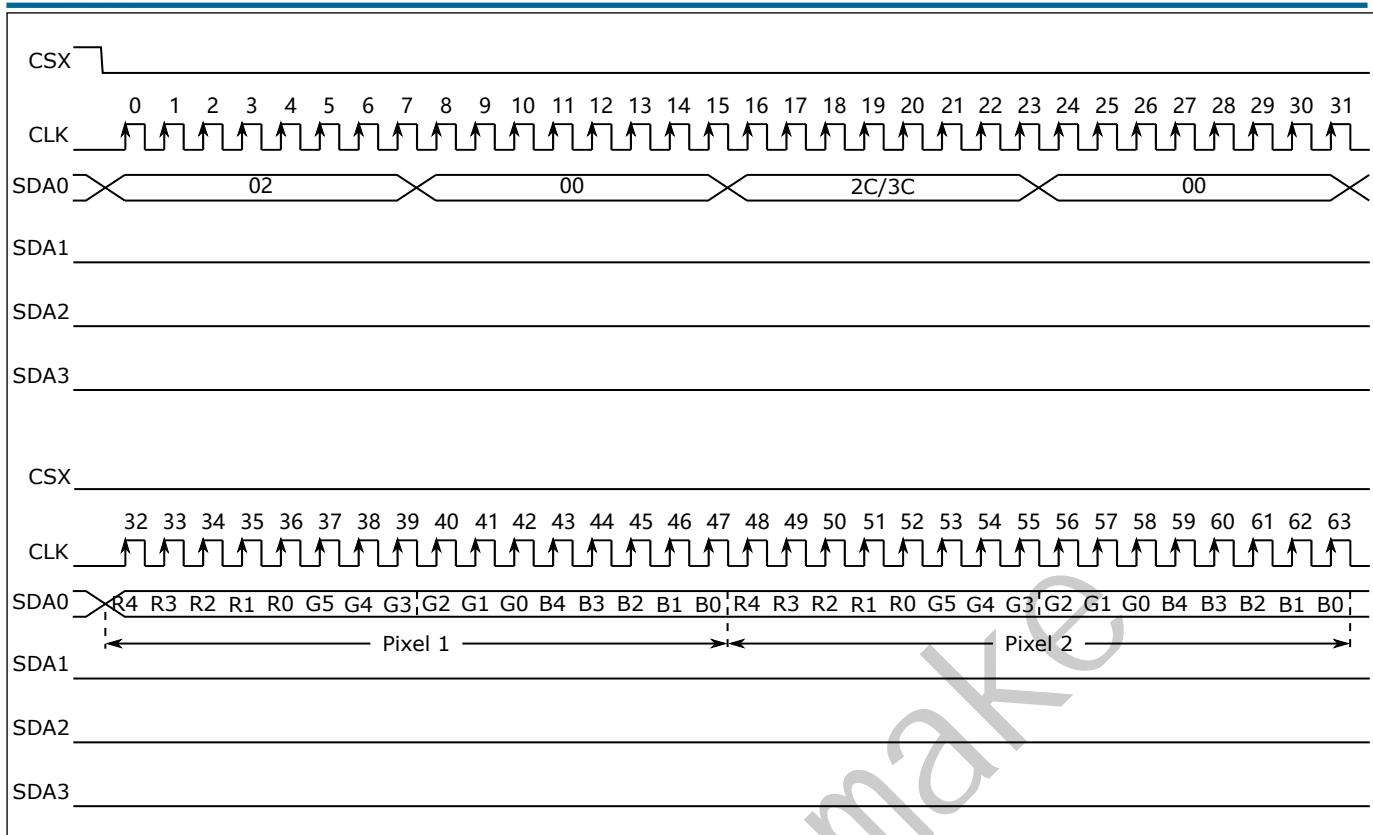


图 22.19: RGB565 输出

RGB666 格式数据输出如下图所示：

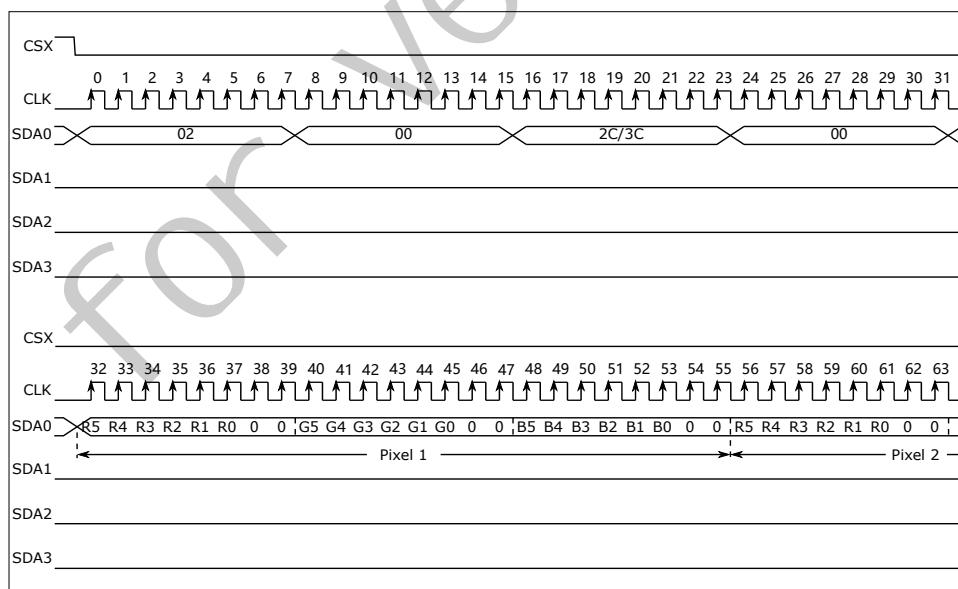


图 22.20: RGB666 输出

RGB888 格式数据输出如下图所示：

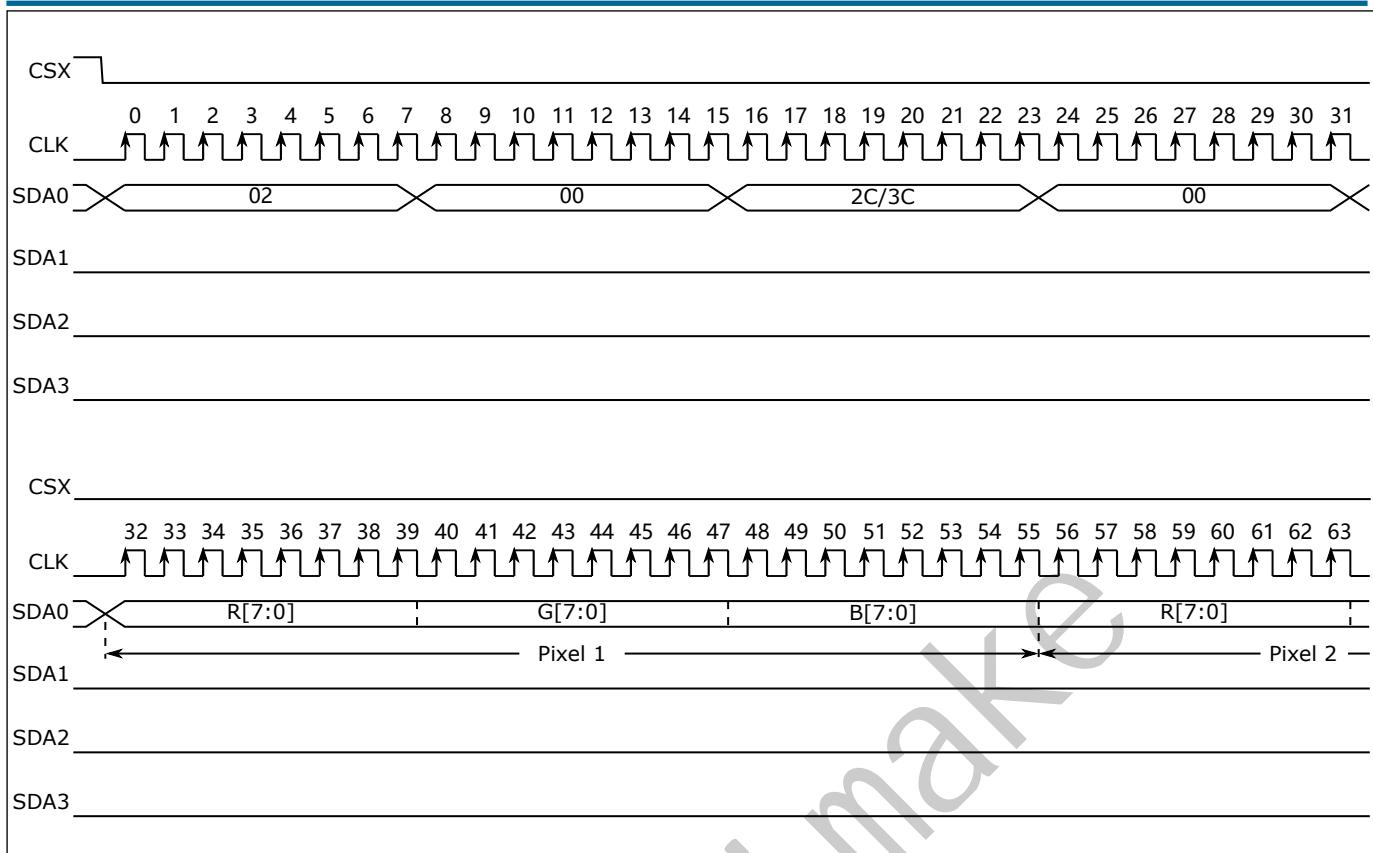


图 22.21: RGB888 输出

#### 22.3.4.4 1 线命令、1 线地址、4 线数据模式

以命令为 0x32，地址为 3 字节 0x002C00/0x003C00 为例，1 线命令、1 线地址、4 线数据模式下 RGB565 格式数据输出如下图所示：

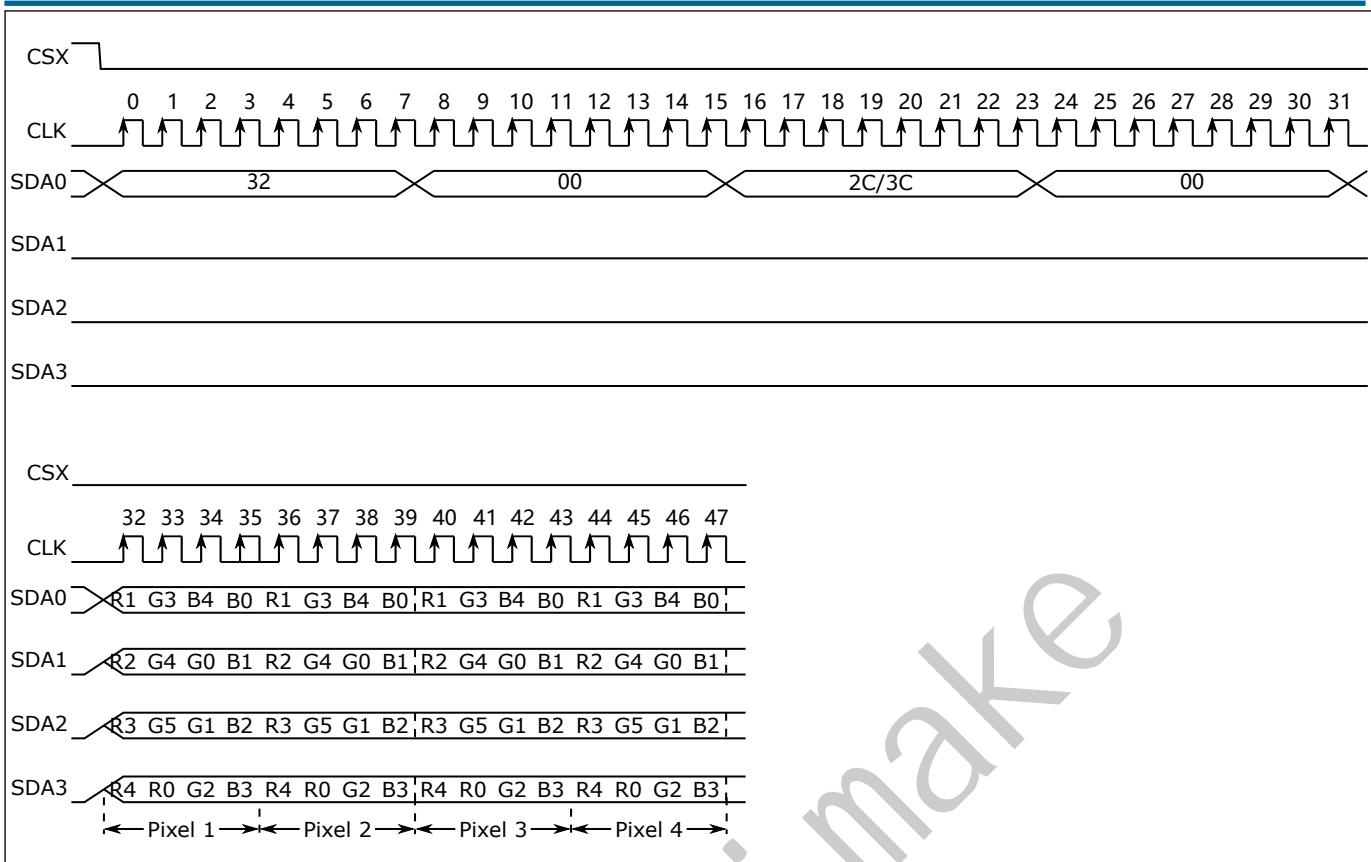


图 22.22: RGB565 输出

RGB666 格式数据输出如下图所示：

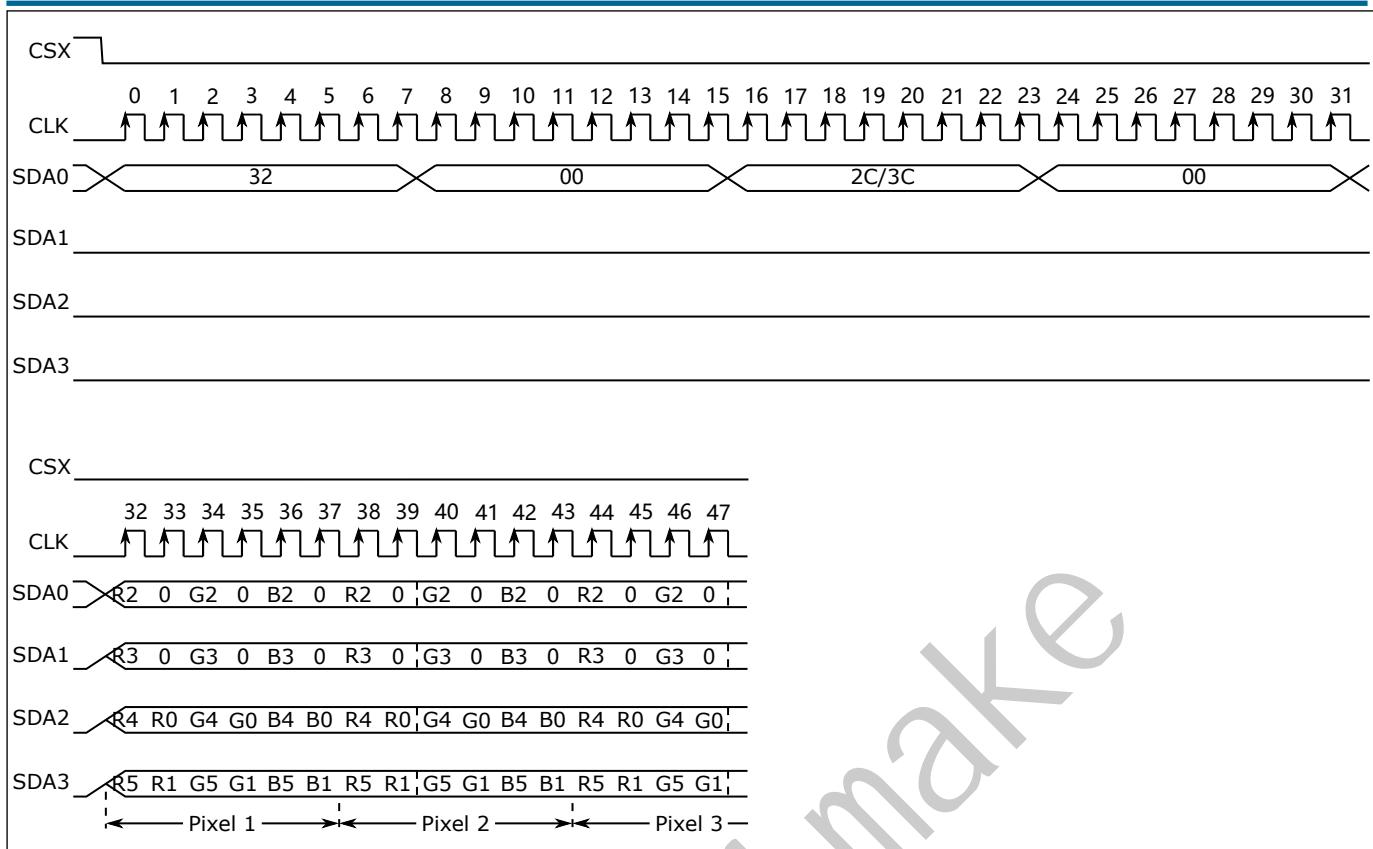


图 22.23: RGB666 输出

RGB888 格式数据输出如下图所示:

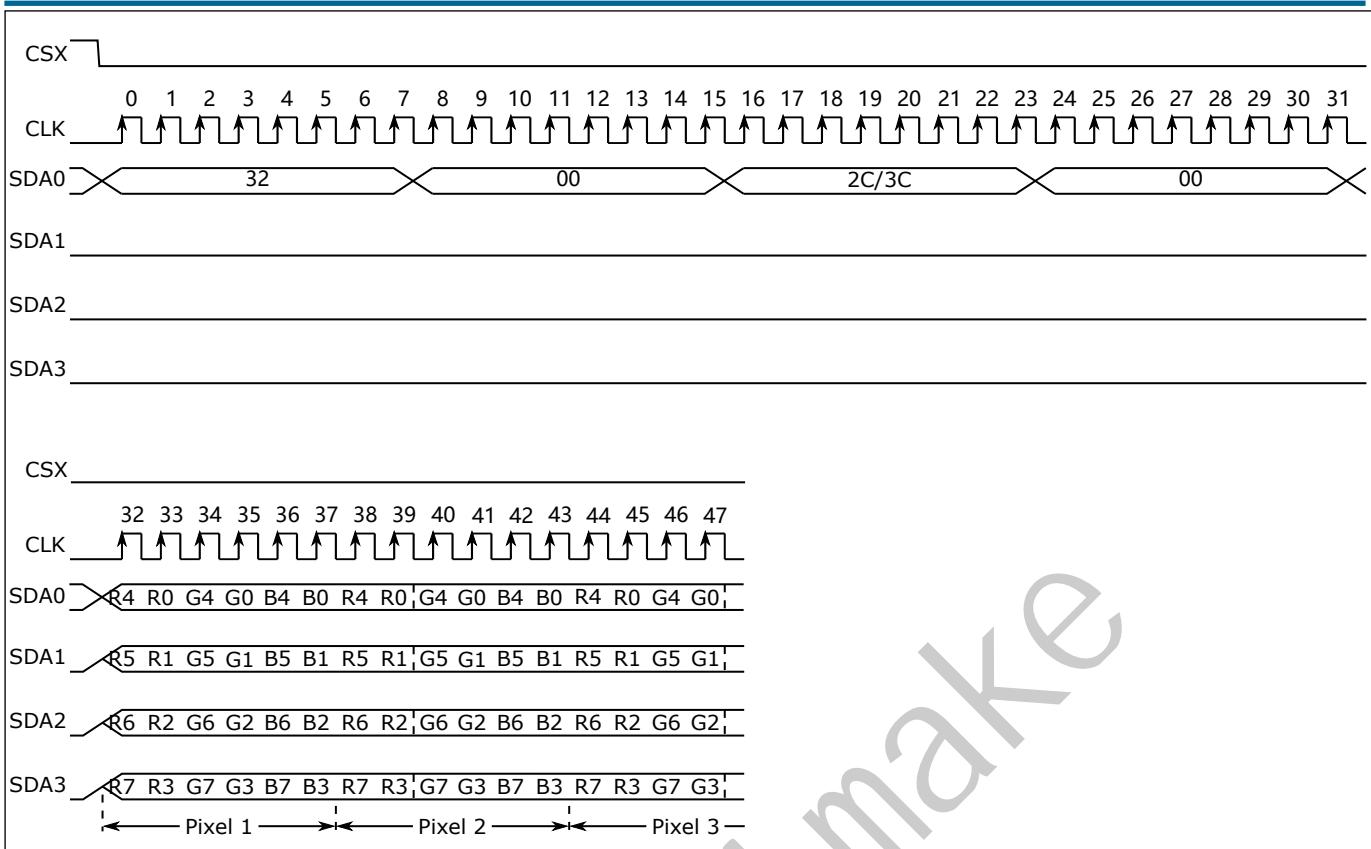


图 22.24: RGB888 输出

#### 22.3.4.5 1 线命令、4 线地址、4 线数据模式

以命令为 0x12，地址为 3 字节 0x002C00/0x003C00 为例，1 线命令、4 线地址、4 线数据模式下 RGB565 格式数据输出如下图所示：

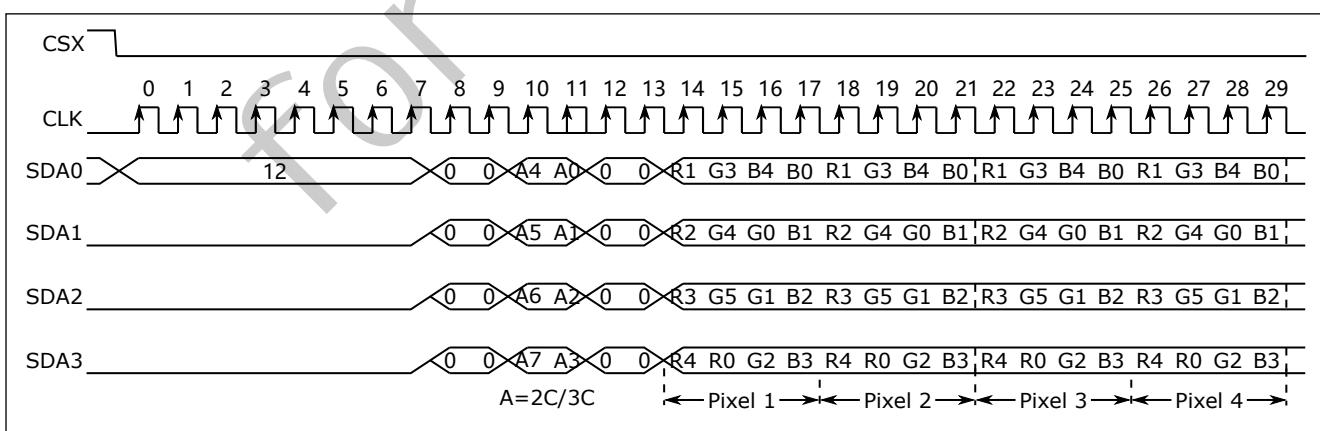


图 22.25: RGB565 输出

RGB666 格式数据输出如下图所示：

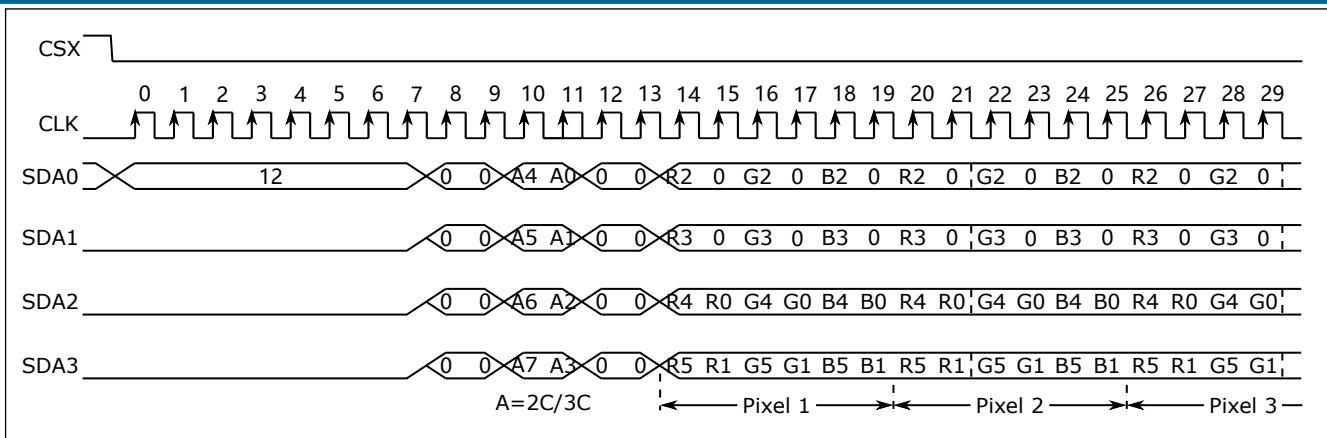


图 22.26: RGB666 输出

RGB888 格式数据输出如下图所示：

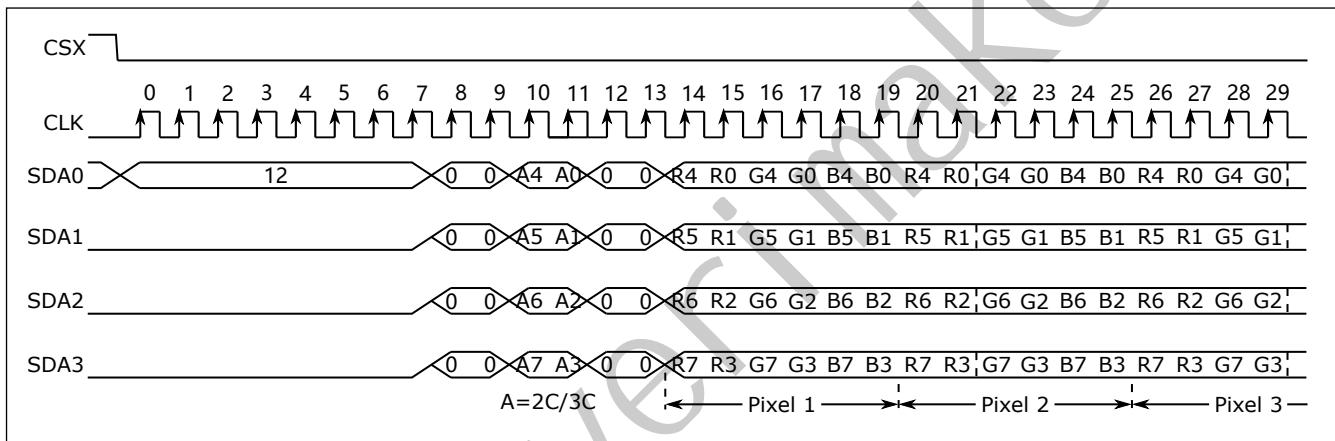


图 22.27: RGB888 输出

### 22.3.5 输入像素格式

DBI 模块支持 8 种不同的输入像素 RGB 格式和 6 种 YUV444 格式，各种 RGB 格式在内存中的排布如下图所示：

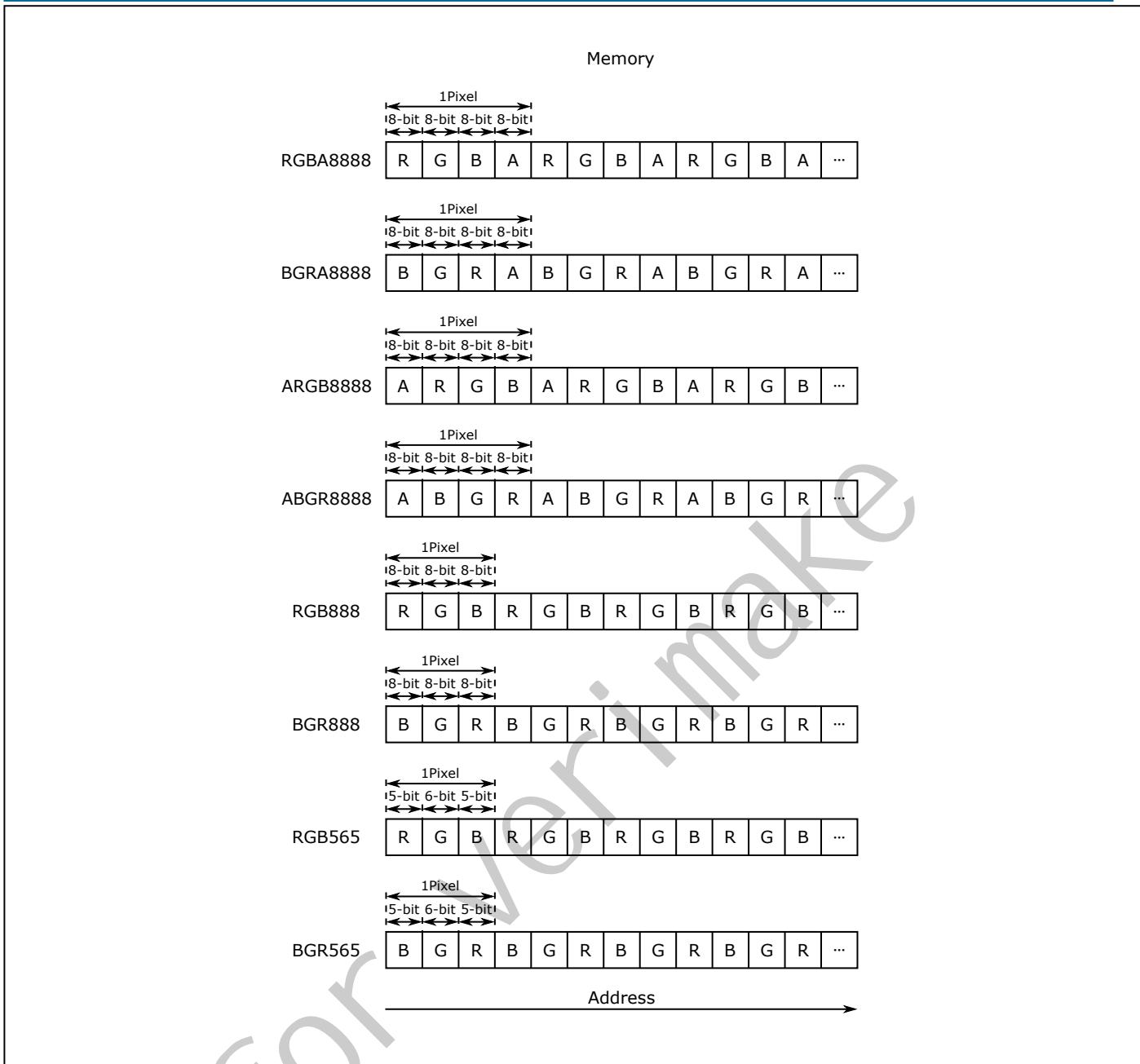


图 22.28: 输入像素 RGB 格式

对于 YUV444 格式的内存排布，只需用 Y 替换 R，用 U 替换 G，用 V 替换 B 即可。

### 22.3.6 CS 信号拉高释放条件配置

在 Type B 和 Type C 模式下，可以通过寄存器 CONFIG 的位 CONT\_EN 配置 CS 信号的拉高释放条件。如果不使能该功能，CS 信号会在每个像素传输完成后释放一次，即每两个像素之间 CS 信号都会拉高。如果使能该功能，在一次传输过程中，CS 信号都不会释放，只有当该次传输全部完成后 CS 信号才会拉高释放。

在 QSPI 模式下，可以通过寄存器 CONFIG 的位 CS\_STRETCH 配置 CS 信号的拉高释放条件。如果不使能该功能，在一次传输过程中，当 FIFO 为空（即 FIFO 中的数据全部发出且没有新的数据填入）时，CS 信号会被拉高释放。如果使能该功能，在一次传输过程中，当 FIFO 为空（即 FIFO 中的数据全部发出且没有新的数据填入）时，CS 信号会维持在低电平，等待新的数据填入。

### 22.3.7 中断

DBI 具有以下几种中断：

- 传输结束中断
- TX FIFO 请求中断
- FIFO 溢出错误中断

通过设置一个像素计数值，传输结束中断会在传输的像素数据达到计数值时产生，Type B 和 Type C 都会产生该中断；

当 DBI\_FIFO\_CONFIG\_1 中 TFICNT 大于 TFITH 时，产生 TX FIFO 请求中断，当条件不满足时该中断标志会自动清除；

FIFO 溢出错误中断会在 TX 发生 Overflow 或者 Underflow 时产生。

### 22.3.8 DMA

DBI TX 支持 DMA 传输模式，使用该模式需要通过寄存器 DBI\_FIFO\_CONFIG\_1 的位 <TFITH> 设置 TX FIFO 的阈值。当该模式启用后，如果 TFICNT 大于 TFITH，则会触发 DMA TX 请求，配置好 DMA 后，DMA 在收到该请求时，会按照设定从内存中将数据搬运到 TX FIFO。

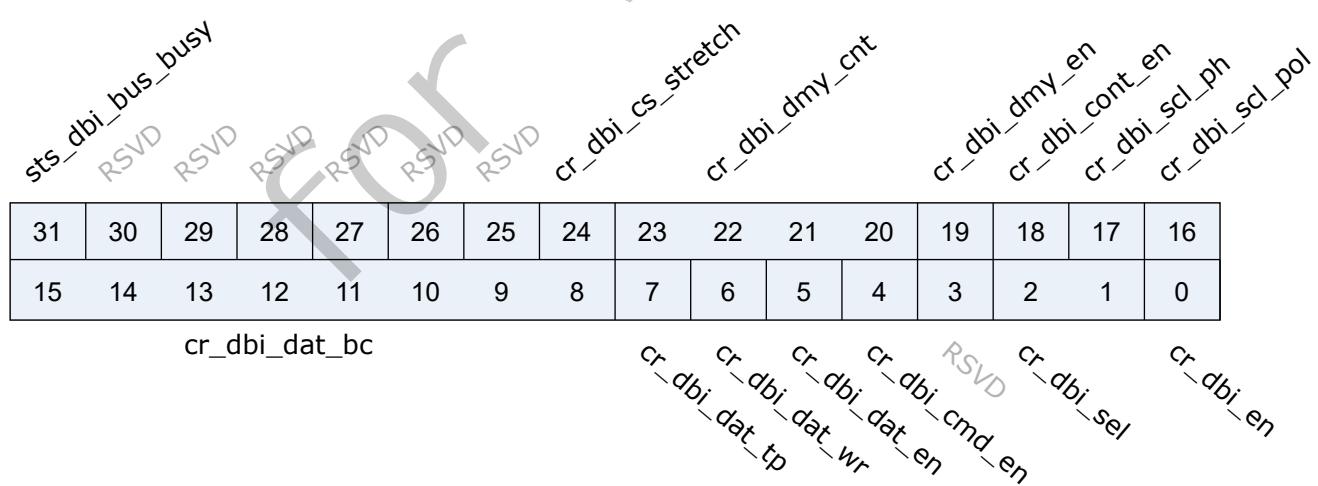
## 22.4 寄存器描述

名称	描述
dbi_config	
qspi_config	
dbi_pix_cnt	
dbi_prd	
dbi_cmd	
dbi_qspi_adr	

名称	描述
dbi_rdata_0	
dbi_rdata_1	
dbi_int_sts	
dbi_yuv_rgb_config_0	
dbi_yuv_rgb_config_1	
dbi_yuv_rgb_config_2	
dbi_yuv_rgb_config_3	
dbi_yuv_rgb_config_4	
dbi_yuv_rgb_config_5	
dbi_fifo_config_0	
dbi_fifo_config_1	
dbi_fifo_wdata	
dbi_dummy	

#### 22.4.1 dbi\_config

地址: 0x3001b000



位	名称	权限	复位值	描述
31	sts_dbi_bus_busy	r	1'b0	Indicator of dbi bus busy

位	名称	权限	复位值	描述
30:25	RSVD			
24	cr_dbi_cs_stretch	r/w	1'b0	<p>Enable signal of CS-low stretch mode</p> <p>1'b0: Disabled, if FIFO is empty during a pixel data transfer, CS will de-assert before FIFO is filled again and new transfer starts</p> <p>1'b1: Enabled, if FIFO is empty during a pixel data transfer, CS will stay asserted while waiting for FIFO to be filled again</p>
23:20	cr_dbi_dmy_cnt	r/w	4'd0	<p>Dummy cycle count, unit: period defined by dbi_prd_d</p> <p>Effective only in Type C (Fixed to 1 dbi_prd_d period in Type B)</p>
19	cr_dbi_dmy_en	r/w	1'b0	<p>Enable signal of dummy cycle(s)</p> <p>1'b0: Disabled, no dummy cycle(s) between command phase and data phase</p> <p>1'b1: Enabled, dummy cycle(s) will be inserted between command phase and data phase</p> <p>Note: Don't-care if QSPI mode is selected (no dummy cycle)</p>
18	cr_dbi_cont_en	r/w	1'b1	<p>Enable signal of pixel data continuous transfer mode</p> <p>1'b0: Disabled, CS_n will de-assert between each pixel</p> <p>1'b1: Enabled, CS_n will stay asserted between each consecutive pixel</p> <p>Note: Don't-care if QSPI mode is selected (always in continuous mode)</p>
17	cr_dbi_scl_ph	r/w	1'b0	SCL clock phase inverse signal
16	cr_dbi_scl_pol	r/w	1'b1	<p>SCL clock polarity</p> <p>0: SCL output LOW at IDLE state</p> <p>1: SCL output HIGH at IDLE state</p>
15:8	cr_dbi_dat_bc	r/w	8'd0	<p>Data byte count of normal data (pixel data count is determined by cr_dbi_pix_cnt)</p> <p>Note: Normal read data can only be up to 8-byte (8'd7). No limit for normal write data (can be up to 256-byte using FIFO)</p>
7	cr_dbi_dat_tp	r/w	1'b0	<p>Data type select</p> <p>1'b0: Normal data (parameter)</p> <p>1'b1: Pixel data</p> <p>Note: Read command supports normal data only</p>
6	cr_dbi_dat_wr	r/w	1'b1	<p>Data phase Read/Write select</p> <p>1'b0: Read data</p> <p>1'b1: Write data</p>

位	名称	权限	复位值	描述
5	cr_dbi_dat_en	r/w	1'b1	Data enable signal 1'b0: Data phase disabled 1'b1: Data phase enabled
4	cr_dbi_cmd_en	r/w	1'b1	Command enable signal 1'b0: No command phase 1'b1: Command will be sent Note: Don't-care if QSPI mode is selected (Command always enabled)
3	RSVD			
2:1	cr_dbi_sel	r/w	2'd0	DBI mode select 2'd0: DBI Type B 2'd1: DBI Type C, 4-wire mode 2'd2: DBI Type C, 3-wire mode 2'd3: QSPI mode
0	cr_dbi_en	r/w	1'b0	Enable signal of DBI function Asserting this bit will trigger the transaction, and should be de-asserted after finish

## 22.4.2 qspi\_config

地址: 0x30001b004

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	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD
RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD
cr_qspi_dat_4b	cr_qspi_adr_4b	cr_qspi_adr_4b	cr_qspi_cmd_4b												

位	名称	权限	复位值	描述
31:6	RSVD			
5:4	cr_qspi_adr_bc	r/w	2'd2	QSPI Address byte count
3	RSVD			

位	名称	权限	复位值	描述
2	cr_qspi_dat_4b	r/w	1'b1	QSPI Data 4-bit (quad) mode 1'b0: Data sent/received in 1-bit mode 1'b1: Data sent/received in 4-bit mode
1	cr_qspi_adr_4b	r/w	1'b1	QSPI Address (display command) 4-bit (quad) mode 1'b0: Address sent in 1-bit mode 1'b1: Address sent in 4-bit mode
0	cr_qspi_cmd_4b	r/w	1'b0	QSPI Command 4-bit (quad) mode 1'b0: Command sent in 1-bit mode 1'b1: Command sent in 4-bit mode

### 22.4.3 dbi\_pix\_cnt

地址: 0x3001b008

cr_dbi_pix_format															
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_dbi_pix_cnt															

位	名称	权限	复位值	描述
31	cr_dbi_pix_format	r/w	1'b0	Pixel format 1'b0: RGB565 1'b1: RGB888/RGB666
30:24	RSVD			
23:0	cr_dbi_pix_cnt	r/w	24'h0	Pixel count Note: Should be a multiple of 2 if RGB565 is selected and a multiple of 4 if RGB888/RGB666 mode is selected

## 22.4.4 dbi\_prd

地址: 0x3001b00c

cr\_dbi\_prd\_d\_ph\_1

cr\_dbi\_prd\_d\_ph\_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\_dbi\_prd\_i

cr\_dbi\_prd\_s

位	名称	权限	复位值	描述
31:24	cr_dbi_prd_d_ph_1	r/w	8'd15	Length of DATA phase 1 (please refer to "Timing" tab)
23:16	cr_dbi_prd_d_ph_0	r/w	8'd15	Length of DATA phase 0 (please refer to "Timing" tab)
15:8	cr_dbi_prd_i	r/w	8'd15	Length of INTERVAL between pixel data (please refer to "Timing" tab)
7:0	cr_dbi_prd_s	r/w	8'd15	Length of START/STOP condition (please refer to "Timing" tab)

## 22.4.5 dbi\_cmd

地址: 0x3001b010

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\_dbi\_cmd

位	名称	权限	复位值	描述
31:8	RSVD			
7:0	cr_dbi_cmd	r/w	8'h2C	DBI Command

## 22.4.6 dbi\_qspi\_adr

地址: 0x3001b014

cr\_qspi\_adr

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\_qspi\_adr

位	名称	权限	复位值	描述
31:0	cr_qspi_adr	r/w	32'h000002C0	QSPI Address (display command) Note: LSB is sent first

## 22.4.7 dbi\_rdata\_0

地址: 0x3001b018

sts\_dbi\_rdata\_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

sts\_dbi\_rdata\_0

位	名称	权限	复位值	描述
31:0	sts_dbi_rdata_0	r	32'h0	Data read from display IC using read command

## 22.4.8 dbi\_rdata\_1

地址: 0x3001b01c

sts\_dbi\_rdata\_1

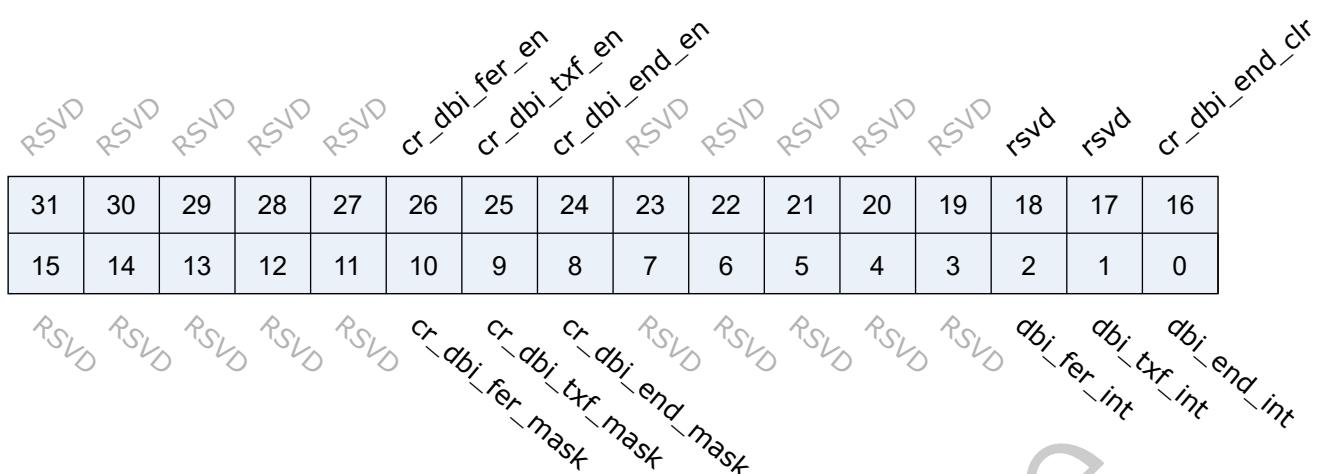
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

sts\_dbi\_rdata\_1

位	名称	权限	复位值	描述
31:0	sts_dbi_rdata_1	r	32'h0	Data read from display IC using read command

## 22.4.9 dbi\_int\_sts

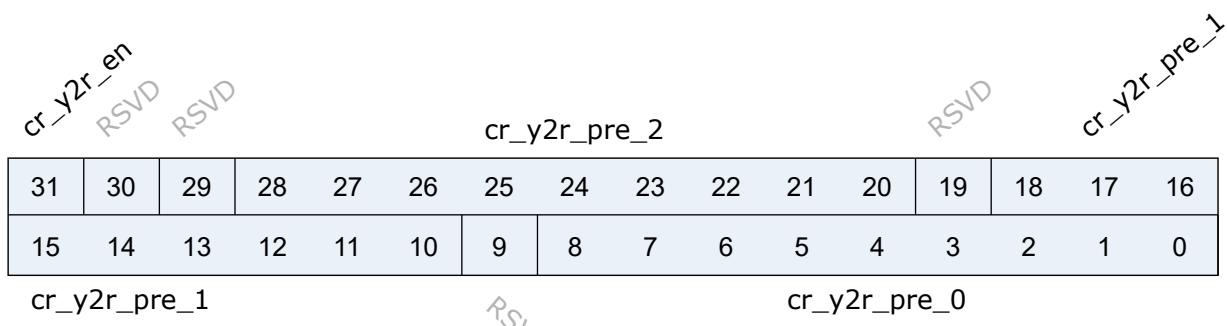
地址: 0x3001b030



位	名称	权限	复位值	描述
31:27	RSVD			
26	cr_dbi_fer_en	r/w	1'b1	Interrupt enable of dbi_fer_int
25	cr_dbi_txf_en	r/w	1'b1	Interrupt enable of dbi_txe_int
24	cr_dbi_end_en	r/w	1'b1	Interrupt enable of dbi_end_int
23:19	RSVD			
18	rsvd	rsvd	1'b0	
17	rsvd	rsvd	1'b0	
16	cr_dbi_end_clr	w1c	1'b0	Interrupt clear of dbi_end_int
15:11	RSVD			
10	cr_dbi_fer_mask	r/w	1'b1	Interrupt mask of dbi_fer_int
9	cr_dbi_txf_mask	r/w	1'b1	Interrupt mask of dbi_txe_int
8	cr_dbi_end_mask	r/w	1'b1	Interrupt mask of dbi_end_int
7:3	RSVD			
2	dbi_fer_int	r	1'b0	TX/RX FIFO error interrupt, auto-cleared when FIFO overflow/underflow error flag is cleared
1	dbi_txf_int	r	1'b1	TX FIFO ready ( $tx\_fifo\_cnt > tx\_fifo\_th$ ) interrupt, auto-cleared when data is pushed
0	dbi_end_int	r	1'b0	Transfer end interrupt, shared by both Type B and C mode

## 22.4.10 dbi\_yuv\_rgb\_config\_0

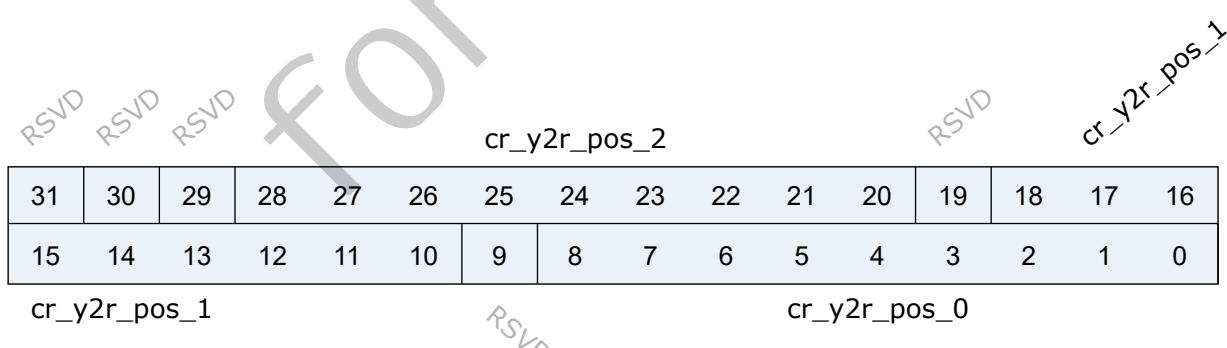
地址: 0x3001b060



位	名称	权限	复位值	描述
31	cr_y2r_en	r/w	1'b1	Display module YUV2RGB enable signal
30:29	RSVD			
28:20	cr_y2r_pre_2	r/w	9'd0	Display module YUV2RGB "pre_offset_2" parameter
19	RSVD			
18:10	cr_y2r_pre_1	r/w	9'd0	Display module YUV2RGB "pre_offset_1" parameter
9	RSVD			
8:0	cr_y2r_pre_0	r/w	9'd0	Display module YUV2RGB "pre_offset_0" parameter

## 22.4.11 dbi\_yuv\_rgb\_config\_1

地址: 0x3001b064



位	名称	权限	复位值	描述
31:29	RSVD			
28:20	cr_y2r_pos_2	r/w	9'd0	Display module YUV2RGB "post_offset_2" parameter

位	名称	权限	复位值	描述
19	RSVD			
18:10	cr_y2r_pos_1	r/w	9'd0	Display module YUV2RGB "post_offset_1" parameter
9	RSVD			
8:0	cr_y2r_pos_0	r/w	9'd0	Display module YUV2RGB "post_offset_0" parameter

#### 22.4.12 dbi\_yuv\_rgb\_config\_2

地址: 0x3001b068

cr_y2r_mtx_02_l								cr_y2r_mtx_01							
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_y2r_mtx_00															

位	名称	权限	复位值	描述
31:24	cr_y2r_mtx_02_l	r/w	8'h0	Display module YUV2RGB "matrix_02" parameter lower bits
23:12	cr_y2r_mtx_01	r/w	12'h0	Display module YUV2RGB "matrix_01" parameter
11:0	cr_y2r_mtx_00	r/w	12'h0	Display module YUV2RGB "matrix_00" parameter

#### 22.4.13 dbi\_yuv\_rgb\_config\_3

地址: 0x3001b06c

cr\_y2r\_mtx\_12\_1

## cr\_y2r\_mtx\_11

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\_y2r\_mtx\_10

cr\_y2r\_mtx\_02\_u

位	名称	权限	复位值	描述
31:28	cr_y2r_mtx_12_l	r/w	4'h0	Display module YUV2RGB "matrix_12" parameter lower bits
27:16	cr_y2r_mtx_11	r/w	12'h0	Display module YUV2RGB "matrix_11" parameter
15:4	cr_y2r_mtx_10	r/w	12'h0	Display module YUV2RGB "matrix_10" parameter
3:0	cr_y2r_mtx_02_u	r/w	4'h0	Display module YUV2RGB "matrix_02" parameter upper bits

## 22.4.14 dbi\_yuv\_rgb\_config\_4

地址: 0x30001b070

## cr\_y2r\_mtx\_21

cr\_y2r\_mtx\_20

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\_y2r\_mtx\_20

## cr\_y2r\_mtx\_12\_u

位	名称	权限	复位值	描述
31:20	cr_y2r_mtx_21	r/w	12'h0	Display module YUV2RGB "matrix_21" parameter
19:8	cr_y2r_mtx_20	r/w	12'h0	Display module YUV2RGB "matrix_20" parameter
7:0	cr_y2r_mtx_12_u	r/w	8'h0	Display module YUV2RGB "matrix_12" parameter upper bits

### 22.4.15 dbi\_yuv\_rgb\_config\_5

地址: 0x3001b074

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\_y2r\_mtx\_22

位	名称	权限	复位值	描述
31:12	RSVD			
11:0	cr_y2r_mtx_22	r/w	12'h0	Display module YUV2RGB "matrix_22" parameter

### 22.4.16 dbi\_fifo\_config\_0

地址: 0x3001b080

RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD
fifo_format	fifo_yuv_mode	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

tx\_fifo\_overflow  
tx\_fifo\_underflow  
tx\_fifo\_clr  
dbi\_dma\_tx\_en

位	名称	权限	复位值	描述
31:29	fifo_format	r/w	3'd0	FIFO pixel data format (see Tab 'FIFO Format' for details) 3'd0: 8'h0, B[7:0], G[7:0], R[7:0] 3'd1: 8'h0, R[7:0], G[7:0], B[7:0] 3'd2: B[7:0], G[7:0], R[7:0], 8'h0 3'd3: R[7:0], G[7:0], B[7:0], 8'h0 3'd4: R_n[7:0], B[7:0], G[7:0], R[7:0] 3'd5: B_n[7:0], R[7:0], G[7:0], B[7:0] 3'd6: 2B[7:3], G[7:2], R[7:3] 3'd7: 2R[7:3], G[7:2], B[7:3] Note: FIFO data format does not affect normal data, which is fixed to LSB sent first Byte3[7:0], Byte2[7:0], Byte1[7:0], Byte0[7:0]
28	fifo_yuv_mode	r/w	1'b0	FIFO data YUV mode R <-> Y G <-> U/Cb B <-> V/Cr
27:6	RSVD			
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	RSVD			
2	tx_fifo_clr	w1c	1'b0	Clear signal of TX FIFO
1	RSVD			
0	dbi_dma_tx_en	r/w	1'b0	Enable signal of dma_tx_req/ack interface

#### 22.4.17 dbi\_fifo\_config\_1

地址: 0x3001b084

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	tx_fifo_cnt														
------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	-------------

位	名称	权限	复位值	描述
31:19	RSVD			
18:16	tx_fifo_th	r/w	3'd0	TX FIFO threshold, dma_tx_req will not be asserted if tx_fifo_cnt is less than this value
15:4	RSVD			
3:0	tx_fifo_cnt	r	4'd8	TX FIFO available count

#### 22.4.18 dbi\_fifo\_wdata

地址: 0x3001b088

dbi\_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

dbi\_fifo\_wdata

位	名称	权限	复位值	描述
31:0	dbi_fifo_wdata	w	x	Pixel data with 8 types of format (determined by fifo_format)

#### 22.4.19 dbi\_dummy

地址: 0x3001b0fc

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

dbi\_dummy

位	名称	权限	复位值	描述
31:8	RSVD			
7:0	dbi_dummy	r/w	8'h0	Dummy register for FPGA mode pad control

## 23.1 简介

SD 主机控制器（SD Host Controller，即 SDH）用于控制与 SDIO 或 SD 卡的通信，所有访问都通过标准的控制寄存器进行设置。

## 23.2 主要特征

- 符合 SD 协会定义的 SD 主机控制器规范
- 适用于 SDIO/SD 存储卡
- 支持 SD 主机控制器标准寄存器设置
- 支持针对高速数据传输的 DMA 操作
- 支持中断
- 支持数据块传输

## 23.3 功能描述

### 23.3.1 SDH 总体结构

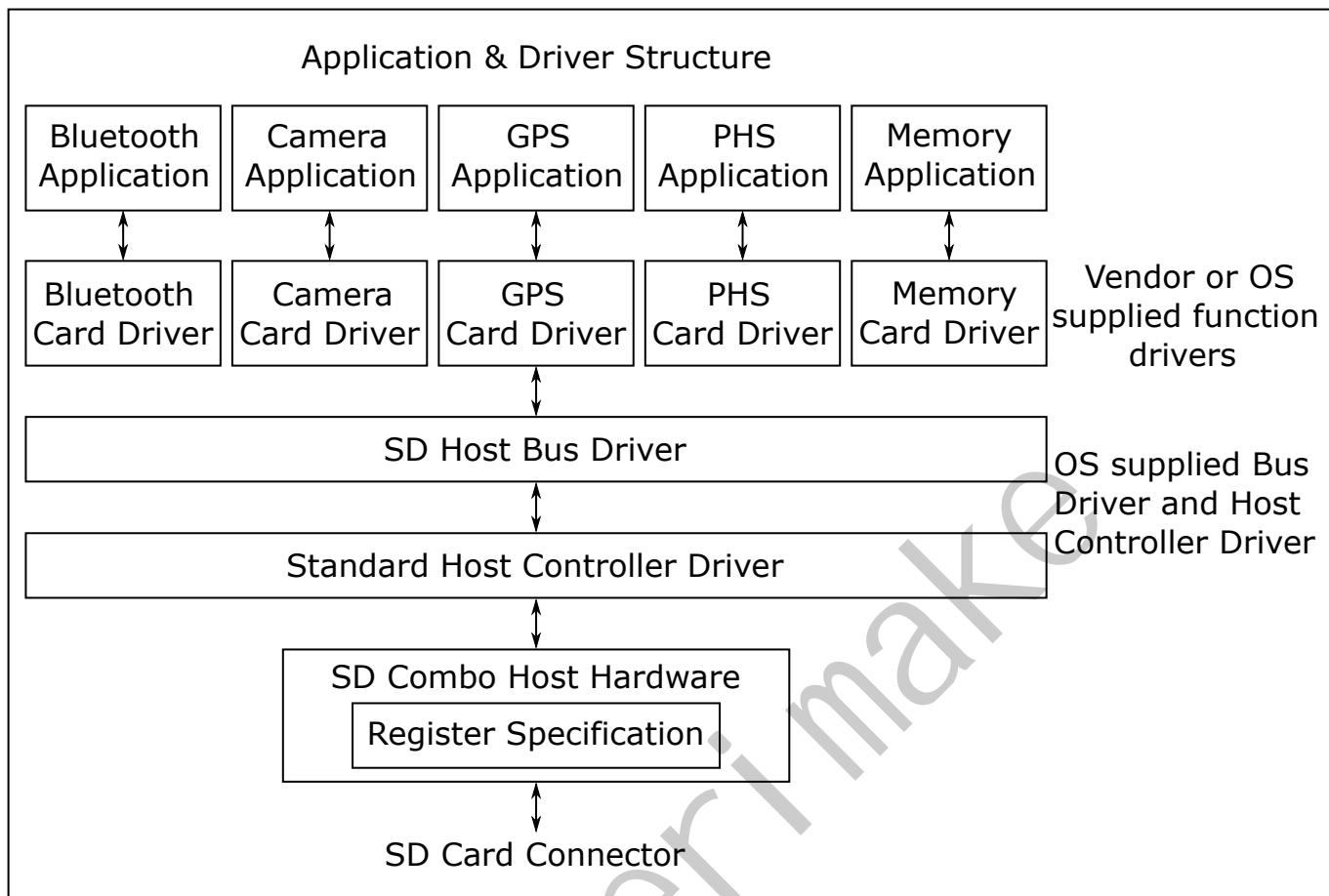


图 23.1: SDH 硬件和驱动结构图

## 23.3.2 寄存器映射

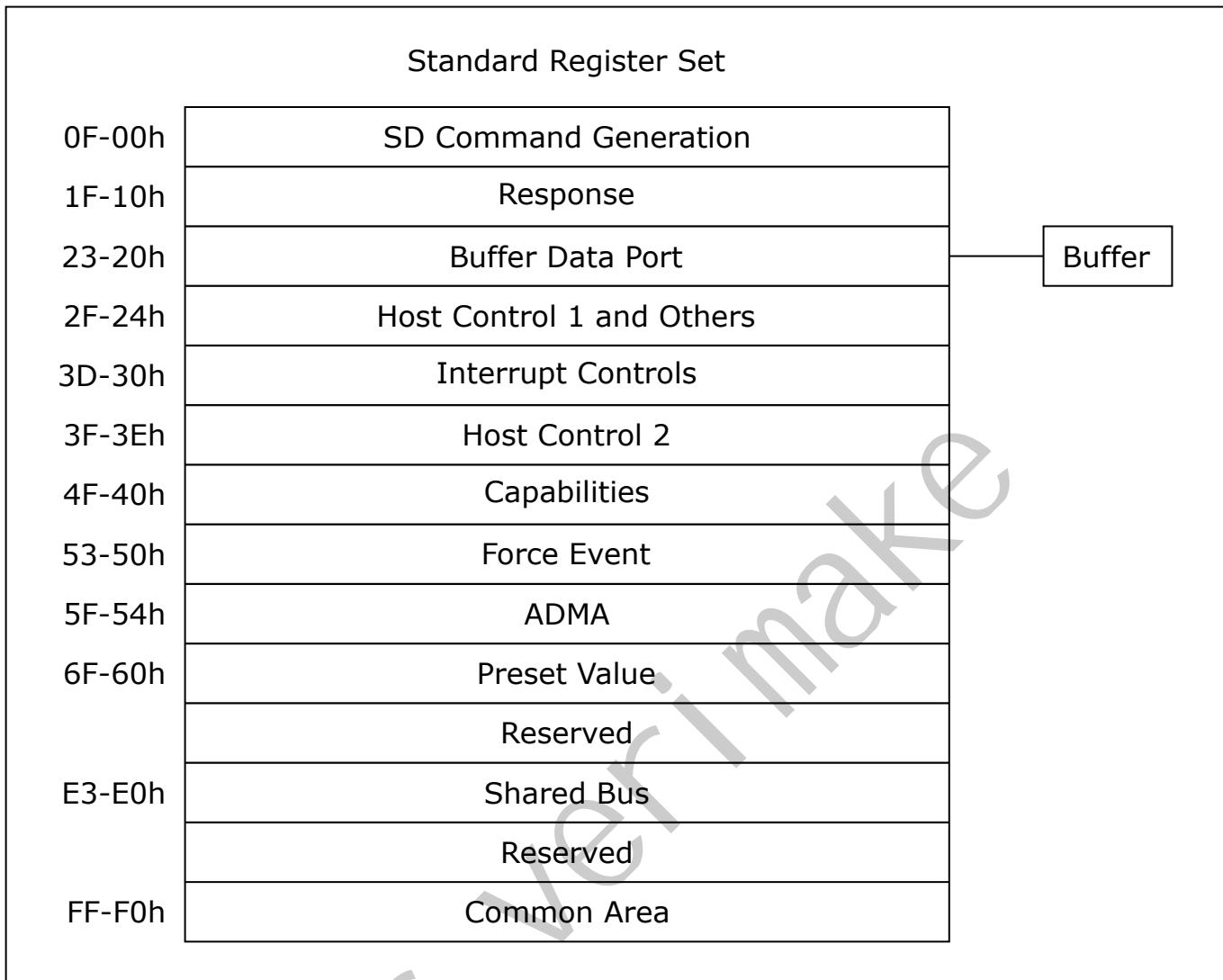


图 23.2: 标准寄存器映射分类图

其中:

- SD Command Generation: 用于生成 SD 命令的参数;
- Response: 来自卡的响应值;
- Buffer Data Port: 内部缓冲区的数据访问端口;
- Host Control 1 and Others: 当前状态、SD 总线控制、主机复位等;
- Interrupt Controls: 中断状态和启用;
- Host Control 2: 供应商特定的主机控制器支持信息;
- Capabilities: 主机控制寄存器的扩展;

- **Force Event:** 通过软件生成事件的测试寄存器;
- **ADMA:** 高级 DMA 寄存器;
- **Preset Value:** 时钟频率选择和驱动强度选择预设;
- **Shared Bus:** 共享总线系统的设备控制;
- **Common Area:** 公共信息区。

### 23.3.3 多卡槽支持

为每个卡槽定义一个标准寄存器集，如果主机控制器有两个卡槽，则需要两个寄存器集。每个卡槽都是独立控制的。这使得支持总线接口电压、总线定时和 SD 时钟频率的组合成为可能。

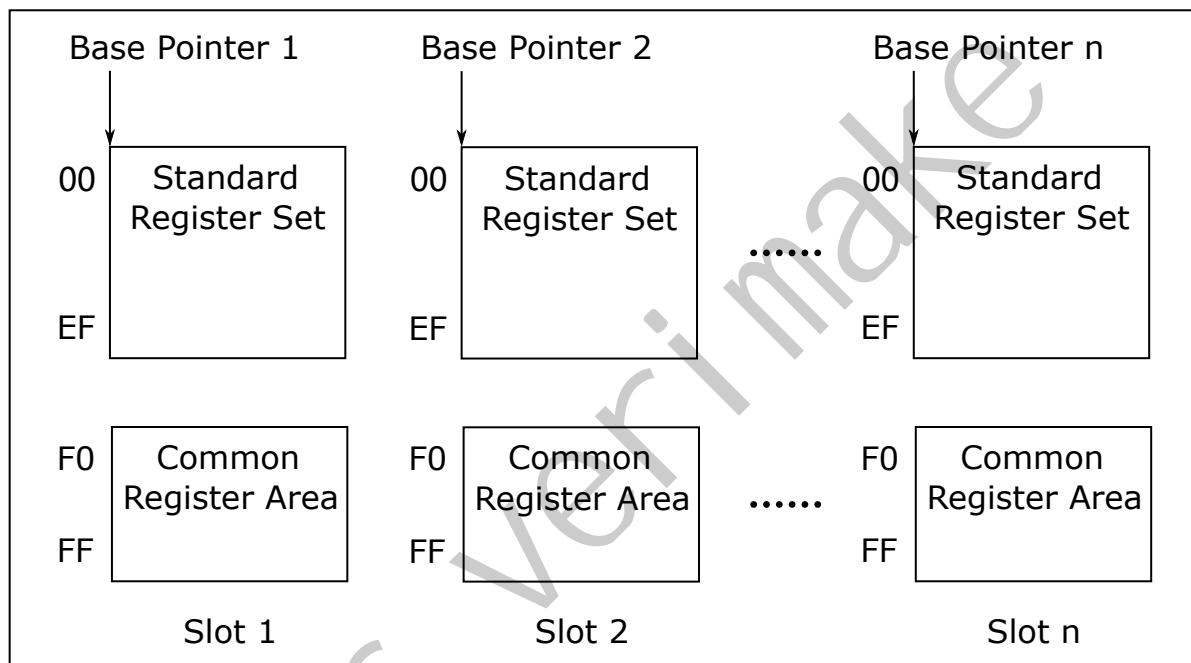


图 23.3: 多卡槽控制器的寄存器映射

上图显示了多卡槽主机控制器的寄存器映射。主机驱动程序应使用 PCI 配置寄存器或供应商特定方法确定卡槽数量和指向每个卡槽标准寄存器集的基指针。从 0F0h 到 OFFh 的偏移量保留给公共寄存器区域，该区域定义卡槽控制和公共状态的信息。公共寄存器区域可从任何卡槽的寄存器集访问。这允许软件独立控制每个卡槽，因为它可以从每个寄存器集中访问卡槽中断状态寄存器和主机控制器版本寄存器。

### 23.3.4 支持 DMA

主机控制器为主机驱动程序提供“编程 I/O”方法，以便使用缓冲区数据端口寄存器传输数据。可选地，主机控制器实现者可以支持使用 DMA 的数据传输。在使用 DMA 之前，主机驱动程序应确认主机控制器和系统总线均支持 DMA (PCI 总线可支持 DMA)。DMA 应支持单块和多块传输。在 DMA 传输执行期间，主机控制器寄存器应保持可访问性，以发出非 DAT 行命令。无论采用何种系统总线数据传输方法，DMA 传输的结果应相同。主机驱动程序可以通过块间隙控制寄存器中的控制位停止和重新启动 DMA 操作。通过设置在块间隙请求时停止，DMA 操作可以在块间隙处停止。通过设置继续请求，可以重新启动 DMA 操作。如果发生错误，DMA 操作应停止。要中止 DMA 传输，主机驱动程序应通过软件重置寄存器中 DAT 行的软件重置来重置主机控制器，并在执行多块读/写命令时发出 CMD12。

### 23.3.5 SD 命令生成

	SDMA Command	ADMA Command	CPU Data Transfer	Non-DAT Transfer
SDMA System Address /Argument 2	Yes /No	No /Auto CMD23	No /Auto CMD23	No /No
Block Size	Yes	Yes	Yes	No (Protected)
Block Count	Yes	Yes	Yes	No (Protected)
Argument 1	Yes	Yes	Yes	Yes
Transfer Mode	Yes	Yes	Yes	No (Protected)
Command	Yes	Yes	Yes	Yes

图 23.4: 用于生成 SD 命令的寄存器

上表显示了三种类型的事务所需的寄存器设置（寄存器集中从 000h 到 00Fh 的偏移量）：DMA 生成的传输（SDMA 或 ADMA）、CPU 数据传输（使用“编程 I/O”）和非 DAT 传输。启动事务时，主机驱动程序应按从 000h 到 00Fh 的顺序对这些寄存器进行编程。起始寄存器偏移量可根据事务类型计算。最后写入的偏移量应始终为 00Fh，因为写入命令寄存器的高位字节将触发 SD 命令的发出。在一个数据事务期间，主机驱动程序不应读取 SDMA 系统地址、块大小和块计数寄存器，除非由于值正在更改且不稳定而停止或挂起传输。为了防止在发出命令时使用数据传输破坏寄存器，块大小、块计数和传输模式寄存器应由主机控制器进行写保护，同时在当前状态寄存器中将命令禁止（DAT）设置为 1（此信号无法保护 SDMA 系统地址）。当命令禁止（CMD）设置为 1 时，主机驱动程序不得写入参数 1 和命令寄存器。

### 23.3.6 挂起和恢复机制

通过检查功能寄存器中的挂起/恢复支持，可以确定对挂起/恢复的支持。当 SD 卡接受挂起请求时，主机驱动程序在发出其他 SD 命令之前将信息保存在前 14 字节寄存器（即偏移量 000h-00Dh）中。恢复时，主机驱动程序将恢复这些寄存器，然后发出 **Resume** 命令以继续挂起的操作。SDIO 卡在响应恢复命令时设置 DF（恢复数据标志）。（由于挂起和恢复命令是 CMD52 操作，因此响应数据实际上是 CCCR 中的功能选择寄存器。）如果 DF 设置为 0，则表示 SDIO 卡在挂起时无法继续数据传输。该位可用于控制数据传输和中断周期。如果 **Resume Data**（恢复数据）标志设置为 0，则如果正在恢复的事务处于 4 位模式，则不再传输更多数据，并启动中断周期。如果 DF 设置为 1，则数据传输将继续。需要注意的是，要使用挂起和恢复功能，SDIO 卡必须支持挂起和恢复命令以及读取等待控制。

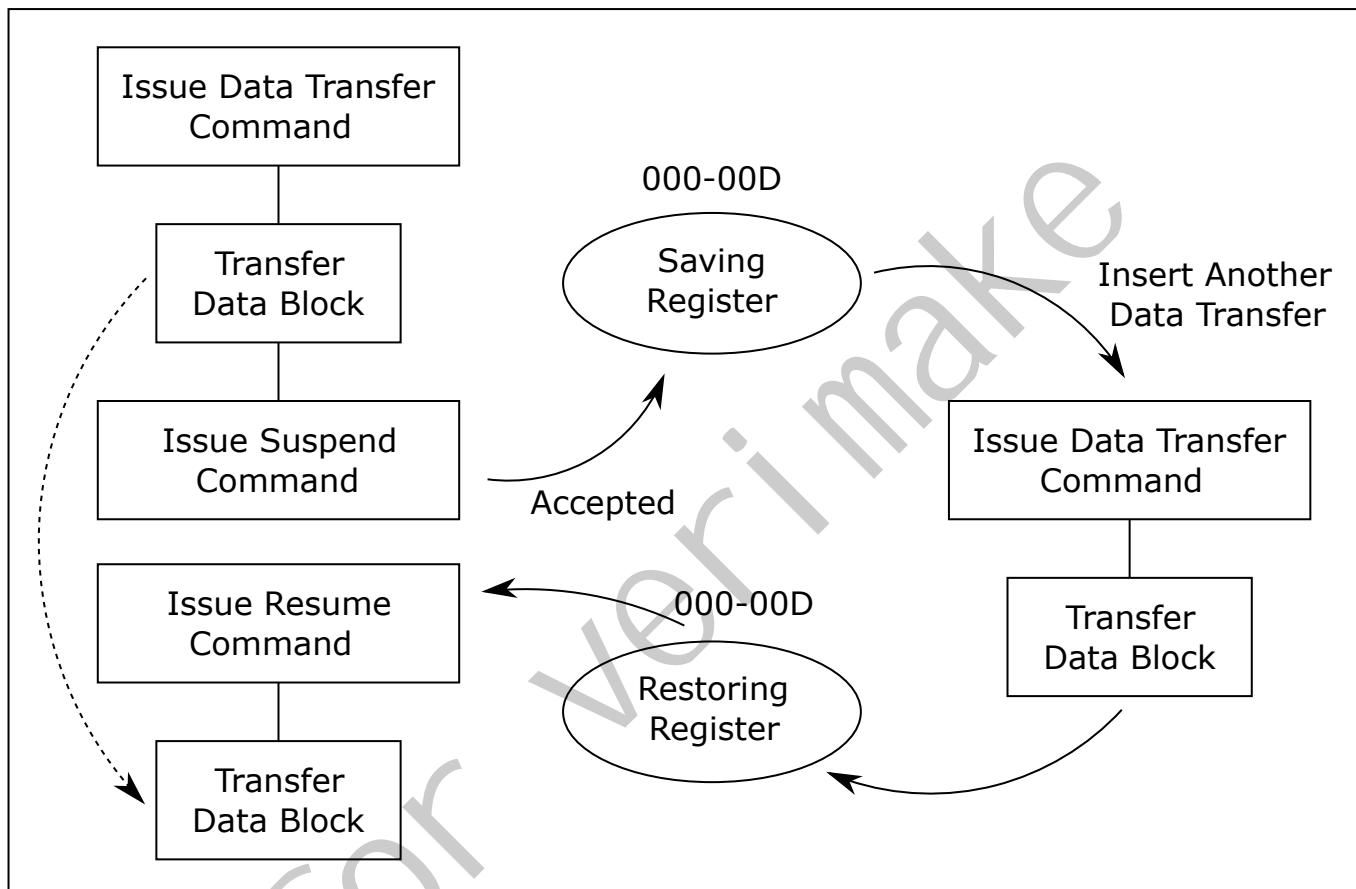


图 23.5: 挂起和恢复机制

### 23.3.7 缓冲区控制

主机控制器具有用于数据传输的数据缓冲区。主机驱动程序通过 32 位缓冲区数据端口寄存器访问内部缓冲区。下面是一些访问缓冲区的规则。

### 23.3.7.1 缓冲区指针的控制

在内部，主机控制器维护一个指针来控制数据缓冲区。主机驱动程序无法直接访问指针，每次访问缓冲区数据端口寄存器时，指针都会根据写入缓冲区的数据量递增。为了适应各种系统总线，无论系统总线宽度如何，都应实现该指针（可支持 8 位、16 位、32 位或 64 位系统总线宽度）。

### 23.3.7.2 确定缓冲块长度

为了能够在突发时传输数据块，主机控制器和 SD 卡缓冲区大小之间的关系非常重要。主机驱动器应为主机控制器和卡使用相同的数据块长度。如果控制器和卡缓冲区大小不同，主机驱动程序应使用较小的值。最大主机控制器缓冲区大小由功能寄存器中的最大块长度字段定义。

### 23.3.7.3 分割大数据传输

SDIO 命令 CMD53 定义根据以下公式限制数据传输的最大数据大小：最大数据大小 = 块大小  $\times$  块计数例如，块大小由缓冲区大小指定，块计数的最大值为 512（9 位计数），如 CMD53 的命令参数中所指定。在最坏的情况下，如果卡只有一个 1 字节的缓冲区，则最多可以使用 CMD53 传输 512 字节的数据（块大小 =1，块计数 =512）。如果卡不支持多块模式，在这种情况下只能传输一个字节。如果应用程序或卡驱动程序希望传输更大尺寸的数据，主机驱动程序应将大数据划分为多个 CMD53 块。

### 23.3.8 中断控制寄存器之间的关系

主机控制器实现许多中断源。中断源可以作为中断或系统唤醒信号启用。如果正常中断状态启用或错误中断状态启用寄存器中的中断源对应位为 1，且中断变为激活状态，则其激活状态被锁定，并可供正常中断状态寄存器或错误中断状态寄存器中的主机驱动程序使用。当中断状态启用被清除时，中断状态应被清除。如果在正常中断信号启用寄存器或错误中断信号启用寄存器中也设置了相应位，则在中断状态寄存器中设置位的中断源应断言系统中断信号。一旦发出信号，大多数中断将通过向中断状态寄存器中的相关位写入 1 来清除。但是，卡中断应由卡驱动程序清除。如果生成卡中断，主机驱动程序可以清除卡中断状态启用，以在卡驱动程序处理卡中断时禁用卡中断。清除所有中断源后，主机驱动程序再次将其设置为启用另一个卡中断。禁用卡中断状态启用可避免在处理中断服务期间产生多个中断。唤醒控制寄存器允许对卡中断、卡插入或卡移除状态更改进行配置，以生成系统唤醒信号。这些中断的启用或屏蔽独立于正常中断信号启用寄存器。可从正常中断状态寄存器读取唤醒事件的类型。中断信号和唤醒信号为逻辑或，应从插槽中断状态寄存器中读取。

### 23.3.9 硬件框图和定时部分

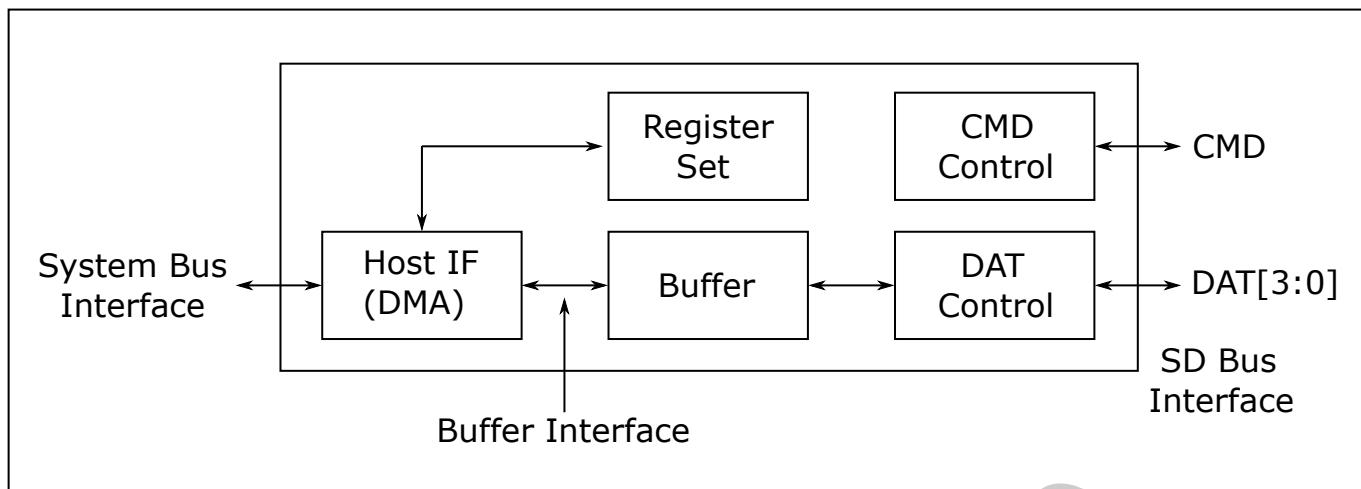


图 23.6: 主机控制器框图

主机控制器有两个总线接口，系统总线接口和 SD 总线接口。主机控制器假定这些接口是异步的（即，在不同的时钟频率下工作）。主机驱动程序处于系统总线时间（因为它是由主机控制器 CPU 在其系统时钟上执行的软件）。SD 卡在 SD 总线时间上（也就是说，其操作由 SDCLK 同步）。主机控制器应同步信号，以便在这些接口之间进行通信。数据块应在缓冲模块上同步。所有状态寄存器应通过系统时钟同步，并在输出至系统接口期间保持同步。触发 SD 总线事务的控制寄存器应由 SDCLK 同步。因此，在两个接口之间传播信号时将存在定时延迟。这意味着主机驱动程序无法实时控制 SD 总线，需要依靠主机控制器根据寄存器设置控制 SD 总线。缓冲区接口启用内部读写缓冲区。传输完成中断状态指示 DMA 和非 DMA 传输的读/写传输完成。但是，读和写之间的计时是不同的。读取传输应在所有有效数据传输到主机系统并准备好供主机驱动程序访问后完成。写入传输应在所有有效数据传输到 SD 卡且忙碌状态结束后完成。

### 23.3.10 自动 CMD12

SD 内存的多个块传输需要 CMD12 停止事务。最后一次数据块传输完成时，主机控制器自动发出 CMD12。主机控制器的此功能称为自动 CMD12。发出多块传输命令时，主机驱动程序应在传输模式寄存器中设置 Auto CMD12 Enable（自动 CMD12 启用）。应通过主机控制器中的硬件完成与最后一个数据块的自动 CMD12 定时同步。在多次块传输期间，可以发出不使用 DAT 行的命令。这些命令使用 CMD\_wo\_DAT 符号表示。为了防止 DAT 线路命令和 CMD\_wo\_DAT 命令发生冲突，主机控制器应仲裁在 SD 总线上发出每个命令的时间。因此，在主机驱动程序写入命令寄存器后，可能不会立即发出命令。该命令可以在自动 CMD12 之前或之后发出，具体取决于时间。为了能够区分 DAT line 和 CMD\_wo\_DAT 命令的响应，可以从响应寄存器的前四个字节（在标准寄存器集中的偏移量 01Ch 处）确定自动 CMD12 响应。如果检测到与自动 CMD12 相关的错误，主机控制器应发出自动 CMD 错误中断。主机驱动程序可以通过读取自动 CMD 错误状态寄存器来检查自动 CMD12 错误状态（命令索引/结束位/CRC/超时错误）。如果未执行自动 CMD12，则主机驱动程序需要从 CMD\_wo\_DAT 错误中恢复，并发出 CMD12 以停止多块传输。如果未执行 CMD\_wo\_DAT，则主机驱动程序可以在从自动 CMD12 错误恢复后再次发出该命令。在 UHS 模式 SDR104 中，主机驱动程序应使用自动 CMD23 停止多块读/写操作，而不是使用自动 CMD12。在其他总线速度模式下，如果卡支持 CMD23，则主机驱动程序应使用自动 CMD23，而不是使用 CMD12。

### 23.3.11 控制 SDCLK

下表显示了 SDCLK 如何由电源控制寄存器中的 SD 总线电源和时钟控制寄存器中的 SD 时钟启用控制：

SD Bus Power	SD Clock Enable	State of SDCLK
Change 0 to 1	0	Drive Low
	1	Start Clock With Specified Period Of High
Change 1 to 0	0	Drive Low
	1	Drive Low Immediately
0	Don't Care	Drive Low
1	Change 0 to 1	Start Clock With Specified Period Of High
	Change 1 to 0	Maintains Period Of High And Then Stops Clock And Drive Low

图 23.7: 通过 SD 总线电源和 SD 时钟启用控制 SDCLK

SDCLK 的时钟周期由时钟控制寄存器中的 SDCLK 频率选择和功能寄存器中的 SD 时钟的基本时钟频率指定。由于 SD 卡可以同时使用两个时钟边缘，因此 SD 时钟的占空比应为平均 50%（分散在 45-55% 范围内），高电平周期应为时钟周期的一半。SDCLK 的振荡从驱动规定的高电压周期开始。当 SDCLK 被 SD 时钟启用停止时，主机控制器应在高驱动周期后停止 SDCLK，以维持时钟占空比。当 SDCLK 被 SD 总线电源停止时，主机控制器应立即停止 SDCLK（驱动器低），并且应清除 SD 时钟启用。

### 23.3.12 高级 DMA

SD 主机控制器标准规范 2.00 版定义了 ADMA (高级 DMA) 传输算法。SD 主机控制器标准规范 1.00 版中定义的 DMA 算法称为 SDMA (单操作 DMA)。SDMA 的缺点是，在每个页面边界产生的 DMA 中断会干扰 CPU 重新编程新的系统地址。这种 SDMA 算法通过在每个页面边界处中断而形成性能瓶颈。ADMA 采用分散-聚集 DMA 算法，因此具有更高的数据传输速度。在执行 ADMA 之前，主机驱动程序可以将系统内存和 SD 卡之间的数据传输列表编程到描述符表中。它使 ADMA 能够在不中断主机驱动程序的情况下运行。此外，ADMA 不仅可以支持 32 位系统内存寻址，还可以支持 64 位系统内存寻址。32 位系统内存寻址使用 64 位地址寄存器的低 32 位字段。ADMA 有两种类型：ADMA1 和 ADMA2。ADMA1 仅支持系统内存中 4KB 对齐数据的数据传输。ADMA2 改进了该限制，使任何位置和大小的数据都可以在系统内存中传输。描述符表的格式在它们之间是不同的。本文档中使用术语“ADMA”时，指 ADMA2。

## 23.3.12.1 ADMA2 的框图

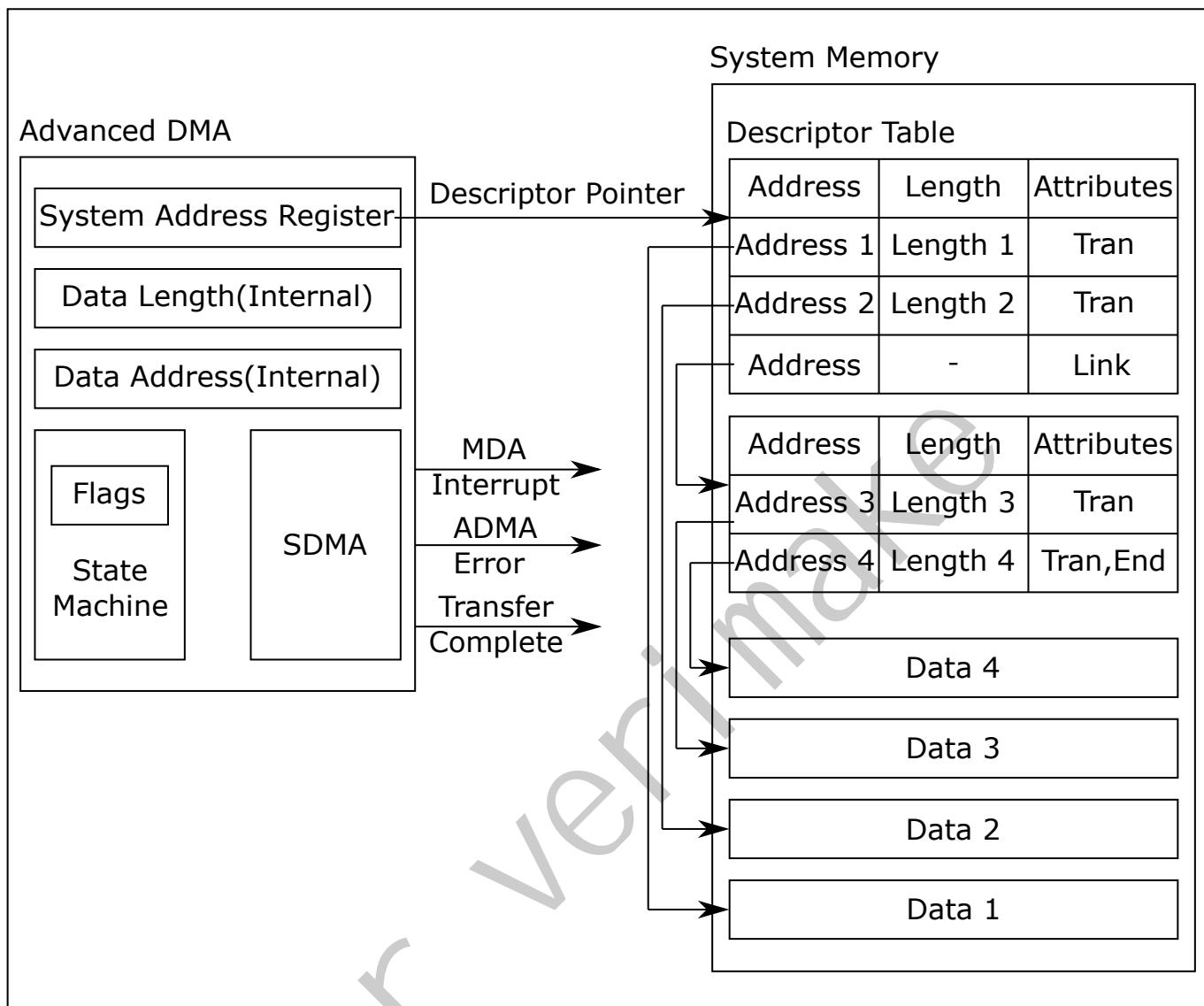


图 23.8: ADMA2 的框图

描述符表由主机驱动程序在系统内存中创建。32 位地址描述符表用于 32 位寻址的系统，64 位地址描述符表用于 64 位寻址的系统。每个描述符行（一个可执行单元）由地址、长度和属性字段组成。该属性指定描述符行的操作。ADMA2 包括 SDMA、状态机和寄存器电路。ADMA2 不使用 32 位 SDMA 系统地址寄存器（偏移量 0），但使用 64 位高级 DMA 系统地址寄存器（偏移量 058h）作为描述符指针。写入命令寄存器会触发关闭 ADMA2 传输。ADMA2 获取一个描述符行并执行它。重复此过程，直到找到描述符的末尾（属性中的 end=1）。

### 23.3.12.2 数据地址和数据长度要求

对描述符进行编程有 3 个要求:

- 地址的最小单位为 4 字节;
- 每个描述符行的最大数据长度小于 64KB;
- 总长度 = 长度 1 + 长度 2 + 长度 3 + …+ 长度 n = 块大小的倍数。

若描述符的总长度不是块大小的倍数，则可能不会终止 ADMA2 传输。在这种情况下，应通过数据超时中止传输。块计数寄存器限制最大 65535 块传输。如果 ADMA2 操作小于或等于 65535 块传输，则可以使用块计数寄存器。在这种情况下，描述符表的总长度应等于乘以块大小和块计数。如果 ADMA2 操作超过 65535 块传输，则应通过在传输模式寄存器中将 0 设置为块计数启用来禁用块计数寄存器。在这种情况下，数据传输的长度不是由块计数指定的，而是由描述符表指定的。因此，检测 SD 总线上最后一个块的定时可能不同，并且它影响当前状态寄存器中的读传输活动、写传输活动和 DAT 线活动的控制。在读取操作的情况下，可能会读取多个块。如果读取操作是针对最后一块内存区域，则主机驱动程序应忽略超出范围的错误。

## 23.3.12.3 描述符表

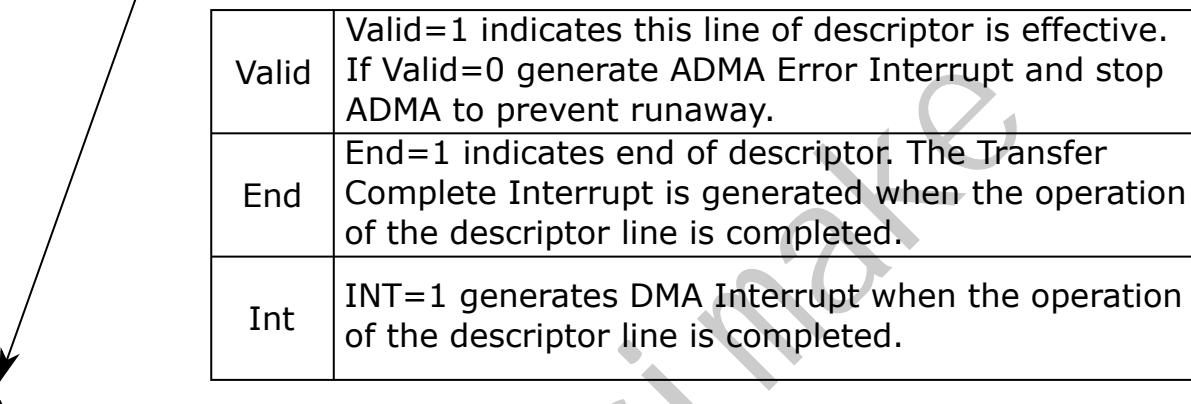
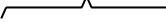
Address Field		Length		Reserved		Attribute							
63	32	31	16	15	06	05	04	03	02	01	00		
32-bit Address		16-bit Length		000000		Act2	Act1	0	Int	End	Valid		
													
		Valid		Valid=1 indicates this line of descriptor is effective. If Valid=0 generate ADMA Error Interrupt and stop ADMA to prevent runaway.									
		End		End=1 indicates end of descriptor. The Transfer Complete Interrupt is generated when the operation of the descriptor line is completed.									
		Int		INT=1 generates DMA Interrupt when the operation of the descriptor line is completed.									
													
Act2	Act1	Symbol	Comment		Operation								
0	0	Nop	No Operation		Do not execute current line and go to next line.								
0	1	Rsv	Reserved		Do not execute current line and go to next line.								
1	0	Tran	Transfer Data		Transfer data of one descriptor line.								
1	1	Link	Link Descriptor		Link to another descriptor.								

图 23.9: 32 位地址描述符表

上图显示了 32 位地址描述符表的定义。一个描述符行消耗 64 位（8 字节）内存空间。属性用于控制描述符。指定了 3 个动作符号：“Nop”操作跳过当前描述符行并获取下一行；“Tran”操作传输由地址和长度字段指定的数据；“Link”操作用于连接分开的两个描述符。链接的地址字段指向下一个描述符表。Act2=0 和 Act1=1 的组合被保留并定义为与 Nop 相同的操作。控制器的未来版本可能会使用此字段并重新定义新操作。32 位地址存储在 64 位地址寄存器的低 32 位中。对于 32 位地址描述符表，地址字段应设置在 32 位边界上（低 2 位始终设置为 0）。

### 23.3.12.4 ADMA2 状态

下图是 ADMA2 的状态图, 定义了 4 个状态, 获取描述符状态、更改地址状态、传输数据状态和停止 ADMA 状态:

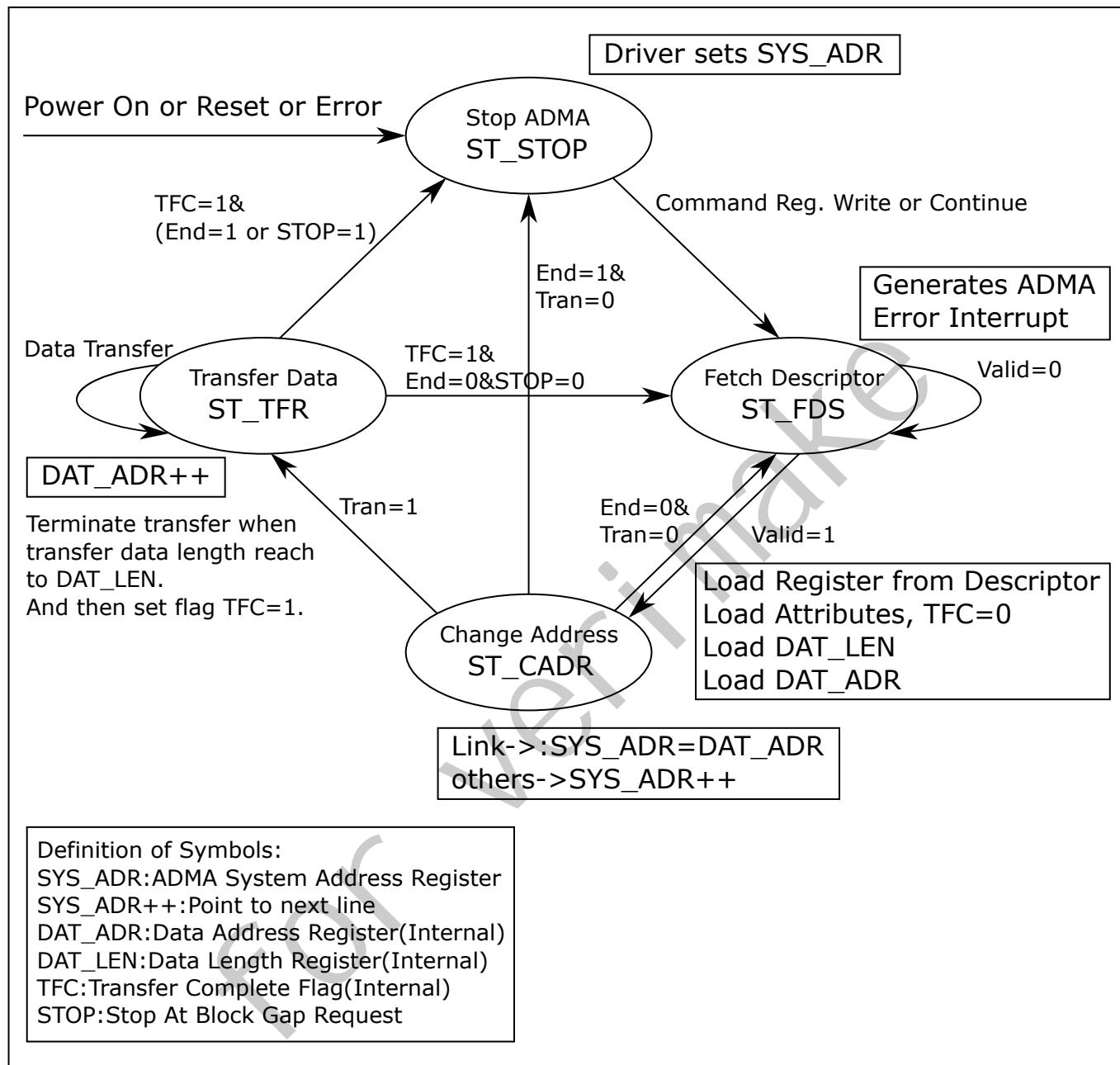


图 23.10: ADMA2 的状态图

其中:

- **ST\_FDS (获取描述符):** ADMA2 获取描述符行并在内部寄存器中设置参数, 接下来转到 **ST\_CADR** 状态;
- **ST\_CADR (更改地址):** 链接操作将另一个描述符地址加载到 ADMA 系统地址寄存器。在其他操作中, ADMA 系统地址寄存器递增到下一个描述符行。如果 End=0, 则转到 **ST\_TFR** 状态。即使出现一些错误, ADMA2 也不得在此状态下停止;

- ST\_TFR (传输数据): 在系统内存和 SD 卡之间执行一条描述符行的数据传输。如果数据传输继续 (End=0), 则转到 ST\_FDS 状态。如果数据传输完成, 则转到 ST\_STOP 状态;
- ST\_STOP (停止 DMA): 在以下情况下, ADMA2 将保持此状态, 一是上电复位或软件复位后, 一是所有描述符数据传输都已完成。如果通过写入命令寄存器启动新的 ADMA2 操作, 转至 ST\_FDS 状态。

ADMA2 不支持挂起/恢复功能, 但可以使用停止和继续。当在 ADMA2 操作期间设置块间隙控制寄存器中的块间隙停止请求时, 当 ADMA2 在块间隙停止时, 将生成块间隙事件中断。主机控制器应使用读取等待或停止 SD 时钟停止 ADMA2 读取操作。停止 ADMA2 时, 无法发出任何 SD 命令。ADMA2 传输期间发生错误可能会停止 ADMA2 操作并生成 ADMA 错误中断。ADMA 错误状态寄存器中的 ADMA 错误状态字段保存 ADMA2 已停止的状态。主机驱动程序可以通过以下方法识别错误描述符位置: 如果 ADMA 在 ST\_FDS 状态下停止, 则 ADMA 系统地址寄存器将指向错误描述符行。如果 ADMA 在 ST\_TFR 或 ST\_STOP 状态下停止, ADMA 系统地址寄存器将指向错误描述符行的下一个位置。因此, ADMA2 不得在 ST\_CADR 状态停止。

### 23.3.13 测试寄存器

测试寄存器是为测试目的而定义的。当很难有意生成某些中断时, 可以使用此功能手动生成这些中断以进行驱动程序调试。为此, 定义了用于控制错误中断状态和自动 CMD 错误状态的强制事件寄存器。有意控制插卡和拔卡也很困难, 主机控制 1 寄存器中的卡检测信号选择和卡检测测试级别使能够人为控制当前状态寄存器所插入的卡, 并在正常中断状态寄存器中生成卡插入和卡移除中断。

### 23.3.14 块计数

块计数命令 (CMD23) 提供了停止多块操作的无计时方法。在 CMD23 的参数中设置块计数, 以指定其后的 CMD18 或 CMD25 的传输长度。自动 CMD23 是在发送 CMD18 或 CMD25 之前自动发出 CMD23 的功能。此功能的目的是通过删除 CMD23 的中断服务, 避免在内存访问期间性能下降。偏移量 008h 参数 1 寄存器用于 CMD18 或 CMD25。然后为 CMD23 的参数 2 寄存器分配偏移量 000h。主机控制器不使用参数 2 寄存器计算数据传输长度。主机端数据传输操作的数据长度有两种情况: 非 ADMA 和 ADMA。AMDA 描述符的总长度是指 ADMA 描述符每行的 16 位数据长度之和。应为 ADMA 禁用块计数启用。需要注意的是, 主机控制器的总数据传输长度应等于卡的传输长度。

### 23.3.15 采样时钟调谐

在 UHS-I 模式下, SD 总线可以在高时钟频率模式下运行, 然后 CMD 和 DAT[3:0] 线路上的卡的数据窗口变小。数据窗口的位置将根据卡和主机系统的实施情况而变化。因此, 当通过执行调谐程序并调整采样时钟来支持 SDR104 或 SDR50 (如果在能力寄存器中将 SDR50 的使用调谐设置为 1) 时, 主机控制器应支持调谐电路。主机控制 2 寄存器中的执行调谐和采样时钟选择用于控制调谐电路。

### 23.3.16 SD 主机标准寄存器

#### 23.3.16.1 SD 主机控制寄存器映射

下表总结了标准 SD 主机控制器寄存器集。主机驱动程序需要确定由主机系统特定方法设置的寄存器的基址。寄存器集的大小为 256 字节。对于多个卡槽控制器，每个卡槽分配一个寄存器集，但偏移量 0F0h-0FFh 处的寄存器分配为公共区域，这些寄存器包含每个卡槽寄存器集的相同值。

Offset	15-08 bit	07-00 bit	Offset	15-08 bit	07-00 bit
002h	SDMA System Address(High),Argument 2(High)		000h	SDMA System Address(Low),Argument 2(Low)	
006h	Block Count		004h	Block Size	
00Ah	Argument 1(High)		008h	Argument 1(Low)	
00Eh	Command		00Ch	Transfer Mode	
012h	Response1		010h	Response0	
016h	Response3		014h	Response2	
01Ah	Response5		018h	Response4	
01Eh	Response7		01Ch	Response6	
022h	Buffer Data Port1		020h	Buffer Data Port0	
026h	Present State		024h	Present State	
02Ah	Wakeup Control	Block Gap Control	028h	Power Control	Host Control 1
02Eh	Software Reset	Timeout Control	02Ch	Clock Control	
032h	Error Interrupt Status		030h	Normal Interrupt Status	
036h	Error Interrupt Status Enable		034h	Normal Interrupt Status Enable	
03Ah	Error Interrupt Signal Enable		038h	Normal Interrupt Signal Enable	
03Eh	Host Control 2		03Ch	Auto CMD Error Status	
042h	Capabilities		040h	Capabilities	
046h	Capabilities		044h	Capabilities	
04Ah	Maximum Current Capabilities		048h	Maximum Current Capabilities	
04Eh	Maximum Current Capabilities(Reserved)		04Ch	Maximum Current Capabilities(Reserved)	
052h	Force Event for Error Interrupt Status		050h	Force Event for Auto CMD Error Status	
056h	---		054h	---	ADMA Error Status
05Ah	ADMA System Address [31:16]		058h	ADMA System Address [15:00]	
05Eh	ADMA System Address [63:48]		05Ch	ADMA System Address [47:32]	
062h	Preset Value		060h	Preset Value	
066h	Preset Value		064h	Preset Value	
06Ah	Preset Value		068h	Preset Value	
06Eh	Preset Value		06Ch	Preset Value	
---	---		---	---	
0E2h	Shared Bus Control(High)		0E0h	Shared Bus Control(Low)	
---	---		---	---	
0F2h	---		0F0h	---	
---	---		---	---	
0FEh	Host Controller Version		0FCh	Slot Interrupt Status	

图 23.11: SD 主机控制寄存器映射

## 23.3.16.2 配置寄存器类型

配置寄存器字段被分配下表描述的一个属性：

Register Attribute	Description
RO	Read-only register: Register bits are read-only and cannot be altered by software or any reset operation. Writes to these bits are ignored.
ROC	Read-only status: These bits are initialized to zero at reset. Writes to these bits are ignored.
RW	Read-Write register: Register bits are read-write and may be either set or cleared by software to the desired state.
RW1C	Read-only status, Write-1-to-clear status: Register bits indicate status when read, a set bit indicating a status event may be cleared by writing a 1. Writing a 0 to RW1C bits has no effect.
RWAC	Read-Write, automatic clear register: The Host Driver requests a Host Controller operation by setting the bit. The Host Controllers shall clear the bit automatically when the operation of complete. Writing a 0 to RWAC bits has no effect.
HwInit	Hardware Initialized: Register bits are initialized by firmware or hardware mechanisms such as pin strapping or serial EEPROM. Bits are read-only after initialization, and writes to these bits are ignored.
RsVd	Reserved. These bits are initialized to zero, and writes to them are ignored.
WO	Write-only register. It is not physically implemented register. Rather, it is an address at which registers can be written.

图 23.12: 寄存器（和寄存器位字段）类型

### 23.3.16.3 寄存器初始值

主机控制器在上电复位时将所有寄存器设置为其初始值，所有其他寄存器的默认值应为设置为零的所有位。能力寄存器和最大当前能力寄存器的值取决于主机控制器，主机控制器版本寄存器的值也取决于主机控制器。

### 23.3.16.4 寄存器的保留位

“保留”表示该位可定义供将来使用，当前设置为 0。这些位应写为 0。

## 24.1 简介

安全数字输入输出（Secure Digital Input and Output, SDIO）是在 SD 存储卡接口基础上发展起来的一种外设接口。SDIO 卡兼容 SD 存储卡，目的是提供高速数据 I/O 与低功耗的移动电子设备。

## 24.2 主要特征

- 针对便携式和固定式应用
- 最小或不需要修改 SD 物理总线
- 内存驱动软件的最小更改
- 可用于特殊应用的扩展物理形状因子
- 即插即用
- 多功能支持，包括多个 I/O、组合 I/O 和内存
- 一张卡最多支持 7 个 I/O 功能和 1 个内存
- 允许卡中断主机

## 24.3 功能描述

### 24.3.1 SDIO 信号定义

#### 24.3.1.1 SDIO 卡类型

SDIO 卡分为两种类型，全速 SDIO 卡和低速 SDIO 卡。在 0~25MHz 的全时钟范围内，全速卡支持 SPI、1 位 SD 和 4 位 SD 传输模式。全速 SDIO 卡的数据传输速率超过 100Mb/秒（10MB/秒）。低速 SDIO 卡只需要 SPI 和 1 位 SD 传输模式，支持 0~400KHz 的全时钟范围。低速卡的预期用途是用最少的硬件支持低速 I/O 能力，低速卡支持调制解调器、条码扫描器、GPS 接收器等功能。

### 24.3.1.2 SDIO 卡模式

有 3 种为 SD 存储卡定义的信号模式，也适用于 SDIO 卡：**SPI 模式**：在这种模式下，引脚 8 被用作中断引脚，所有其他引脚和信令协议都与 SD 物理规范相同。**1 位 SD 数据传输模式**：在这种模式下，数据只在 DAT[0] 引脚上传输。引脚 8 被用作中断引脚，所有其他引脚和信令协议都与 SD 内存规范相同。**4 位 SD 数据传输模式**：在这种模式下，数据在所有 4 个数据引脚 DAT[3:0] 上传输。此模式下中断引脚不可独占使用，因为它被用作数据传输线。因此，如果需要中断功能，则需要特殊的定时来提供中断。**4 位 SD 模式**提供最快的数据传输可能，最高可达 100Mb/秒。

### 24.3.1.3 信号引脚

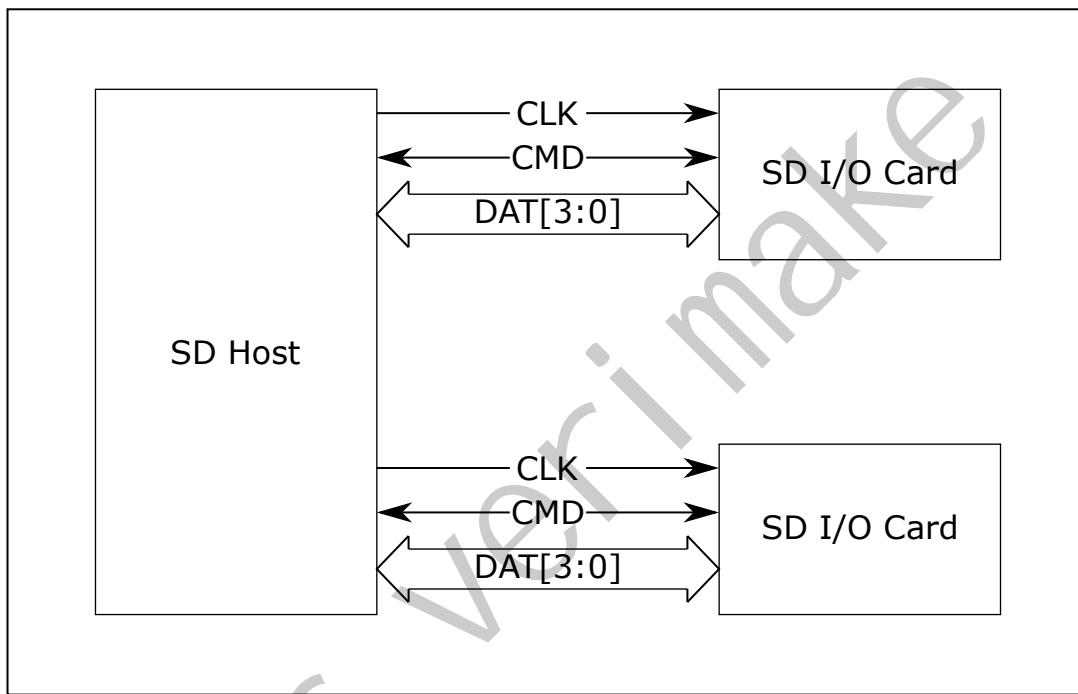


图 24.1: 2 个 4 位 SDIO 卡信号连接示意图

### 24.3.2 SDIO 卡初始化

#### 24.3.2.1 I/O 卡初始化差异

SDIO 规范的一个要求是，当插入 SDIO 卡时，不应导致非 I/O 感知主机失败。为了防止在非 I/O 感知的主机中操作 I/O 功能，需要更改 SD 卡识别模式流程图。添加了一个新的命令 (IO\_SEND\_OP\_COND, CMD5) 来替换 ACMD41，通过 I/O 感知主机进行 SDIO 初始化。复位或上电后，卡上的所有 I/O 功能都被禁用，当 CS 为低时，除 CMD5 或 CMD0 外，卡的 I/O 部分不会执行任何操作。如果有 SD 内存安装在卡上，该内存应正常响应所有正常的强制内存命令。仅 I/O 卡不得响应 ACMD41，因此最初显示为 MMC 卡。仅 I/O 卡也不得响应于初始化 MMC 卡的 CMD1，并显示为非响应卡。然后主机放弃并禁用该卡。因此，非感知主机不接收来自仅 I/O 卡的响应，并强制其进入非活动状态。下图显示了具有非 I/O 感知主机的 I/O 卡的操作，实线为实际路径，而虚线未执行：

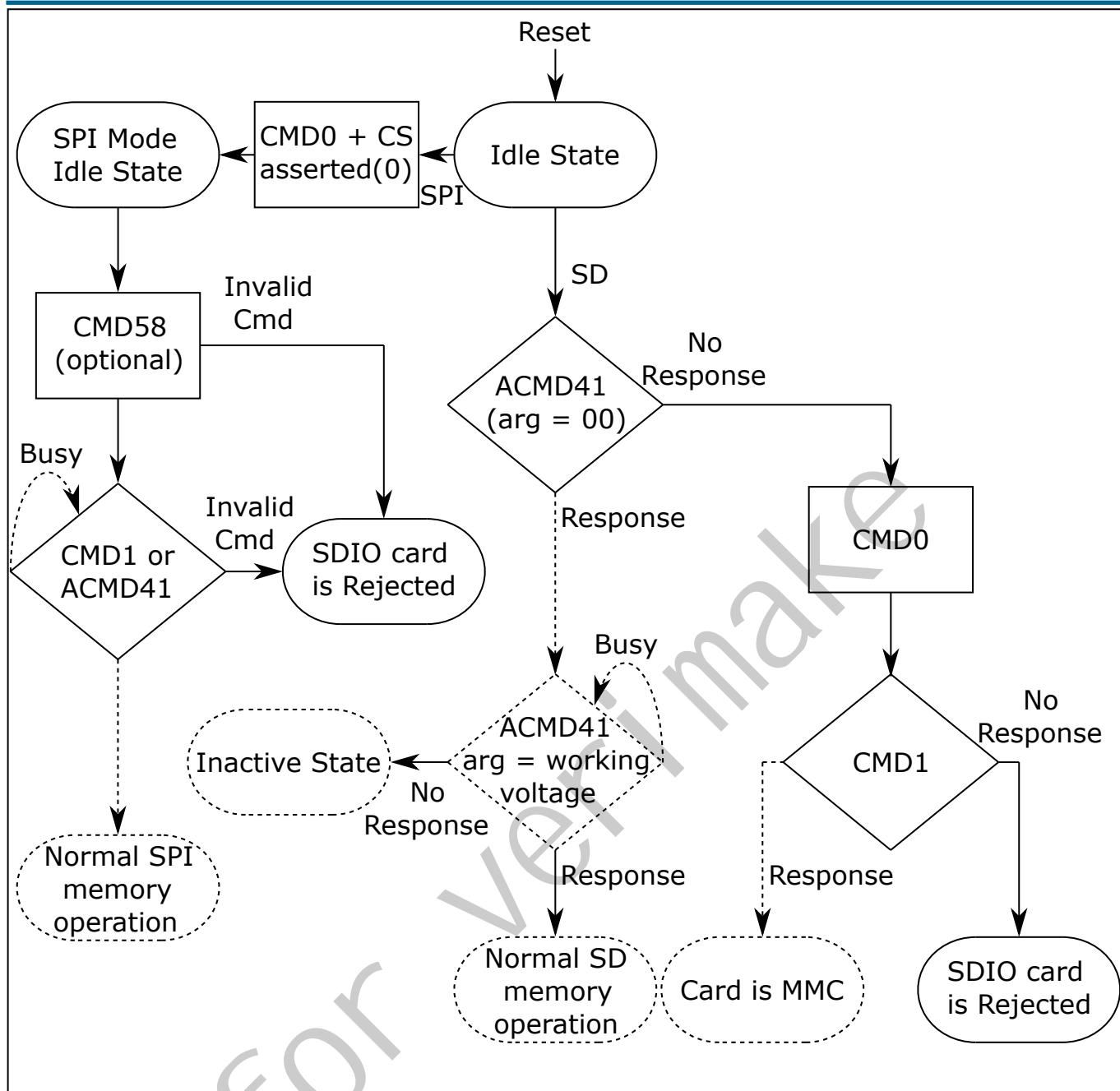


图 24.2: SDIO 对非 I/O 感知初始化的响应

### 24.3.2.2 IO\_SEND\_OP\_COND 命令 (CMD5)

IO\_SEND\_OP\_COND 命令 (CMD5) 对 SDIO 卡的功能类似于对 SD 存储卡的 ACMD41 的操作，用于查询 I/O 卡所需要的电压范围。CMD5 的正常响应是 SD 或 SPI 格式的 R4。CMD5 的格式如下图所示：

S	D	Command Index 000101b	Stuff Bits	I/O OCR	CRC7	E
1	1	6	8	24	7	1

图 24.3: IO\_SEND\_OP\_COND 命令

其中：

- S: 起始位。总是 0;
- D: 方向。总是 1，表示从主机到卡的传输;
- Command Index: 使用值 000101b 来标识 CMD5 命令;
- Stuff Bits: 不使用，应设置为 0;
- I/O OCR: 操作条件寄存器。VDD 支持的最小值和最大值;
- CRC7: 7 位 CRC 数据;
- E: 结束位。总是 1。

### 24.3.2.3 IO\_SEND\_OP\_COND 响应 (R4)

接收到 CMD5 的 SDIO 卡应响应一个唯一的 SDIO 响应 R4。R4 在 SD 和 SPI 模式下的格式分别如下图所示：

S	D	Reserved	C	Number of I/O functions	Memory Present	Stuff Bits	I/O OCR	Reserved	E
1	1	6	1	3	1	3	24	7	1

图 24.4: SD 模式下的响应 R4

Modified R1	C	Number of I/O functions	Memory Present	Stuff Bits	I/O OCR
8	1	3	1	3	24

图 24.5: SPI 模式下的响应 R4

其中:

- S: 起始位。总是 0;
- D: 方向。总是 0, 表示从卡到主机的传输;
- Reserved: 为将来使用而保留的位。这些位应设置为 1;
- C: 如果卡在初始化完后准备好了操作, 则设置为 1;
- Stuff Bits: 不使用, 应设置为 0;
- I/O OCR: 操作条件寄存器。VDD 支持的最小值和最大值;
- E: 结束位。总是 1。
- Memory Present: 如果卡还包含 SD 内存, 则设置为 1。如果卡仅为 I/O, 则设置为 0;
- Number of I/O Functions: 指示此卡支持的 I/O 功能总数, 范围是 0~7。需要注意的是, 此计数不包括功能 0 的所有 I/O 卡上的公共区域。I/O 功能应从功能 1 开始按顺序执行;
- Modified R1: SD 物理规范中描述的 SPI R1 响应字节针对 I/O 的改进如下;

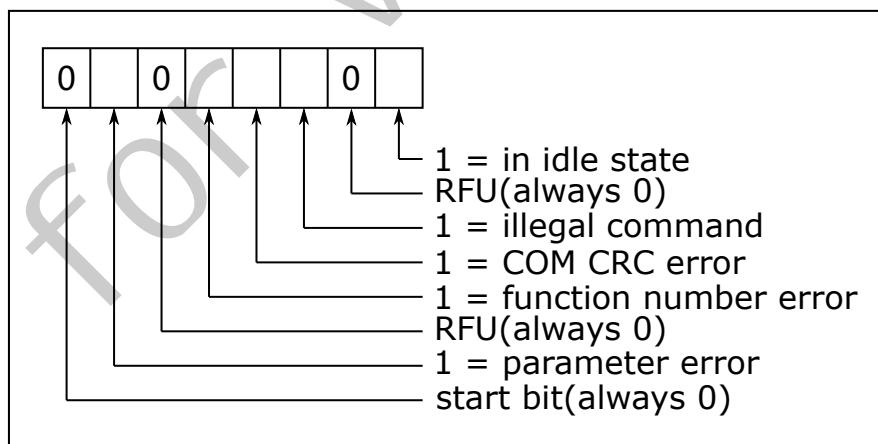


图 24.6: 改进的 R1 响应

#### 24.3.2.4 组合卡的特殊初始化注意事项

主机在初始化组合卡时必须注意一些特殊情况（SDIO 加上同一卡上的 SD 内存）。这是因为组合卡的实现实际上可以在同一个包中使用两个单独的控制器（内存和 I/O），并共享相同的总线线路。主机必须检测并正确配置组合卡的两个部分（控制器），以防止 SDIO 和 SD 内存控制器之间发生冲突。这些问题是由两个控制器对复位（硬或软）的响应不同而引起的。另一个问题是内存控制器中存在的 RCA（相对卡地址）的值。上述注意事项仅适用于 SD 1 位和 SD 4 位模式。在 SPI 模式下，卡选择/取消选择使用硬件 CS 线而不是 RCA 完成。

### 24.3.3 与 SD 内存规范的差异

#### 24.3.3.1 SDIO 命令列表

使用 SD 总线接口时 SD 内存和 SDIO 卡接受的命令列表如下图所示：

Supported Commands	Abbreviation	SDMEM System	SDIO System	Comments
CMD0	GO_IDLE_STATE	Mandatory	Mandatory	Used to change from SD to SPI mode
CMD2	ALL_SEND_CID	Mandatory		CID not supported by SDIO
CMD3	SEND_RELATIVE_ADDR	Mandatory	Mandatory	
CMD4	SET_DSR	Optional		DSR not supported by SDIO
CMD5	IO_SEND_OP_COND		Mandatory	
CMD6	SWITCH_FUNC	Mandatory	Mandatory	
CMD7	SELECT/DESELECT_CARD	Mandatory	Mandatory	
CMD9	SEND_CSD	Mandatory		CSD not supported by SDIO
CMD10	SEND_CID	Mandatory		CID not supported by SDIO
CMD12	STOP_TRANSMISSION	Mandatory		
CMD13	SEND_STATUS	Mandatory		Card Status includes only SDMEM information
CMD15	GO_INACTIVE_STATE	Mandatory	Mandatory	
CMD16	SET_BLOCKLEN	Mandatory		
CMD17	READ_SINGLE_BLOCK	Mandatory		
CMD18	READ_MULTIPLE_BLOCK	Mandatory		
CMD24	WRITE_BLOCK	Mandatory		
CMD25	WRITE_MULTIPLE_BLOCK	Mandatory		
CMD27	PROGRAM_CSD	Mandatory		CSD not supported by SDIO
CMD28	SET_WRITE_PROT	Optional		
CMD29	CLR_WRITE_PROT	Optional		
CMD30	SEND_WRITE_PROT	Optional		
CMD32	ERASE_WR_BLK_START	Mandatory		
CMD33	ERASE_WR_BLK_END	Mandatory		
CMD38	ERASE	Mandatory		
CMD42	LOCK_UNLOCK	Optional		
CMD52	IO_RW_DIRECT		Mandatory	
CMD53	IO_RW_EXTENDED		Mandatory	Block mode is optional
CMD55	APP_CMD	Mandatory		
CMD56	GEN_CMD	Mandatory		
ACMD6	SET_BUS_WIDTH	Mandatory		
ACMD13	SD_STATUS	Mandatory		
ACMD22	SEND_NUM_WR_BLOCKS	Mandatory		
ACMD23	SET_WR_BLK_ERASE_COUNT	Mandatory		
ACMD41	SD_APP_OP_COND	Mandatory		
ACMD42	SET_CLR_CARD_DETECT	Mandatory		
ACMD51	SEND_SCR	Mandatory		SCR not supported by SDIO

图 24.7: SD 模式命令列表

使用 SPI 总线接口时 SD 内存和 SDIO 卡接受的命令列表如下图所示:

Supported Commands	Abbreviation	SDMEM System	SDIO System	Comments
CMD0	GO_IDLE_STATE	Mandatory	Mandatory	Used to change from SD to SPI mode
CMD1	SEND_OP_COND	Mandatory		
CMD5	IO_SEND_OP_COND		Mandatory	
CMD6	SWITCH_FUNC	Mandatory	Mandatory	
CMD9	SEND_CSD	Mandatory		CSD not supported by SDIO
CMD10	SEND_CID	Mandatory		CID not supported by SDIO
CMD12	STOP_TRANSMISSION	Mandatory		
CMD13	SEND_STATUS	Mandatory		Card Status includes only SDMEM information
CMD16	SET_BLOCKLEN	Mandatory		
CMD17	READ_SINGLE_BLOCK	Mandatory		
CMD18	READ_MULTIPLE_BLOCK	Mandatory		
CMD24	WRITE_BLOCK	Mandatory		
CMD25	WRITE_MULTIPLE_BLOCK	Mandatory		
CMD27	PROGRAM_CSD	Mandatory		CSD not supported by SDIO
CMD28	SET_WRITE_PROT	Optional		
CMD29	CLR_WRITE_PROT	Optional		
CMD30	SEND_WRITE_PROT	Optional		
CMD32	ERASE_WR_BLK_START	Mandatory		
CMD33	ERASE_WR_BLK_END	Mandatory		
CMD38	ERASE	Mandatory		
CMD42	LOCK_UNLOCK	Optional		
CMD52	IO_RW_DIRECT		Mandatory	
CMD53	IO_RW_EXTENDED		Mandatory	Block mode is optional
CMD55	APP_CMD	Mandatory		
CMD56	GEN_CMD	Mandatory		
CMD58	READ_OCR	Mandatory		
CMD59	CRC_ON_OFF	Mandatory	Mandatory	
ACMD13	SD_STATUS	Mandatory		
ACMD22	SEND_NUM_WR_BLOCKS	Mandatory		
ACMD23	SET_WR_BLK_ERASE_COUNT	Mandatory		
ACMD41	SD_APP_OP_COND	Mandatory		
ACMD42	SET_CLR_CARD_DETECT	Mandatory		
ACMD51	SEND_SCR	Mandatory		SCR includes only SDMEM information

图 24.8: SPI 模式命令列表

### 24.3.3.2 不支持的 SD 内存命令

SDIO 专用卡或组合卡的 I/O 部分不支持 SD 存储卡所需的几个命令。其中一些命令在 SDIO 卡中没有用处，例如 Erase 命令，因此在 SDIO 中不受支持。此外，有几个用于 SD 存储卡的命令在与卡的 SDIO 部分一起使用时具有不同的命令。下表列出了这些 SD 内存命令和等效的 SDIO 命令：

SD Memory Command	SDIO Command	Comments
CMD0	CMD52 (写入CCCR中的I/O复位)	重置命令 (CMD0) 仅用于组合卡的内存或内存部分。要重置仅I/O卡或组合卡的I/O部分，请使用CMD52将a1写入CCCR中的RES位 (寄存器6的位3)。请注意，在SD模式下，CMD0仅用于指示进入SPI模式，应予以支持。仅I/O卡或组合卡的I/O部分不会使用CMD0重置
CMD12	CMD52 (写入I/O中止)	为了中止数据块传输，SD内存使用CMD12。要中止I/O事务，请使用CMD52写入CCCR中的中止寄存器 (寄存器6的位2:0)
CMD16	CMD52 (写入I/O块长度)	CMD16设置SD内存的块长度。要为每个I/O函数设置块长度，请使用CMD52在FBR中写入块长度
CMD2	无	仅SDIO卡中不存在CID寄存器
CMD4	无	仅SDIO卡中不存在DSR寄存器
CMD9	无	仅SDIO卡中不存在CSD寄存器
CMD10	无	仅SDIO卡中不存在CID寄存器
CMD13	无	仅SDIO卡或组合卡的I/O部分不支持SD内存使用的相同的SEND_STATUS (CMD13) 协议
ACMD6	CMD52 (写入CCCR中的Bus_Width[1:0])	SET_BUS_WIDTH由对CCCR的写入处理
ACMD13	无	仅SDIO卡中不存在SD状态寄存器
ACMD41	CMD5	SDIO卡和主机使用IO_SEND_OP_COND命令 (CMD5)
ACMD42	CMD52	在SD模式下，DAT[3]上的上拉电阻通过写入CCCR中的CD禁用位来控制。对于组合卡，此电阻启用，除非内存和I/O控制寄存器都设置为禁用电阻
ACMD51	无	仅SDIO卡中不存在SCR寄存器
CMD17 CMD18 CMD24 CMD25	CMD53	I/O块操作使用CMD53，而不是内存块读/写命令

图 24.9: 不支持的 SD 内存命令

### 24.3.3.3 改进的 R6 响应

存储卡对 CMD3 的正常响应为 R6，如下表所示：

Bit position	47	46	[45:40]	[39:8]Argument field	[7:1]	0	
Width(bits)	1	1	6	16	16	7	1
Value	'0'	'0'	X	X	X	X	'1'
Description	Start bit	Direction bit	Command index ('000011')	New published RCA [31:16] of the card	[15:0] Card status	CRC7	end bit

图 24.10: CMD3 的 R6 响应

当 CMD3 被发送到仅 I/O 卡时，卡状态位 (8~23) 会发生更改。在这种情况下，16 位响应应为下表所示的仅 SDIO 值：

Bits	Identifier	Type	Value	Description	Clear Condition
15	COM_CRC_ERROR	E R	'0' = no error '1' = error	上一个命令的CRC检查失败	B
14	ILLEGAL_COMMAND	E R	'0' = no error '1' = error	命令对于卡状态不合法	B
13	ERROR	E R X	'0' = no error '1' = error	操作过程中发生常规错误或未知错误	C
12:0	未定义。仅SDIO卡的读数应为0。主机应该忽略这些位				

图 24.11: SDIO R6 状态位

### 24.3.3.4 SDIO 复位

为了复位 SDIO 卡或组合卡的 SDIO 部分中的所有功能，定义了一种不同于用于 SD 内存的方法。复位命令（CMD0）只用于内存或组合卡的内存部分。要复位仅 I/O 卡或组合卡的 I/O 部分，请使用 CMD52 将 1 写入 CCCR 中的 RES 位（寄存器 6 的位 3）。要注意的是，在 SD 模式下，CMD0 仅用于指示进入 SPI 模式，CMD0 不会复位仅 I/O 卡或组合卡的 I/O 部分。

### 24.3.3.5 总线宽度

对于 SD 存储卡，SD 模式的总线宽度使用 ACMD6 设置。SDIO 卡使用 CMD52 写入 CCCR 来选择总线宽度。对于组合卡，两种选择方法都存在。这种情况下主机应在开始任何数据传输之前，通过使用具有相同宽度设置的 ACMD6 和写入 CCCR 的 CMD52，在两个位置设置总线宽度。

### 24.3.3.6 卡检测电阻

SD 内存和 I/O 卡使用 DAT[3] 上的上拉电阻来检测卡插入。SD 存储器和 SDIO 之间启用/禁用此电阻的步骤不同。SD 内存使用 ACMD42 控制该电阻，而 SDIO 使用 CMD52 写入 CCCR。如果是组合卡，两个控制方式都存在，并应由主机管理。对于组合卡，仅当内存和 I/O 控制寄存器都启用电阻时，电阻才启用。也就是说，通电后，主机应使用 ACMD42 对内存控制器或对 SDIO 控制器的 CCCR 写入禁用电阻，因为电阻启用是两个启用中的逻辑与。通电后，两个位置默认为电阻启用。请注意，在 I/O 复位后，I/O 电阻启用不会改变。

### 24.3.3.7 数据传输块大小

SDIO 卡可以多字节（1 到 512 字节）或可选块格式传输数据，而 SD 存储卡固定在块传输模式下。SD 物理规范将数据传输的块大小限制为 2 的次方（即 512、1024、2048），除非使用部分读写。SDIO 规范允许从 1 字节到 2048 字节的任何块大小，以适应 I/O 功能的各种自然块大小。值得注意的是，SDIO 卡功能可在 CIS 中定义小于上述最大值的最大块大小或字节数。

### 24.3.3.8 数据传输中止

与 SD 存储卡通信的主机使用 CMD12 中止从/向该卡读取或写入数据的传输。对于 SDIO 卡，CMD12 中止由对 CCCR 中 ASx 位的写入代替。通常，中止用于停止无限块传输（块计数 =0）。如果要传输的块的数量是确定的，建议主机发出具有正确块计数的块命令，而不是使用无限计数并在正确的时间中止数据。

## 24.3.4 新的 I/O 读或写命令

增加了两条额外的数据传输指令以支持 I/O。IO\_RW\_DIRECT 是一个类似于 MMC “快速 I/O” 命令的直接 I/O 命令，IO\_RW\_EXTENDED 允许使用字节或块地址进行快速访问。这两个命令都属于类 9 (I/O 命令)。

### 24.3.4.1 IO\_RW\_DIRECT 命令 (CMD52)

IO\_RW\_DIRECT 是在任何 I/O 功能，包括公共 I/O 区 (CIA) 中访问 128K 寄存器空间内的单个寄存器的最简单方法。此命令读取或写入 1 个字节仅使用 1 个命令/响应对。通常用于初始化寄存器或监视 I/O 功能的状态值。此命令是读取或写入单个 I/O 寄存器的最快方法，因为它只需要单个命令/响应对。该命令结构如下图所示：

S	D	Command Index 110100b	R/W flag	Function Number	RAW flag	Stuff	Register Address	Stuff	Write Data or Stuff Bits	CRC7	E
1	1	6	1	3	1	1	17	1	8	7	1

图 24.12: IO\_RW\_DIRECT 命令

其中：

- **S:** 起始位。总是 0；
- **D:** 方向。总是 1，表示从主机到卡的传输；
- **Command Index:** 使用值 110100b 来标识 IO\_RW\_DIRECT 命令；
- **R/W Flag:** 该位确定 I/O 操作的方向。如果该位为 0，则该命令应在功能号指定的地址和主机的寄存器地址从 SDIO 卡读取数据。数据字节在响应 R5 中返回。如果该位设置为 1，则命令应将写入数据字段中的字节写入由功能号和寄存器地址寻址的 I/O 位置。如果原始标志为 0，则应读取寄存器中写入的数据，并在响应中返回该值；
- **RAW Flag:** 写入后读取标志。如果该位设置为 1 且 R/W 标志设置为 1，则命令应在写入后读取寄存器的值。这对于允许写入控制寄存器和读取同一地址的状态非常有用。如果清除该位，R5 响应中返回的值应与命令中的写入数据相同。如果设置了该位，R5 响应的数据字段应包含写入操作后从寻址寄存器读取的值；
- **Function Number:** 希望读取或写入的 I/O 卡中的功能编号。功能 0 选择公共 I/O 区域 (CIA)；
- **Register Address:** 这是所选功能中要读取或写入的数据字节的地址。有 17 位地址可用，因此寄存器位于该功能的前 128K (131072) 地址内；
- **Write Data/Stuff Bits:** 对于直接写入命令 (R/W=1)，这是写入所选地址的字节。对于直接读取 (R/W=0)，不使用此字段，应将其设置为 0；
- **CRC7:** 7 位 CRC 数据；
- **E:** 结束位。总是 1。

### 24.3.4.2 IO\_RW\_DIRECT 响应 (R5)

SDIO 卡对 CMD52 的响应应采用两种格式之一。如果卡和主机之间的通信处于 1 位或 4 位 SD 模式，则响应应为 48 位响应 (R5)。如果操作是读取命令，则正在读取的数据将作为 8 位值返回。此外，还返回 15 位状态信息。如下图所示：

S	D	Command Index 110100b	Stuff	Response Flags 7-----Bit-----0	Read or Write Data	CRC7	E
1	1	6	16	8	8	7	1

图 24.13: SD 模式下的 IO\_RW\_DIRECT 响应

其中：

- S: 起始位。总是 0;
- D: 方向。总是 0，表示从卡到主机的传输;
- Command Index: 使用值 110100b 来标识 IO\_RW\_DIRECT 命令;
- Stuff Bits: 不使用，应设置为 0;
- Response Flags: 指示 SDIO 卡状态的 8 位标志数据;
- Read or Write Data: 对于设置了原始标志 (RAW=1) 的 I/O 写入 (R/W=1)，该字段应包含写入命令中包含的数据后从寻址寄存器读取的值。注意，在这种情况下，根据硬件的设计，读回数据可能与写入寄存器的数据不同。对于原始位为 0 的 I/O 写入，SDIO 功能不应进行写后读取操作，且该字段中的数据应与写入命令中的数据字节相同。对于 I/O 读取 (R/W=0)，从该 I/O 位置读取的实际值将在该字段中返回。
- CRC7: 7 位 CRC 数据;
- E: 结束位。总是 1。

如果通信使用 SPI 模式，响应应为 16 位 R5 响应。如果操作是读取命令，则正在读取的数据将作为 8 位值返回。此外，在 SPI R1 响应字节中返回 8 位状态信息。如下图所示：

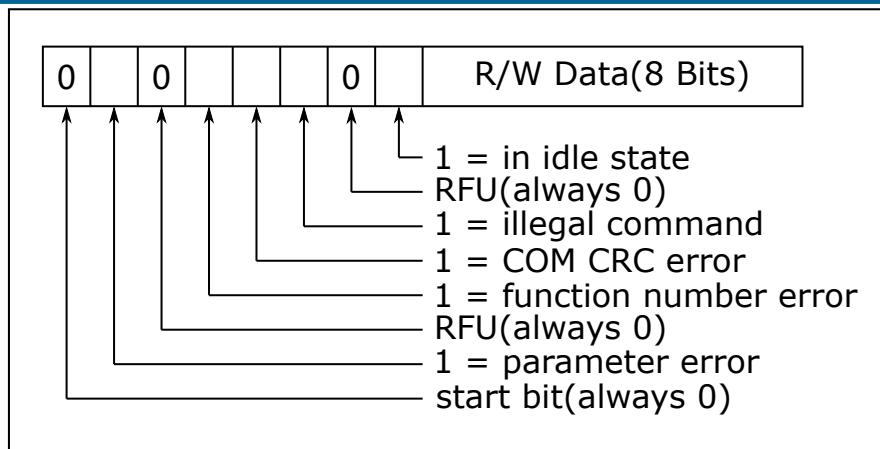


图 24.14: SPI 模式下的 IO\_RW\_DIRECT 响应

注: 读/写 (R/W) 数据与 SD R5 响应所述的读/写数据相同。SPI 模式中的参数错误状态对应于超出范围和 SD 模式响应中的错误。对于 CMD53, 还应使用数据错误标记指示超出范围和错误。

#### 24.3.4.3 IO\_RW\_EXTENDED 命令 (CMD53)

为了用一个命令读写多个 I/O 寄存器, 定义了一个新命令 IO\_RW\_EXTENDED。此命令包含在命令类 9 (I/O 命令) 中。此命令允许使用单个命令读取或写入大量 I/O 寄存器。因为这是一个数据传输命令, 所以它提供了可能的最高传输速率。IO\_RW\_EXTENDED 命令如下图所示:

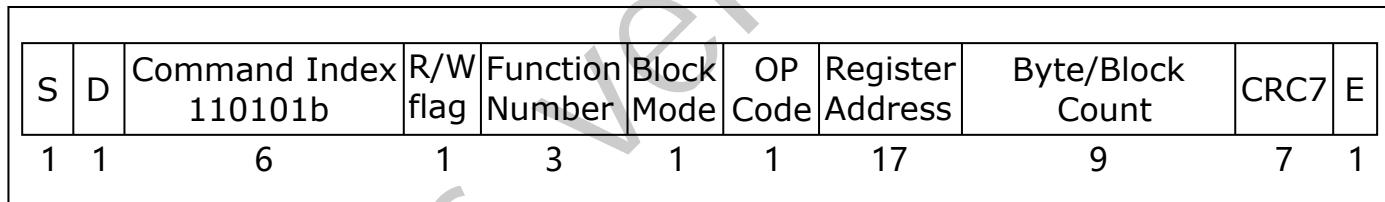


图 24.15: IO\_RW\_EXTENDED 命令

其中:

- **S:** 起始位。总是 0;
- **D:** 方向。总是 1, 表示从主机到卡的传输;
- **Command Index:** 使用值 110101b 来标识 IO\_RW\_EXTENDED 命令;
- **R/W Flag:** 该位确定 I/O 操作的方向。如果该位为 0, 则该命令将从 SDIO 卡中以功能号和寄存器地址指定的地址向主机读取数据。读取的数据应在 DAT[x] 行上返回。如果该位设置为 1, 则命令应将 DAT[x] 行中的字节写入由功能号和寄存器地址寻址的 I/O 位置;
- **Function Number:** 希望读取或写入的 I/O 卡中的功能编号。功能 0 选择公共 I/O 区域 (CIA);
- **Block Mode:** (可选) 如果该位设置为 1, 则表示读取或写入操作应在块基础上执行, 而不是在正常字节基础上执

行。如果设置了该位，字节/块计数值应包含要读取/写入的块数。通过将块大小写入 FBR 中的 I/O 块大小寄存器来设置功能 1-7 的块大小。功能 0 的块大小通过写入 CCCR 中的 FN0 块大小寄存器来设置。块 I/O 模式的卡和主机支持是可选的。主机可以通过读取 CCCR 中的卡支持 MBIO 位 (SMB) 来确定卡是否支持块 I/O。块模式 =1 时使用的块大小和块模式 =0 时使用的每个命令的最大字节数可以从元组 TPLFE\_MAX\_BLK\_SIZE 中的 CIS 中按功能读取；

- **OP code:** 定义读/写操作。0 表示对固定地址的多字节读/写，1 表示对递增地址的多字节读/写；
- **Register Address:** 要读取或写入的 I/O 寄存器的起始地址。范围为 0~0x1FFF；
- **Byte/Block Count:** 如果命令以字节（块模式 =0）操作，则此字段包含要读取或写入的字节数。0x000 的值将导致 512 字节被读取或写入；
- **CRC7:** 7 位 CRC 数据；
- **E:** 结束位。总是 1。

### 24.3.5 SDIO 卡内部操作

I/O 访问与内存的不同之处在于，寄存器可以单独或直接写入和读取，而无需 FAT 文件结构或块的概念（尽管支持块访问）。这些寄存器允许访问 I/O 数据、控制 I/O 功能、报告状态或向主机传输 I/O 数据。SD 内存依赖于固定块长度的概念，命令读取/写入这些固定大小块的倍数。I/O 可能有固定的块长度，也可能没有固定的块长度，读取大小可能不同于写入大小。因此，I/O 操作可能基于长度（字节计数）或块大小。

#### 24.3.5.1 概述

每个 SDIO 卡可能有 1 到 7 个功能，以及内置的一个存储功能。功能是一个自包含的 I/O 设备。I/O 功能可能相同，也可能完全不同。所有 I/O 功能都组织为寄存器集合。每个 I/O 功能最多可能有 131072 ( $2^{17}$ ) 个寄存器。这些寄存器及其各个位可以是只读 (RO)、只写 (WO) 或读/写 (R/W)。这些寄存器在卡中可以是 8、16 或 32 位宽。所有寻址都基于字节访问。这些寄存器可以一次写入和/或读取一个，乘法到同一地址或乘法到递增地址。单个 R/W 访问通常用于初始化 I/O 功能或读取单个状态或数据值。对固定地址的多次读取用于从卡中的数据 FIFO 寄存器读取或写入数据。读到递增地址用于向卡内的 RAM 区域读写数据集合。下图显示了 SDIO 卡的 CIA 和可选 CSA 空间的映射：

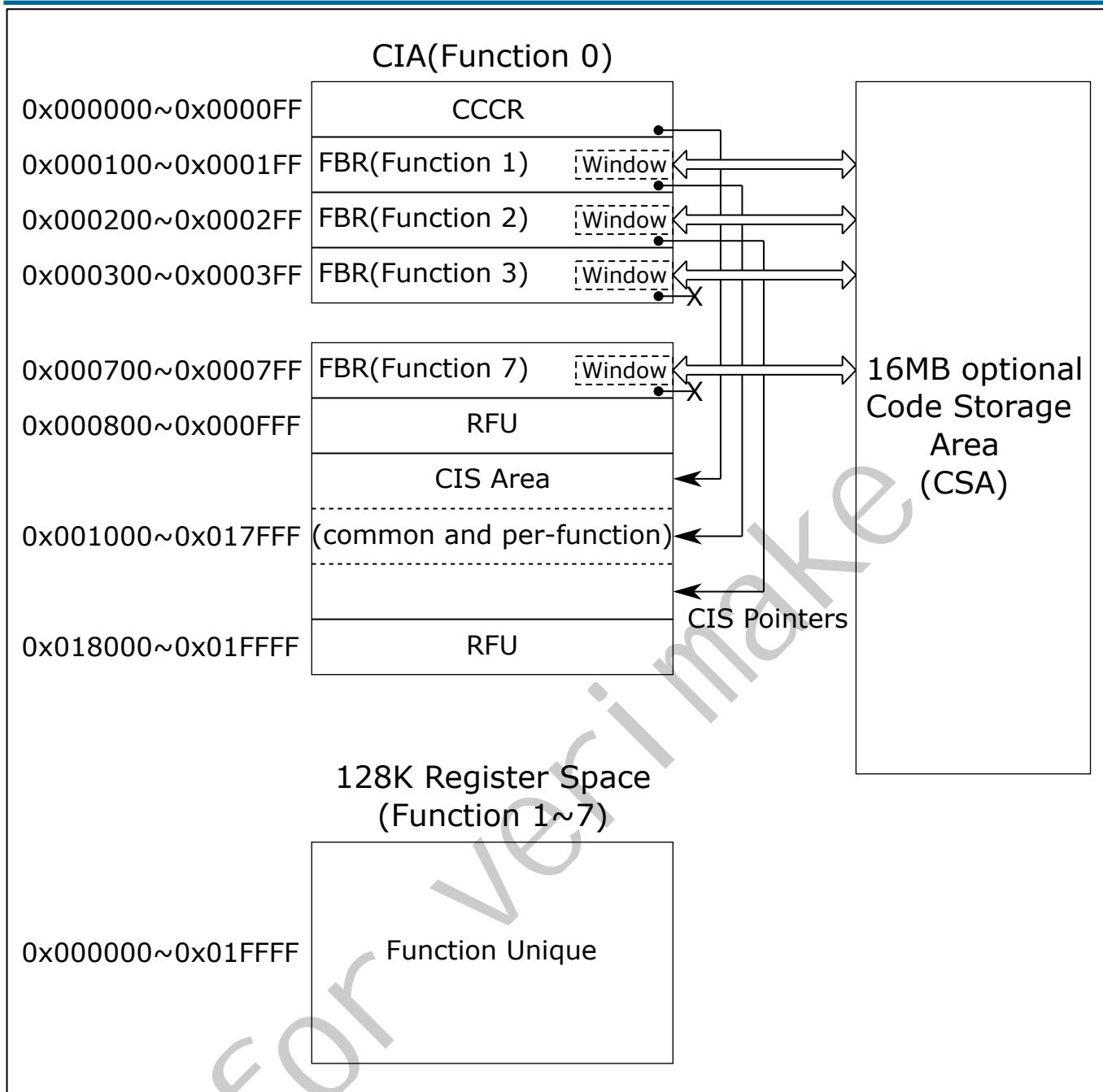


图 24.16: SDIO 内部映射

### 24.3.5.2 寄存器访问时间

仅 SDIO 卡和组合卡 SDIO 部分中的所有寄存器应在 1 秒内完成读写数据传输。此超时值与请求数据在 DAT[x] 线上传输到主机或从主机传输的时间有关，而不是与命令和响应之间的时间有关。该等待时间由卡向主机发出信号，使用 busy 进行写入或延迟开始位进行读取操作。主机可以使用 1 秒作为无响应位置的超时值。

### 24.3.5.3 中断

所有 SDIO 主机都应支持硬件中断。如果主机不支持中断，则可能难以使用期望对中断条件做出快速响应的 SDIO 卡。SDIO 或组合卡中的每个功能都可以根据需要实现中断。SDIO 功能上使用的中断类型通常称为“电平敏感”。电平敏感是指任何功能可在任何时候发出中断信号，但一旦该功能发出中断信号，在主机移除中断原因或命令这样做之前，不得释放（停止信号）中断。由于只有一条中断线，它可能被多个中断源共享。该功能应继续发出中断信号，直到主机响应并清除中断。由于多个中断可能同时处于活动状态，因此主机有责任确定中断源并根据需要进行处理。这是通过使用两位（中断启用和中断挂起）在 SDIO 功能上完成的。每个可能产生中断的功能都有一个中断启用位。此外，SDIO 卡有一个主中断启用，用于控制所有功能。如果功能的启用和卡的主启用都已设置，则仅向 SD 总线发送中断信号。第二个中断位称为中断挂起。该只读位告诉主机哪些功能可能发出中断信号。每个可以生成中断的功能都有一个中断挂起位。这些位位于 CCCR 区域中。

### 24.3.5.4 挂起/恢复

在多功能 SDIO 或组合卡中，有多个设备（I/O 和内存）共享对 SD 总线的访问。为了允许在多个设备之间共享对主机的访问，SDIO 和组合卡可以实现挂起/恢复的可选概念。如果卡支持挂起/恢复，主机可能会暂时停止对一个功能或内存的数据传输操作（挂起），以便释放总线，以便向不同的功能或内存进行更高优先级的传输。完成此高优先级传输后，原始传输将在停止的位置重新开始（恢复）。支持挂起/恢复是每个卡的可选功能。如果执行挂起/恢复，则组合卡的内存（如果有）和除 0（CIA）之外的所有 I/O 功能应支持挂起/恢复。主机可以挂起多个事务，并按所需的任何顺序恢复它们。I/O 功能 0 不支持挂起/恢复。支持挂起/恢复的任何卡还应支持读取等待和直接命令（SRW 和 SDC=1）。注意，挂起/恢复仅针对 SD 1 和 4 位模式定义，它不适用于 SPI 传输。

### 24.3.5.5 读等待

根据 SD 物理规范构建的主机设备应控制 SDCLK，以便在主机无法接受更多数据时，停止执行多重读取命令的卡的读取数据块输出。在主机停止 SDCLK 期间，无法发出 CMD52。此限制会导致一个问题，即按照 SD 物理规范构建的主机设备无法在多个读取周期内执行 I/O 命令。为了消除此限制，SDIO 规范添加了读取等待控件，以使主机能够在多个读取周期中发出 CMD52。读取等待使用 DAT[2] 线允许主机向卡发送信号，以暂时停止卡发送读取数据。此功能对于 SDIO 或组合卡是可选的。但是，如果 SDIO 或组合支持读取等待，则所有功能和任何内存都应支持读取等待。支持挂起/恢复的任何卡也应支持读取等待。注意，读取等待仅针对 SD 1 和 4 位模式定义，它不适用于 SPI 传输。

### 24.3.5.6 数据传输中的 CMD52

如果卡支持直接命令，则可以在数据传输期间接受 CMD52。对于 SD 和 SPI 模式，如果在数据传输过程中发生错误，SDIO 卡应接受 CMD52，以允许 I/O 中止和复位，而不管 SDC 值的该位值如何。

### 24.3.5.7 固定内部映射

SDIO 卡具有固定的内部寄存器空间和功能唯一区域。固定区域包含有关卡的信息以及固定位置的某些强制和可选寄存器。固定位置允许任何主机获取有关卡的信息，并以常见方式执行简单操作，如启用。功能唯一区域是每个功能区域，由标准 SDIO 功能的应用规范或非标准功能的供应商定义。

### 24.3.5.8 公共 I/O 区 (CIA)

公共 I/O 区 (CIA) 应在所有 SDIO 卡上实施。主机通过对功能 0 的 I/O 读写访问 CIA。CIA 内的寄存器用于启用/禁用 I/O 功能的操作、控制中断的产生以及可选地加载软件以支持 I/O 功能。CIA 中的寄存器还提供有关功能能力和要求的信息。CIA 支持三种不同的寄存器结构。他们是：

- 卡通用控制寄存器 (CCCR)
- 功能基础寄存器 (FBR)
- 卡信息结构 (CIS)

### 24.3.5.9 卡通用控制寄存器 (CCCR)

卡通用控制寄存器允许快速主机检查和控制 I/O 卡在每个卡（主卡）和每个功能的基础上的启用和中断。CCCR 中的位混合为读/写和只读。如果 SDIO 卡上未提供 7 种可能功能中的任何一种，则与未使用功能相对应的位应全部为只读，并读取为 0。所有保留供将来使用的位 (RFU) 应为只读，并返回值 0。通电或复位后，所有可写位均设置为 0。当 I/O 功能被禁用时，即使在初始化之后也可以访问 CCCR。这允许主机在初始化后启用功能。

### 24.3.5.10 功能基础寄存器 (FBR)

除 CCCR 外，每个受支持的 I/O 功能都有一个 256 字节的区域，用于让主机快速确定每个功能的能力和要求，为每个功能启用电源选择，并启用软件加载。该区域的地址从 0x00n00 到 0x00nFF，其中 n 是功能号 (0x1 到 0x7)。

### 24.3.5.11 卡信息结构 (CIS)

卡信息结构提供有关卡和单个功能的更完整信息。CIS 是用于读取卡中所有 I/O 功能信息的公共区域。该设计基于 PCMCIA 标准化的 PC Card16 设计。支持 I/O 的所有卡应具有通用 CIS 和每个功能的 CIS。通过读取 0x1000~0x17FFF 区域来访问 CIS，该区域作为公共 CIS 和每个功能的存储区域服务于卡。公共区域和每个功能都有一个指针，指向该内存空间中 CIS 的开始。

### 24.3.5.12 多功能 SDIO 卡

多功能 SDIO 卡为卡上的每个功能配备一套单独的配置寄存器。多功能 SDIO 卡使用卡上所有功能共用的 CIS 和卡上每个功能专用的单独功能专用 CIS 的组合。通用 CIS 描述了卡上所有功能的通用功能。每个功能特定 CIS 描述 SDIO 卡上特定功能的细节特征。功能从 1 开始按顺序编号。CMD5 响应表示功能总数，其中包括“虚拟”功能。主机应根据 CMD5 响应遍历 CIS 条目。R5 响应的错误状态标志为“E R X”类型，可指示前一个命令中的错误。由于主机软件需要一种方法来确定检测到错误的功能，因此多功能 SDIO 卡只能在向同一功能发出的后续命令中返回 R5 错误状态标志。

### 24.3.5.13 使用 CMD53 设置块大小

主机通过写入 FBR 中的 16 位功能 I/O 块大小寄存器，为函数的多个块传输设置块大小。主机不得使用块模式设置为 1 的 CMD53 写入此寄存器。如果卡在执行 CMD53 之前检测到无效的块大小，且块模式设置为 1，则卡应在当前响应中指示超出范围错误，并且不应执行数据传输。这也将停止中断周期。

### 24.3.5.14 总线状态图

下图是 SDIO 卡的总线状态图，它显示了总线状态及其与 SDIO 命令和挂起/恢复的关系：

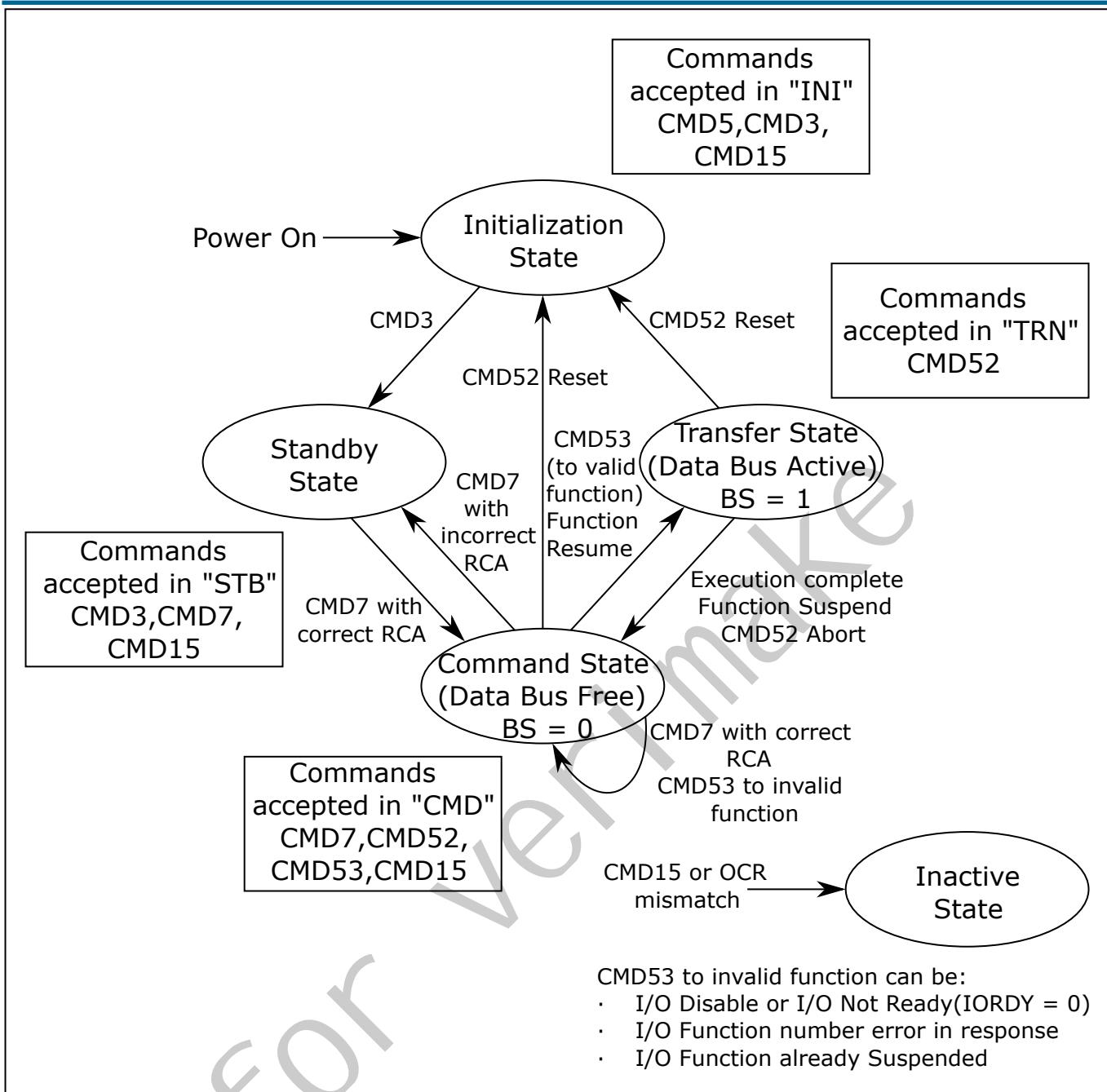


图 24.17: 总线状态机状态图

## 24.3.6 嵌入式 I/O 代码存储区 (CSA)

为了支持 SDIO 卡的“即插即用”概念，卡中包含的每个功能可能需要包含一块内存，用于存储驱动程序和/或应用程序。此外，由于同一 SDIO 卡可在多个不同的主机平台上使用，因此每个功能可能需要多个不同版本的代码。一种选择是将这些程序存储在组合卡的标准 SD 内存部分。或者，用于加载代码的标准访问装置包含在可选代码存储区域 (CSA) 中。CSA 是一个单独的 16MB 内存区域，可使用 FBR 寄存器中包含的 CSA 地址指针和 CSA 窗口寄存器进行访问。

### 24.3.6.1 CSA 访问

为了让主机访问功能的 CSA，它首先应确定该功能是否支持 CSA。主机读取地址 0x00n00 处的 FBR 寄存器，其中 n 是功能号 (0x1 到 0x7)。如果位 6=1，则函数支持 CSA，主机通过写入位 7=1 启用访问。下一步是主机加载 24 位地址以开始读取或写入。这是通过将 24 位 (A23-0) 写入寄存器 0x00n0C 到 0x00n0E 来实现的，其中 n 是功能号 (0x1 到 0x7)。写入起始地址后，可以通过访问寄存器 0x00n0F (CSA 数据窗口寄存器) 读取或写入数据。如果需要读取或写入超过 1 个字节，则可以使用 0 (固定地址) 的操作码执行扩展 I/O 命令 (字节或块)。每次访问窗口寄存器时，地址指针应自动递增，以便访问 CSA 内的顺序地址。一旦操作完成，下一个操作的地址应保存在 24 位地址寄存器中，以便主机读取。

### 24.3.6.2 CSA 数据格式

存储在 CSA 中的数据应使用 FAT12/FAT16 格式进行结构化。使用 CSA 存储不同主机类型的程序或数据需要 SDIO 卡制造商以主机可以识别的文件格式加载程序和数据。这方面的一个例子是使用保存在特定子目录中的特定文件名，该文件名由特定主机操作系统识别和执行。这些格式是特定的，有时是不同主机实现和操作系统的专有格式。

## 24.3.7 SDIO 中断

为了允许 SDIO 卡中断主机，在 SD 接口的引脚上添加了中断功能。在 4 位 SD 模式下操作时用作 DAT[1] 的引脚 8 用于向主机发送卡中断信号。对于卡中的每个卡或功能，中断的使用是可选的。SDIO 中断是“电平敏感”的，也就是说，中断线应保持激活（低），直到主机识别并对其进行操作，或者由于中断周期结束而取消断言。一旦主机为中断提供服务，它将通过一些功能独特的 I/O 操作被清除。所有主机应在数据线 DAT[3:0] 上提供上拉电阻。

### 24.3.7.1 SPI 和 SD 1 位模式中断

在 SPI 和 1 位 SD 模式下，引脚 8 专用于中断功能。因此，在 SPI 和 SD 1 位模式中，中断没有时间限制。SPI 或 1 位 SD 模式下的卡通过断言引脚 8 为低，随时向主机发送中断信号。主机使用电平敏感输入检测此挂起中断。主机负责清除中断。如果 SDIO 卡在 SPI 模式下运行，如果未选择该卡，则可能不会断言该卡的中断。（CS=0）。只有当卡在未选择时（CCCR 中的 SCSI 位 =1）能够中断，并且该功能已打开（ECSI 位 =1）时，才会出现此要求的例外情况。在这种情况下，无论 CS 线路的状态如何，卡都可以断言中断。

### 24.3.7.2 SD 4 位模式中断

由于引脚 8 在 4 位 SD 模式下的 IRQ 和 DAT[1] 使用之间共享，因此中断只能由卡发送，并在特定时间内由主机识别。引脚 8 上的低电平被识别为中断的时间被定义为中断周期。在中断期间，SDIO 主机只能将引脚 8 (DAT[1]/IRQ) 上的电平采样到中断检测器中。在所有其他时间，主机中断控制器应忽略引脚 8 上的电平。注意，中断周期适用于内存和 I/O 操作。对于单块和多块数据传输的操作，中断周期的定义是不同的。

### 24.3.7.3 中断清除时间

由于 SDIO 卡使用电平敏感中断，主机应通过对某个功能唯一区域的 I/O 读取或写入来清除未决中断。在某些主机实现中，向卡发送 CMD52 由主机适配器硬件处理，而主机 CPU 可以执行其他操作。如果中断清除的时间不受控制，此情况可能允许已处理的中断重新中断主机。为防止出现这种情况，任何实现中断的 SDIO 卡在写入清除中断的功能唯一区后，应遵循从 DAT[1] 行中删除中断所需的时间。中断的清除可由功能唯一方法中的 I/O 写入或功能唯一 I/O 读取引起。使用 I/O 读取清除中断的一个例子是，数据寄存器的读取可自动清除数据就绪中断。

### 24.3.8 SDIO 挂起/恢复操作

用于在 SD 总线上执行挂起/恢复操作的步骤为：

- 主机确定当前使用 DAT[3:0] 行的功能。
- 主机请求较低优先级或较慢的事务挂起。
- 主机检查事务挂起是否完成。
- 主机开始更高优先级的事务。
- 主机等待高优先级事务的完成。
- 主机恢复挂起的事务。

如果当前事务可以接受挂起，并且卡在读取等待期间收到挂起命令，则应接受挂起请求。

### 24.3.9 SDIO 读取等待操作

可选的读取等待 (RW) 操作仅针对 SD 1 位和 4 位模式定义。读取等待操作允许主机向正在执行多次读取 (CMD53) 操作的卡发送信号，以临时暂停数据传输，同时允许主机向 SDIO 卡内的任何功能发送命令。为了确定卡是否支持读取等待协议，主机应测试 CCCR 卡容量字节中的 SRW 容量位。读取等待的计时基于中断周期。如果卡不支持读等待协议，则主机必须在读多命令的中间停顿（不中止）数据的唯一手段是控制 SDCLK。读等待支持对于支持挂起/恢复的卡是强制性的。

## 25.1 概述

低功耗是物联网应用的一项重要指标。芯片的处理器包含 3 种功耗模式，包含工作模式、空闲省电模式和休眠模式，可以根据当前应用场景选择合适的功耗模式，降低芯片功耗延长电池寿命。

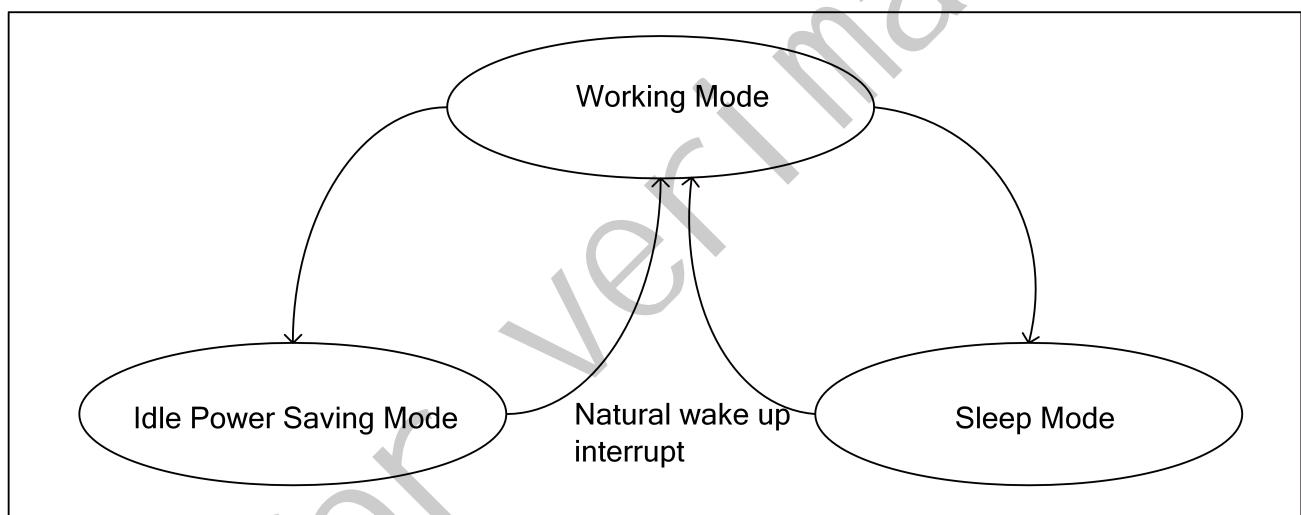


图 25.1: 低功耗模式

## 25.2 主要特征

- 时钟控制: GLB 中对各外设的时钟控制，小范围的省电，响应速度较快
- 睡眠控制 (PDS) : 包含 PDS1/2/3/7/15 这 5 个等级，大范围省电，响应速度中等
- 深度睡眠控制 (HBN) : 包含 HBN0/1/2/3 这 4 个等级，全局省电，响应时间长

## 25.3 功能描述

### 25.3.1 电源域

BL616/BL618 芯片中有 8 个电源域，每个电源域的主要功能如下所示：

- PD\_AON
  - HBN 状态机，可控制 PD\_AON\_HBNRTC/PD\_AON\_HBNCORE/PD\_CORE 的电源/隔离单元/复位/时钟
  - AON\_PIN (GPIO16/17/18/19) 引脚唤醒功能
  - 预留 4 个可读可写的寄存器，可在 PDS/HBN0/HBN1/HBN2 模式下保存数据
  - BOR (Brown Out Reset) 设置功能
  - LDO11\_AON/RT/SOC 输出电压选择单元
  - Root、F32K、Uart 时钟源选择
  - HBN\_OUT0\_PIR (Acomp0/Acomp1/bor/pir) 唤醒屏蔽和使能寄存器、中断状态寄存器
- PD\_AON\_HBNRTC
  - 选择 RTC Counter 时钟源的控制单元
  - RTC 可以用于唤醒，也可用于 LED 闪烁
  - RC32K 和 Xtal32X 控制功能
- PD\_AON\_HBNCORE
  - 部分电源控制寄存器
  - 保留 HBN\_RAM，可用于保存程序和数据，进入 PDS/HBN0 后数据不会消失
  - 在 HBN 模式下，具有控制 AON\_PIN 和保持其他 IO 的功能
  - PIR 数字控制，PIR 是热释电红外传感器，HBN 区域内的一个外设，可以用作 HBN 唤醒源
  - 保留 LDO11SOC、LDO15RF、DCDC0、DCDC1、DCDC2 控制寄存器
  - 保留 XTAL、TSEN、Acomp0/1、GPADC 控制寄存器
  - 预留 4 个可读可写的寄存器，可在 PDS/HBN0 模式下保存数据
- PD\_CORE
  - PDS 状态机控制 PD\_CORE\_MISC/PD\_USB/PD\_CPU/PD\_WB 的电源、隔离单元、复位、时钟和 Memory
  - 保留 PDS\_RAM，进入 PDS 后数据不会消失
  - WIFI/BLE 计时器控制

- 160KB WRAM Retention/Sleep
- 控制 PDS 中断和唤醒功能
- 保留 GPIO0~15/20~36 的控制
- 保留 RC32M 的控制
- PDS\_Timer 定时器
- PD\_CORE\_MISC
  - PD\_CORE\_MISC 是 PD\_CORE\_MISC\_DIG 和 PD\_CORE\_MISC\_ANA 的统称
  - 外设（包括 I2C/SDIO/UART/FLASH/SPI/ADC/DAC/GPIO/PWM 等）
  - 芯片 GLB 寄存器
- PD\_USB
  - USB 控制器
- PD\_CPU
  - NP\_CPU 及缓存单元
  - ROM、TCM
- PD\_WB
  - WIFI PHY/MAC
  - BLE PHY/MAC
  - RF Controller

每个电源域都由 8 个不同的电源模式控制，具体的控制方式如下表所示：

表 25.1: 电源模式

NO.	Scenario	Power Domain							
		PD_AON	PD_AON_- HBNRTC	PD_- AON_- HBNCORE	PD_- CORE	PD_CORE_- MISC	PD_USB	PD_CPU	PD_WB
1	Normal	ON	ON	ON	ON	ON	ON	ON	ON
2	PDS1	ON	ON	ON	ON	ON	ON	ON	OFF
3	PDS2	ON	ON	ON	ON	ON	ON	OFF	ON
4	PDS3	ON	ON	ON	ON	ON	ON	OFF	OFF
5	PDS7	ON	ON	ON	ON	ON	OFF	OFF	OFF
6	PDS15	ON	ON	ON	ON	OFF	OFF	OFF	OFF
7	HBN0	ON	ON	ON	OFF	OFF	OFF	OFF	OFF

表 25.1: 电源模式 (continued)

NO.	Scenario	Power Domain							
		PD_AON	PD_AON_- HBNRTC	PD_- AON_- HBNCORE	PD_- CORE	PD_CORE_- MISC	PD_USB	PD_CPU	PD_WB
8	HBN1	ON	ON	OFF	OFF	OFF	OFF	OFF	OFF
9	HBN2	ON	OFF	OFF	OFF	OFF	OFF	OFF	OFF
10	HBN3	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF

### 25.3.2 唤醒源

芯片内部支持多种唤醒源，可从不同电源模式中唤醒。

不同电源模式的唤醒源如下表所示：

表 25.2: 唤醒源

电源模式	唤醒源
PDS0	AON_PIN/BOR/RTC/Pir/Acomp0/Acomp1/PDS_Timer/GPIO/IRRX/DM/USB/WIFI
PDS1	AON_PIN/BOR/RTC/Pir/Acomp0/Acomp1/PDS_Timer/GPIO/IRRX/DM/USB
PDS2	AON_PIN/BOR/RTC/Pir/Acomp0/Acomp1/PDS_Timer/GPIO/IRRX/DM/USB/WIFI
PDS3	AON_PIN/BOR/RTC/Pir/Acomp0/Acomp1/PDS_Timer/GPIO/IRRX/DM/USB
PDS7	AON_PIN/BOR/RTC/Pir/Acomp0/Acomp1/PDS_Timer/GPIO/IRRX/DM
PDS15	AON_PIN/BOR/RTC/Pir/Acomp0/Acomp1/PDS_Timer
HBN0	AON_PIN/BOR/RTC/Pir/Acomp0/Acomp1
HBN1	AON_PIN/RTC
HBN2	AON_PIN
HBN3	重新给 VDDIO2 供电

### 25.3.3 功耗模式

工作模式

芯片提供处理器与外设独立的时钟控制，在 GLB 和时钟的章节介绍对各模块的时钟控制，软件可以根据当前应用场景，对于不需要使用的处理器或外设进行时钟控制。时钟控制的响应是实时的，在此工作模式下，不需要担心响应时间。

#### 掉电睡眠模式 (PDS)

掉电模式相较于工作模式功耗较低。进入 PDS 模式后，将 RTC(Real Time Clock) 之外的时钟进行管控，会切换为内

部低速时钟，并将外部晶振与 PLL 关闭达到更加省电的状态，因此进入与离开此低功耗模式会有时间延迟。当进入掉电睡眠模式时，OCRAM 区域的数据可以自动进入 retention 状态而保留下来，当唤醒后可以自行退出 retention 状态。

## 1. 进入空闲省电模式

软件可通过 PDS 配置让此模块进入掉电模式，等待处理，进入等待中断模式 (WFI) 后，PDS 模块会触发时钟控制模块进入 gate clock 操作，并通知模拟电路关闭 PLL 以及外部晶振

## 2. 离开空闲省电模式

离开空闲省电模式的方式有两种，第一是空闲中间有特定的中断或事件打断空闲状态，第二是软件设定 PDS\_TIMER 的空闲时间达到，两者均会触发 PDS 模块进入或离开掉电模式。注意：因为打开晶振需要约 1ms 的时间，PDS 提供软件提前打开晶振的方式，这个做法可以加速 PDS 醒来，当 PDS 模块准备醒来时，此模块会通过中断通知处理器离开等待中断模式 (WFI)。

## 休眠模式 (HBN)

休眠模式在保留 AON(Always On) 电源的状态下，将大部分的芯片逻辑进行断电 (Vcore)，直到收到外部事件才会将内部电路唤醒的。在休眠模式下可以达到极致的省电状态，但相对于前两者需要的响应时间也最长，适合长时间不需要工作的状态下，可以进入此状态，延长电池寿命。休眠时期会将大部分的电路断电，对应的寄存器值和内存的数据也会消失。因此 HBN 内部留有 4KB HBN\_RAM，这个内存在休眠状态时不会断电，软件有需要保存的资料或状态可以在进入休眠前拷贝到这个内存。从休眠恢复时，可以直接从 RAM 中存取数据，通常可以用作状态的纪录或是数据快速恢复。

### 25.3.4 IO 保持

IO 保持可以分为 AON\_IO 保持和 PDS\_IO 保持。PDS1/2/3/7 模式下，由于芯片 MISC domain 仍有电，所以 GPIO 可被 glb 寄存器控制。当 glb 寄存器被断电之后，可由 AON\_CTRL 和 PDS 简单控制 AON\_IO 和 PDS\_IO 的 IE/PD/PU。

#### AON\_IO

AON\_IO 指 GPIO16/17/18/19。GPIO16/17 可以作为 XTAL32K 输入和输出使用。

当 reg\_en\_aon\_ctrl\_gpio 为 1，reg\_en\_aon\_ctrl\_gpio[3:0] 控制 GPIO16/17/18/19 是否被 AON\_HW 控制。当 reg\_en\_aon\_ctrl\_gpio 为 1 时，AON\_IO 的上拉使能由 reg\_aon\_gpio\_pu 控制，下拉使能由 reg\_aon\_gpio\_pd 控制，IE/SMT 由 reg\_aon\_gpio\_ie\_smt 控制，OE 由 reg\_aon\_gpio\_oe 控制。

1、硬件 IO 保持 HBN 可以控制 AON\_IO 的 IE/PD/PU/OE/O，从而实现 IO 保持。当 reg\_en\_aon\_ctrl\_gpio 为 1 时，AON\_IO 的上拉使能由 reg\_aon\_pad\_pu 控制，下拉使能由 reg\_aon\_pad\_pd 控制，OE 由 reg\_aon\_pad\_oe 控制，IE/SMT 由 reg\_aon\_pad\_ie\_smt 控制。例如，reg\_en\_aon\_ctrl\_gpio 为 0，reg\_aon\_pad\_pu 为 1，也不能实现上拉；当 reg\_aon\_gpio\_ie\_smt 为 1，并且 reg\_aon\_pad\_pu 为 1，可以实现上拉功能。

2、软件 IO 保持将 reg\_aon\_gpio\_iso\_mode 配置为 1 后，进入 HBN 模式时，AON PAD 可以保持 OEO，PUPD 不能保持；等到 HBN 唤醒后，AON PAD 状态仍会保持着，需要把 reg\_aon\_gpio\_iso\_mode 请 0 后，才会离开 IO 保持状态。例如，GPIO16 在 HBN 模式下保持高电平，需要先将其配为普通 IO 功能，然后用 glb 寄存器配置为输出高电平，最后将 reg\_aon\_gpio\_iso\_mode 配置为 1 后，进入 HBN 模式。

#### PDS\_IO

PDS\_IO 指除 AON\_IO 以外的其他 GPIO，共 30 个 GPIO，分为 2 组：

- GPIO0~15
- GPIO20~33

注意，GPIO16~19 可以被 PDS 控制实现软件 IO 保持，但是不能实现硬件 IO 保持。

1、硬件 IO 保持 PDS\_IO 的 IE/PD/PU 可由 `pds_gpio_i_set` 寄存器控制，相同的组 GPIO 必须保持相同的电平。例如，GPIO0 是配置为上拉，那么 GPIO8 也是配置为上拉的。

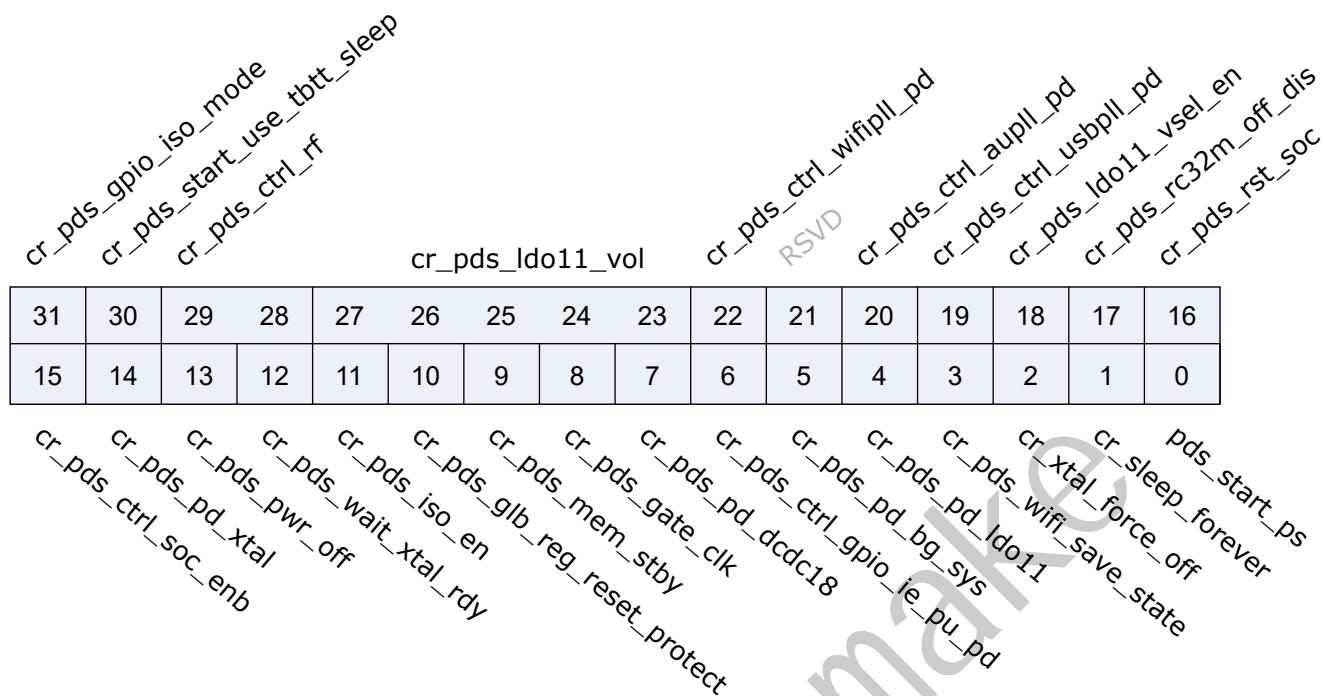
2、软件 IO 保持当 `cr_pds_gpio_iso_mode` 为 1 时，进入 PDS7 模式后，如果 `cr_pds_gpio_kee_en[0]、[1]、[2]` 为 1，GPIO0~15、GPIO20~33(不含 GPIO21/22/28/29)、GPIO16~19 分别进入 GPIO 保持状态，等到 PDS 唤醒后，PDS\_IO 状态仍会保持着，需要把 `cr_pds_gpio_iso_mode` 请 0 后，才会离开 IO 保持状态。这种 IO 保持方式的优势是可以实现同一组 GPIO 保持不同的电平。

## 25.4 寄存器描述

名称	描述
PDS_CTL	
PDS_TIME1	
PDS_INT	
PDS_CTL2	
PDS_CTL3	
PDS_CTL4	
pds_stat	
pds_ram1	
PDS_CTL5	
PDS_RAM2	
pds_gpio_i_set	
pds_gpio_pd_set	
pds_gpio_int	
pds_gpio_stat	
PDS_RAM3	
PDS_RAM4	

### 25.4.1 PDS\_CTL

地址: 0x2000e000



位	名称	权限	复位值	描述
31	cr_pds_gpio_iso_mode	r/w	0	1: HW Keep GPIO @ PDS7 0 : Clear this bit after PDS7 to release GPIO
30	RSVD			
29:28	cr_pds_ctrl_rf	r/w	2'b01	00 : PDS don't control RF on/off 01 : PDS control RF on/off depend on misc_pwr_off 10 : PDS control RF on/off depend on any power off 11 : PDS control RF on/off whe pds at idle state
27:23	cr_pds_ldo11_vol	r/w	5'h8	LDO11 voltage value in PDS mode output voltage sel: 0: 0.9V, 4: 1.0V, 8: 1.1V, 12: 1.2V, 25mV/step
22	cr_pds_ctrl_wifipll_pd	r/w	0	PDS Control WIFI PLL off When pds_pwr_off
21	RSVD			
20	cr_pds_ctrl_aupll_pd	r/w	0	PDS Control Audio PLL off When pds_pwr_off
19	cr_pds_ctrl_usbppl_pd	r/w	0	PDS Control USB PLL off When pds_pwr_off
18	cr_pds_ldo11_vsel_en	r/w	0	PDS "SLEEP" control LDO11 voltage enable
17	cr_pds_rc32m_off_dis	r/w	0	1 : RC32M always on @any state 0 : RC32M on/off controlled by PDS state

位	名称	权限	复位值	描述
16	cr_pds_RST_soc	r/w	0	0 : no pds_reset 1: pds_RST controlled by PDS
15	cr_pds_CTRL_soc_enb	r/w	0	1 : pds_soc_enb controlled by PDS 0 : pds_soc_enb always active
14	cr_pds_PD_xtal	r/w	1	0 : don't_touch xtal during PDS 1 : xtal power down during PDS
13	cr_pds_PWR_off	r/w	1	0 : don't_touch Power during PDS 1 : Power off during PDS (each power domain can has its own control)
12	cr_pds_WAIT_xtal_rdy	r/w	0	0 : Skip wait XTAL Ready before PDS Interrupt 1 : wait XTAL Ready during before PDS Interrupt
11	cr_pds_ISO_en	r/w	1	0 : don't_touch Isolation during PDS (all power domain) 1 : Isolation during PDS (each power domain can has its own control)
10	cr_pds_glb_Reg_RESET_protect	r/w	0	1: avoid glb_Reg reset by any reset
9	cr_pds_MEM_stby	r/w	1	0 : don't_touch mem_stby during PDS 1 : mem_stby during PDS (each power domain can has its own control)
8	cr_pds_GATE_clk	r/w	1	0 : don't_touch clock gating during PDS (all power domain) 1 : gate clock during PDS (each pwr domain has its own control)
7	cr_pds_PD_dcdc18	r/w	0	0 : don't_touch dcdc18 during PDS 1 : power down dcdc18 during PDS
6	cr_pds_CTRL_gpio_ie_pu_pd	r/w	0	1: allow PDS Control the GPIO IE/PU/PD at Sleep Mode
5	cr_pds_PD_bg_sys	r/w	0	0 : don't_touch bg_sys during PDS 1 : power down bg_sys during PDS
4	cr_pds_PD_Ido11	r/w	0	0 : don't_touch ldo11 during PDS 1 : power down ldo11 during PDS
3	cr_pds_WIFI_save_state	r/w	0	Save WIFI State Before Enter PDS
2	cr_xtal_FORCE_off	r/w	0	
1	cr_SLEEP_forever	r/w	0	
0	pds_start_ps	w1p	0	Enter PDS

## 25.4.2 PDS\_TIME1

地址: 0x2000e004

cr\_sleep\_duration

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\_sleep\_duration

位	名称	权限	复位值	描述
31:0	cr_sleep_duration	r/w	32'd3240	PDS Sleep Time (in units of 32K clock cycles)

## 25.4.3 PDS\_INT

地址: 0x2000e00c

ro\_pds\_wakeup\_event

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\_pds\_wakeup\_src\_en

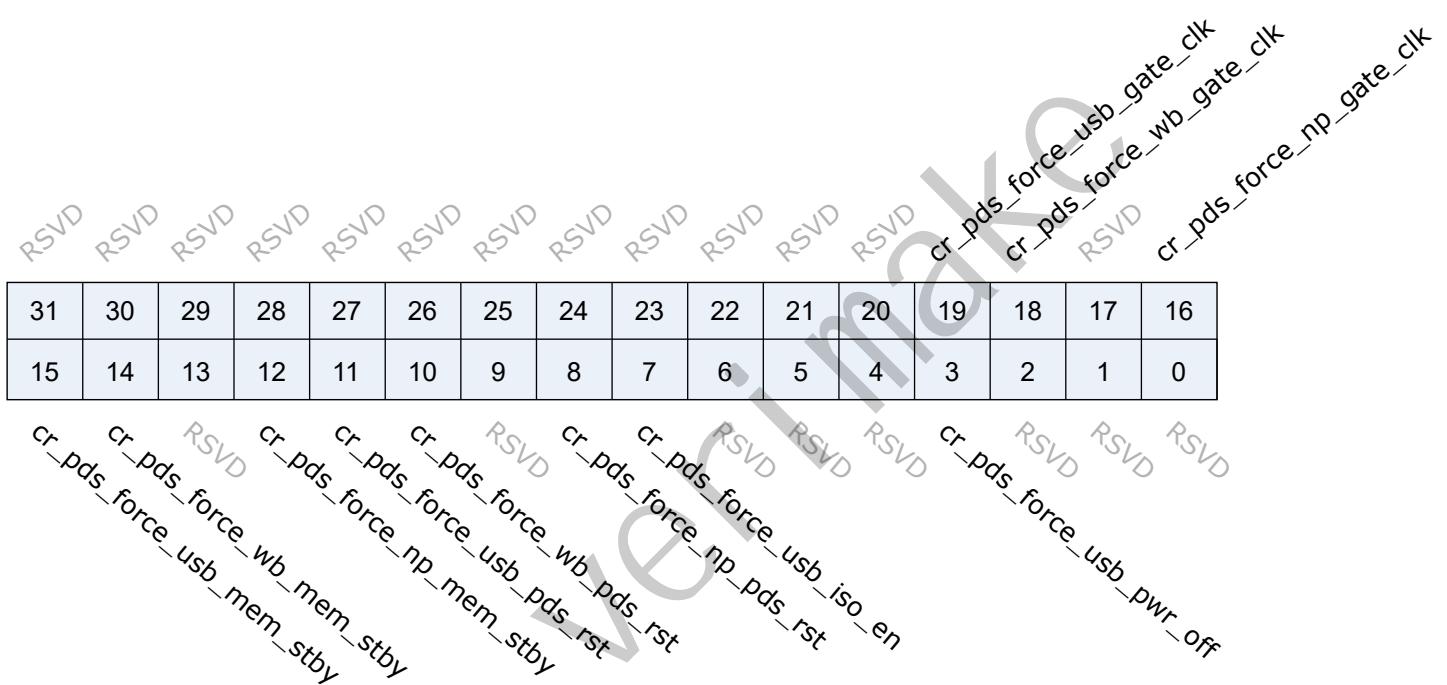
ro\_pds\_rf\_done\_int  
ro\_pds\_wifi\_tbtt\_sleep\_irq  
ro\_pds\_wifi\_tbtt\_wakeup\_irq  
ro\_pds\_wake\_int  
ro\_pds\_rf\_done\_int  
ro\_pds\_rf\_done\_int\_mask  
ro\_pds\_wifi\_tbtt\_sleep\_irq\_mask  
ro\_pds\_wifi\_tbtt\_wakeup\_irq\_mask  
cr\_pds\_int\_clr

位	名称	权限	复位值	描述
31:9	RSVD			
8	cr_pds_int_clr	r/w	0	pds interrupt clear
7:6	RSVD			
5	cr_pds_rf_done_int_mask	r/w	0	Mask pds rf done interrupt
4	cr_pds_wake_int_mask	r/w	0	Mask pds wakeup interrupt

位	名称	权限	复位值	描述
3:2	RSVD			
1	ro_pds_rf_done_int	r	0	pu_rf_done interrupt
0	ro_pds_wake_int	r	0	PDS Wakeup Interrupt

#### 25.4.4 PDS\_CTL2

地址: 0x2000e010

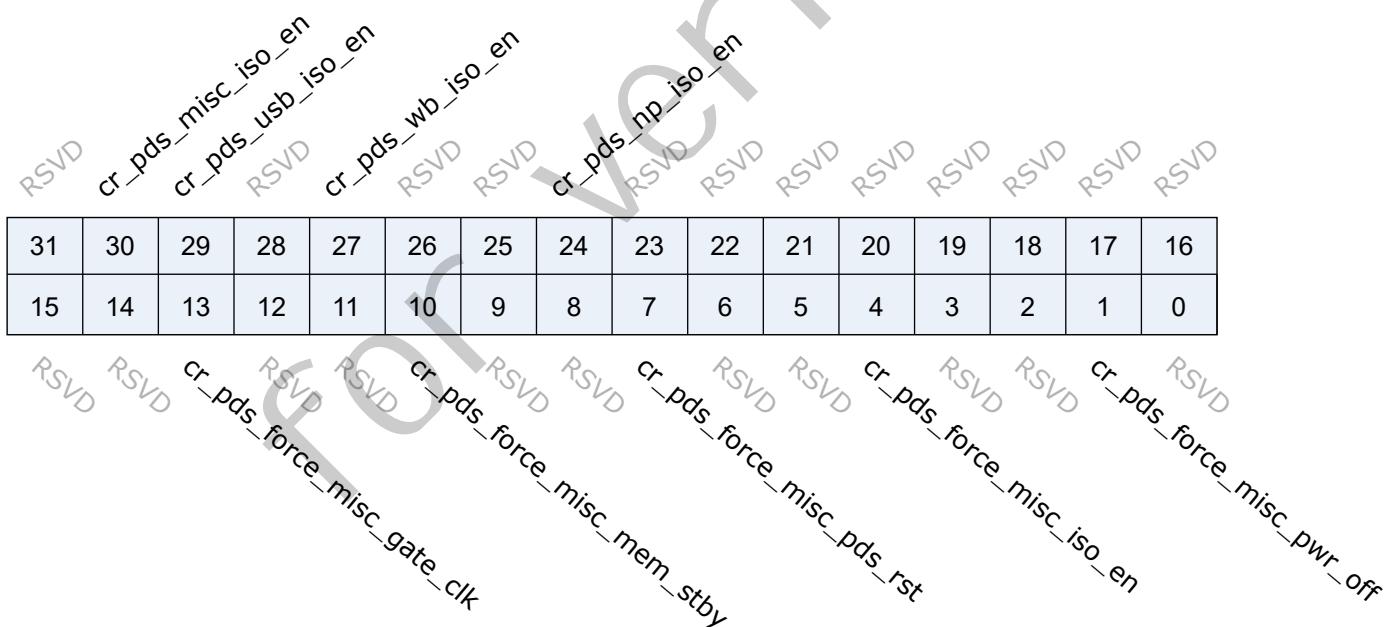


位	名称	权限	复位值	描述
31:20	RSVD			
19	cr_pds_force_usb_gate_clk	r/w	0	manual force usbio clock gated
18	cr_pds_force_wb_gate_clk	r/w	0	manual force WB clock gated
17	RSVD			
16	cr_pds_force_np_gate_clk	r/w	0	manual force NP clock gated
15	cr_pds_force_usb_mem_stby	r/w	0	manual force usbio memory sleep
14	cr_pds_force_wb_mem_stby	r/w	0	manual force WB memory sleep
13	RSVD			
12	cr_pds_force_np_mem_stby	r/w	0	manual force NP memory sleep
11	cr_pds_force_usb_pds_rst	r/w	0	manual force usbio pds reset

位	名称	权限	复位值	描述
10	cr_pds_force_wb_pds_RST	r/w	0	manual force WB pds reset
9	RSVD			
8	cr_pds_force_np_pds_RST	r/w	0	manual force NP pds reset
7	cr_pds_force_usb_iso_en	r/w	0	manual force usbio isolation
6	cr_pds_force_wb_iso_en	r/w	0	manual force WB isolation
5	RSVD			
4	cr_pds_force_np_iso_en	r/w	0	manual force NP isolation
3	cr_pds_force_usb_pwr_off	r/w	0	manual force usbio power off
2	cr_pds_force_wb_pwr_off	r/w	0	manual force WB power off
1	RSVD			
0	cr_pds_force_np_pwr_off	r/w	0	manual force NP power off

## 25.4.5 PDS CTL3

地址: 0x2000e014



位	名称	权限	复位值	描述
31	RSVD			
30	cr_pds_misc_iso_en	r/w	1	1 : make misc isolated at PDS Sleep state 0 : make misc isolated at PDS Sleep state

位	名称	权限	复位值	描述
29	cr_pds_usb_iso_en	r/w	1	1 : make usb isolated at PDS Sleep state 0 : make usb isolated at PDS Sleep state
28	RSVD			
27	cr_pds_wb_iso_en	r/w	1	1 : make WB isolated at PDS Sleep state 0 : make WB isolated at PDS Sleep state
26:25	RSVD			
24	cr_pds_np_iso_en	r/w	1	1 : make NP isolated at PDS Sleep state 0 : make NP isolated at PDS Sleep state
23:14	RSVD			
13	cr_pds_force_misc_gate_clk	r/w	0	manual force MISC gate_clk
12:11	RSVD			
10	cr_pds_force_misc_mem_stby	r/w	0	manual force MISC mem_stby
9:8	RSVD			
7	cr_pds_force_misc_pds_RST	r/w	0	manual force MISC pds_RST
6:5	RSVD			
4	cr_pds_force_misc_iso_en	r/w	0	manual force MISC iso_en
3:2	RSVD			
1	cr_pds_force_misc_pwr_off	r/w	0	manual force MISC pwr_off
0	RSVD			

#### 25.4.6 PDS\_CTL4

地址: 0x2000e018

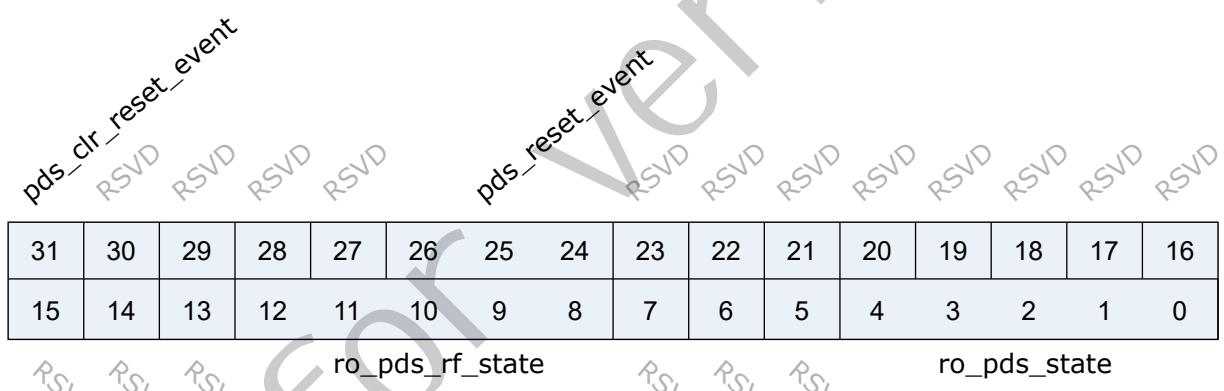
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cr_pds_misc_gate_clk	RSVD	cr_pds_usb_gate_clk	RSVD	cr_pds_usb_gate_clk	RSVD	cr_pds_usb_gate_clk	RSVD								
cr_pds_misc_mem_stby	RSVD	cr_pds_usb_mem_stby	RSVD	cr_pds_usb_mem_stby	RSVD	cr_pds_usb_mem_stby	RSVD								
cr_pds_misc_reset	RSVD	cr_pds_usb_reset	RSVD	cr_pds_usb_reset	RSVD	cr_pds_usb_reset	RSVD								
cr_pds_pwr_off	RSVD	cr_pds_usb_pwr_off	RSVD	cr_pds_usb_pwr_off	RSVD	cr_pds_usb_pwr_off	RSVD								
cr_pds_np_pwr_off	RSVD	cr_pds_np_pwr_off	RSVD	cr_pds_np_pwr_off	RSVD	cr_pds_np_pwr_off	RSVD								
cr_pds_np_gate_clk	RSVD	cr_pds_np_gate_clk	RSVD	cr_pds_np_gate_clk	RSVD	cr_pds_np_gate_clk	RSVD								
cr_pds_np_mem_stby	RSVD	cr_pds_np_mem_stby	RSVD	cr_pds_np_mem_stby	RSVD	cr_pds_np_mem_stby	RSVD								
cr_pds_wb_gate_clk	RSVD	cr_pds_wb_gate_clk	RSVD	cr_pds_wb_gate_clk	RSVD	cr_pds_wb_gate_clk	RSVD								
cr_pds_wb_mem_stby	RSVD	cr_pds_wb_mem_stby	RSVD	cr_pds_wb_mem_stby	RSVD	cr_pds_wb_mem_stby	RSVD								

位	名称	权限	复位值	描述
31:28	RSVD			
27	cr_pds_misc_gate_clk	r/w	1	1 : make core_misc clock gated at PDS Sleep state 0 : make core_misc clocking at PDS Sleep state
26	cr_pds_misc_mem_stby	r/w	1	1 : make core_misc RAM @Retention at PDS Sleep state 0 : make core_misc RAM @ Normal at PDS Sleep state
25	cr_pds_misc_reset	r/w	1	1 : make core_misc reset at PDS Sleep state 0 : make core_misc not reset at PDS Sleep state
24	cr_pds_misc_pwr_off	r/w	1	1 : make core_misc Power off at PDS Sleep state 0 : make core_misc power on at PDS Sleep state
23	cr_pds_usb_gate_clk	r/w	1	1 : make usb clock gated at PDS Sleep state 0 : make usb clocking at PDS Sleep state
22	cr_pds_usb_mem_stby	r/w	1	1 : make usb RAM @Retention at PDS Sleep state 0 : make usb RAM @ Normal at PDS Sleep state
21	cr_pds_usb_reset	r/w	1	1 : make usb reset at PDS Sleep state 0 : make usb not reset at PDS Sleep state
20	cr_pds_usb_pwr_off	r/w	1	1 : make usb Power off at PDS Sleep state 0 : make usb power on at PDS Sleep state
19:16	RSVD			
15	cr_pds_wb_gate_clk	r/w	1	1 : make WB clock gated at PDS Sleep state 0 : make WB clocking at PDS Sleep state
14	cr_pds_wb_mem_stby	r/w	1	1 : make WB RAM @Retention at PDS Sleep state 0 : make WB RAM @ Normal at PDS Sleep state

位	名称	权限	复位值	描述
13	cr_pds_wb_reset	r/w	1	1 : make WB reset at PDS Sleep state 0 : make WB not reset at PDS Sleep state
12	cr_pds_wb_pwr_off	r/w	1	1 : make WB Power off at PDS Sleep state 0 : make WB power on at PDS Sleep state
11:4	RSVD			
3	cr_pds_np_gate_clk	r/w	1	1 : make NP clock gated at PDS Sleep state 0 : make NP clocking at PDS Sleep state
2	cr_pds_np_mem_stby	r/w	1	1 : make NP RAM @Retention at PDS Sleep state 0 : make NP RAM @ Normal at PDS Sleep state
1	cr_pds_np_reset	r/w	1	1 : make NP reset at PDS Sleep state 0 : make NP not reset at PDS Sleep state
0	cr_pds_np_pwr_off	r/w	1	1 : make NP Power off at PDS Sleep state 0 : make NP power on at PDS Sleep state

#### 25.4.7 pds\_stat

地址: 0x2000e01c



位	名称	权限	复位值	描述
31	pds_clr_reset_event	w1c	0	clear pds reset event
30:27	RSVD			
26:24	pds_reset_event	r	0	[2] : pds_rst_n (pds reset) [1]: pwr_rst_n (hbn power on reset) [0]: hreset_n (Bus Reset)
23:13	RSVD			

位	名称	权限	复位值	描述
12:8	ro_pds_rf_state	r	5'b0	ST_PDS_RF_OFF = 5'b0000 ; ST_PDS_PU_MBG = 5'b0001 ; ST_PDS_PU_LDO15RF = 5'b0011 ; ST_PDS_PU_SFREG = 5'b0111 ; ST_PDS_PUD_XTAL18 = 5'b01111; ST_PDS_WB_EN_AON = 5'b11111 ;
7:5	RSVD			
4:0	ro_pds_state	r	5'b0	ST_IDLE = 5'b00000; ST_MEM_STBY = 5'b10000; ST_ECG = 5'b01000; ST_ERST = 5'b01100; ST_EISO = 5'b01111; ST_POFF = 5'b00111; ST_PRE_BGON = 5'b00011; ST_PRE_BGON1 = 5'b00001; ST_BGON = 5'b00101; ST_CLK_SW_32M= 5'b00100; ST_PON_DCDC = 5'b00110; ST_PON_LDO11_MISC = 5'b01110; ST_PON = 5'b01010; ST_DISO = 5'b00010; ST_DCG = 5'b01101; ST_MEM_IDLE = 5'b11000; ST_DRST = 5'b01011; ST_WAIT_EFUSE= 5'b01001;

#### 25.4.8 pds\_ram1

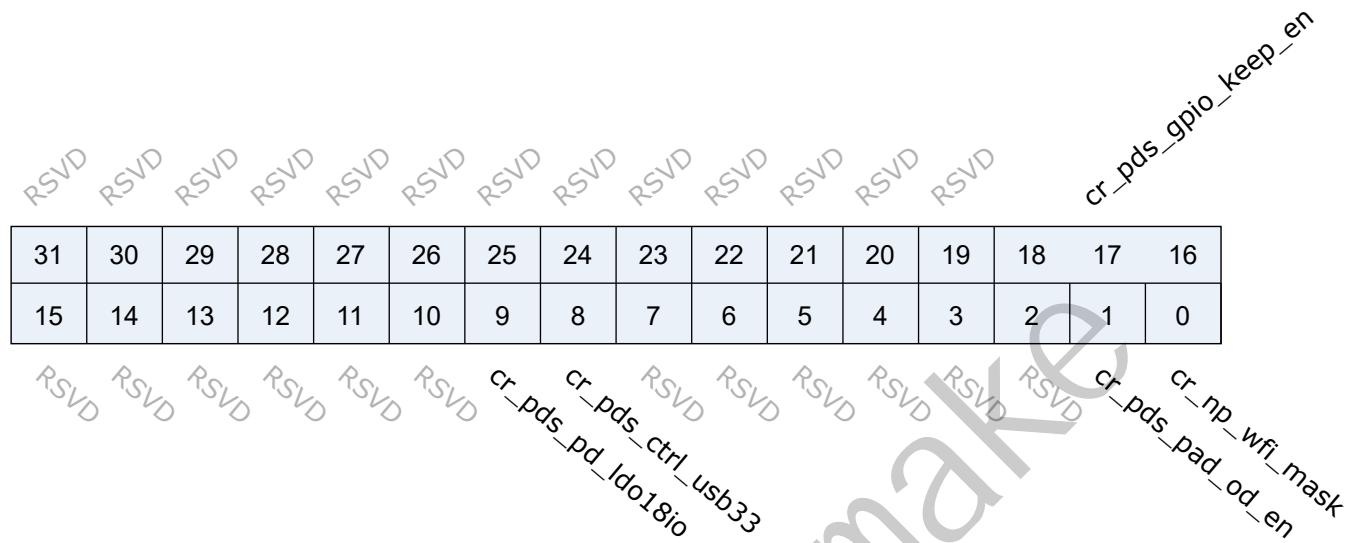
地址: 0x2000e020

cr_pds_ram_clk2_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
cr_pds_ram_clk_cnt															
RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	RSVD

位	名称	权限	复位值	描述
31	cr_pds_ctrl_ram_clk	r/w	1'b0	1 : Enable PDS Control PD_CORE SRAM Clock @ PDS Sequence
30	cr_pds_ctrl_ram_clk2	r/w	1'b0	HW Option To assert extra clock during PDS on sequence
29	RSVD			
28	cr_pds_ctrl_misc_ram_clk	r/w	1'b0	This bit is Enable by bit [31] : cr_pds_ctrl_ram_clk 1 : PDS Control PD_CORE_MISC SRAM Clock @ PDS Sequence 0 : PDS do nothing on SRAM Clock
27	cr_pds_ctrl_usb_ram_clk	r/w	1'b0	This bit is Enable by bit [31] : cr_pds_ctrl_ram_clk 1 : PDS Control PD_usb SRAM Clock @ PDS Sequence 0 : PDS do nothing on PD_usb SRAM Clock
26	cr_pds_ctrl_wb_ram_clk	r/w	1'b0	This bit is Enable by bit [31] : cr_pds_ctrl_ram_clk 1 : PDS Control PD_WB SRAM Clock @ PDS Sequence 0 : PDS do nothing on PD_WB SRAM Clock
25	RSVD			
24	cr_pds_ctrl_np_ram_clk	r/w	1'b0	This bit is Enable by bit [31] : cr_pds_ctrl_ram_clk 1 : PDS Control PD_CORE_CPU SRAM Clock @ PDS Sequence 0 : PDS do nothing on PD_CORE_CPU SRAM Clock
23:22	RSVD			
21:16	cr_pds_ram_clk2_cnt	r/w	6'd24	HW Option : Assert Extra Clock Counter in MEM_IDLE
15:14	RSVD			
13:8	cr_pds_ram_clk_cnt	r/w	6'd8	HW Option : Assert Extra Clock Counter in MEM_STBY
7:0	RSVD			

### 25.4.9 PDS\_CTL5

地址: 0x2000e024



位	名称	权限	复位值	描述
31:19	RSVD			
18:16	cr_pds_gpio_keep_en	r/w	3'b111	if cr_pds_gpio_iso_mode=1, can use bit to enable or disable keep function [0] : GPIO0 15 [1] : GPIO20 36 (not include GPIO21/22/28/29) [2] : GPIO16 19
15:10	RSVD			
9	cr_pds_pd_ldo18io	r/w	0	0 : don't_touch ldo18io during PDS 1 : power down ldo18io during PDS
8	cr_pds_ctrl_usb33	r/w	0	Set this bit to enable HW control turn on/off USB 3.3V @USB1.1V Power On/OFF (Replace the function of reg_pu_usb20_psw)
7:2	RSVD			
1	cr_pds_pad_od_en	r/w	0	GPIO21/22/28/29 5V Tolerant PAD Open Drain Enable
0	cr_np_wfi_mask	r/w	0	pds start condition mask np_wfi



## 25.4.10 PDS RAM2

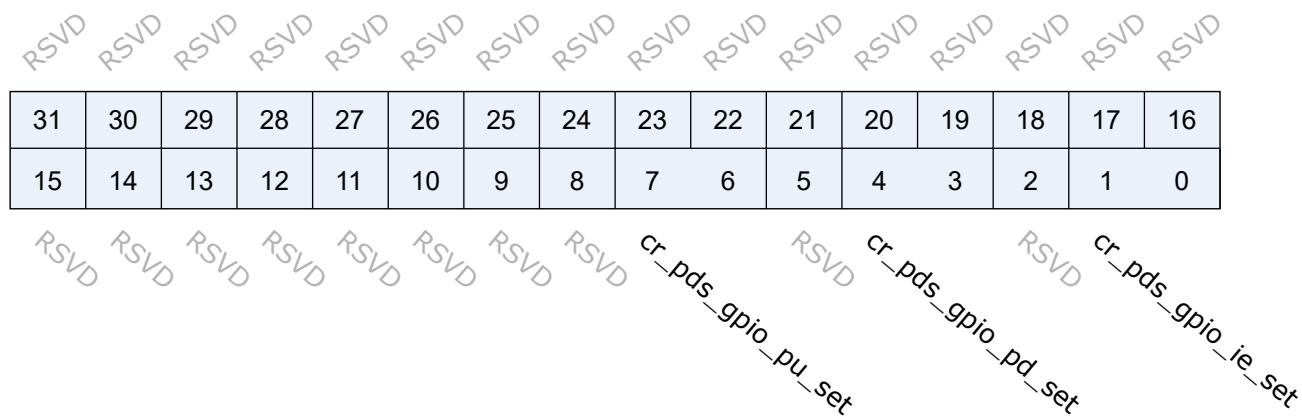
地址: 0x2000e028

RSVD	cr_wram_ret															
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	

位	名称	权限	复位值	描述
31:20	RSVD			
19:10	cr_wram_ret	r/w	10'h0	[9] : 144 160KB WRAM Retention [8] : 128 144KB WRAM Retention [7] : 112 128KB WRAM Retention [6] : 96 112KB WRAM Retention [5] : 80 96KB WRAM Retention [4] : 64 80KB WRAM Retention [3] : 48 64KB WRAM Retention [2] : 32 48KB WRAM Retention [1] : 16 32KB WRAM Retention [0] : 0 16KB WRAM Retention
9:0	cr_wram_slp	r/w	10'h0	[9] : 144 160KB WRAM SLEEP [8] : 128 144KB WRAM SLEEP [7] : 112 128KB WRAM SLEEP [6] : 96 112KB WRAM SLEEP [5] : 80 96KB WRAM SLEEP [4] : 64 80KB WRAM SLEEP [3] : 48 64KB WRAM SLEEP [2] : 32 48KB WRAM SLEEP [1] : 16 32KB WRAM SLEEP [0] : 0 16KB WRAM SLEEP

### 25.4.11 pds\_gpio\_i\_set

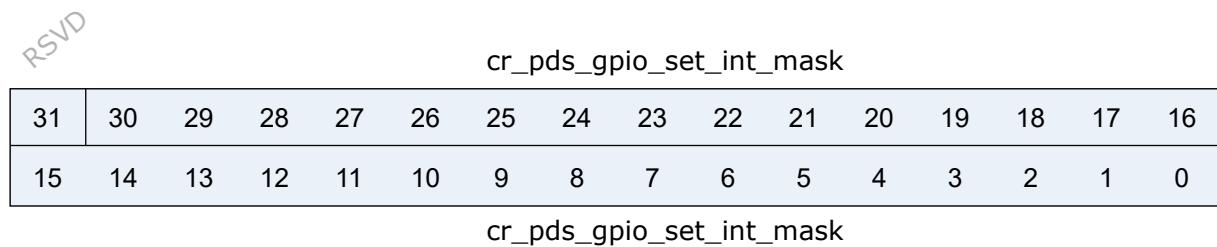
地址: 0x2000e030



位	名称	权限	复位值	描述
31:8	RSVD			
7:6	cr_pds_gpio_pu_set	r/w	2'b0	Enable GPIO PU @ PDS [0] : GPIO0 15 [1] : GPIO20 36
5	RSVD			
4:3	cr_pds_gpio_pd_set	r/w	2'b0	Enable GPIO PD @ PDS [0] : GPIO0 15 [1] : GPIO20 36
2	RSVD			
1:0	cr_pds_gpio_ie_set	r/w	2'b0	Enable GPIO IE @ PDS [0] : GPIO0 15 [1] : GPIO20 36

### 25.4.12 pds\_gpio\_pd\_set

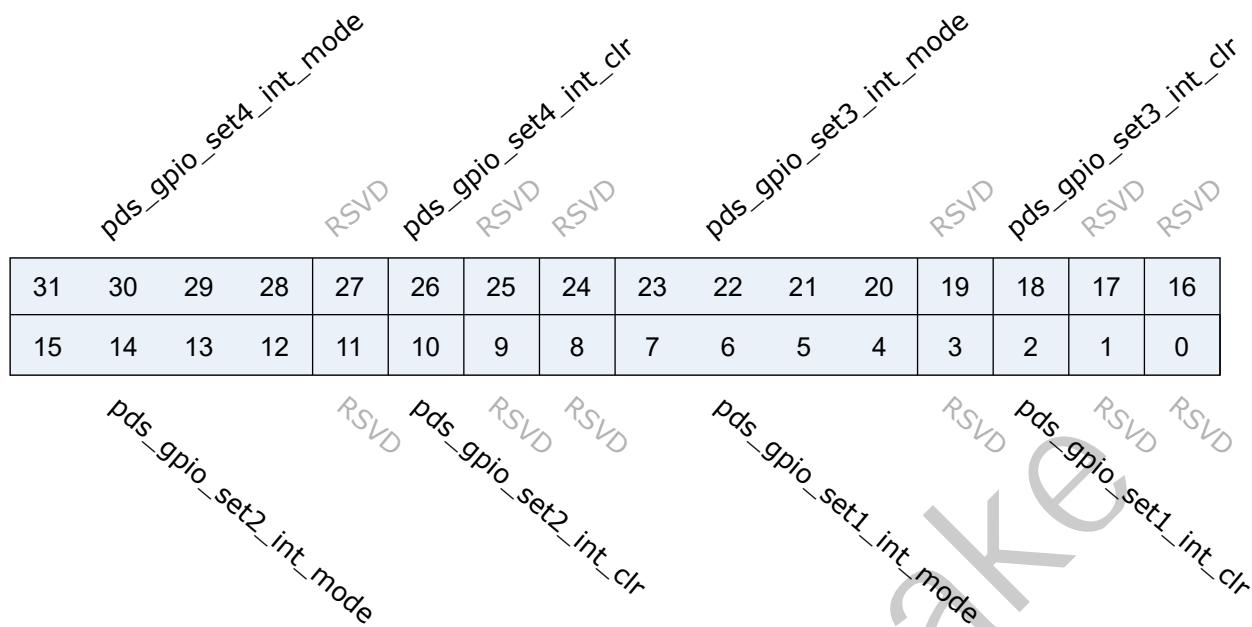
地址: 0x2000e034



位	名称	权限	复位值	描述
31	RSVD			
30:0	cr_pds_gpio_set_int_mask	r/w	31'h7FFFFFFF	PDS Interrupt Mask for GPIO [0] GPIO0 [1] GPIO1 [2] GPIO2 [3] GPIO3 [4] GPIO4 [5] GPIO5 [6] GPIO6 [7] GPIO7 [8] GPIO8 [9] GPIO9 [10] GPIO10 [11] GPIO11 [12] GPIO12 [13] GPIO13 [14] GPIO14 [15] GPIO15 [16] GPIO20 [17] GPIO21 [18] GPIO22 [19] GPIO23 [20] GPIO24 [21] GPIO25 [22] GPIO26 [23] GPIO27 [24] GPIO28 [25] GPIO29 [26] GPIO30 [27] GPIO31 [28] GPIO32 [29] GPIO33 [30] GPIO34

### 25.4.13 pds\_gpio\_int

地址: 0x2000e040



位	名称	权限	复位值	描述
31:28	pds_gpio_set4_int_mode	r/w	4'b0	GPIO28 34 PDS Interrupt Mode 0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
27	RSVD			
26	pds_gpio_set4_int_clr	r/w	1'b0	Clear GPIO28 34 PDS IO Interrupt
25:24	RSVD			

位	名称	权限	复位值	描述
23:20	pds_gpio_set3_int_mode	r/w	4'b0	GPIO20 27 PDS Interrupt Mode 0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
19	RSVD			
18	pds_gpio_set3_int_clr	r/w	1'b0	Clear GPIO20 27 PDS IO Interrupt
17:16	RSVD			
15:12	pds_gpio_set2_int_mode	r/w	4'b0	GPIO8 15 PDS Interrupt Mode 0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
11	RSVD			
10	pds_gpio_set2_int_clr	r/w	1'b0	Clear GPIO8 15 PDS IO Interrupt
9:8	RSVD			
7:4	pds_gpio_set1_int_mode	r/w	4'b0	GPIO0 7 PDS Interrupt Mode 0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger
3	RSVD			
2	pds_gpio_set1_int_clr	r/w	1'b0	Clear GPIO0 7 PDS IO Interrupt

位	名称	权限	复位值	描述
1:0	RSVD			

#### 25.4.14 pds\_gpio\_stat

地址: 0x2000e044

pds_gpio_int_stat																
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	
pds_gpio_int_stat																

位	名称	权限	复位值	描述
31	RSVD			
30:0	pds_gpio_int_stat	r	31'b0	

#### 25.4.15 PDS\_RAM3

地址: 0x2000e048

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_ocram_ret																

位	名称	权限	复位值	描述
31:20	RSVD			

位	名称	权限	复位值	描述
19:0	cr_ocram_ret	r/w	20'h0	[19] : 304 320KB OCRAM RET [18] : 288 304KB OCRAM RET [17] : 272 288KB OCRAM RET [16] : 256 272KB OCRAM RET [15] : 240 256KB OCRAM RET [14] : 224 240KB OCRAM RET [13] : 208 224KB OCRAM RET [12] : 192 208KB OCRAM RET [11] : 176 192KB OCRAM RET [10] : 160 176KB OCRAM RET [9] : 144 160KB OCRAM RET [8] : 128 144KB OCRAM RET [7] : 112 128KB OCRAM RET [6] : 96 112KB OCRAM RET [5] : 80 96KB OCRAM RET [4] : 64 80KB OCRAM RET [3] : 48 64KB OCRAM RET [2] : 32 48KB OCRAM RET [1] : 16 32KB OCRAM RET [0] : 0 16KB OCRAM RET

#### 25.4.16 PDS\_RAM4

地址: 0x2000e04c

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

cr\_ocram\_slp

位	名称	权限	复位值	描述
31:20	RSVD			

位	名称	权限	复位值	描述
19:0	cr_ocram_slp	r/w	20'h0	[19] : 304 320KB OCRAM SLEEP [18] : 288 304KB OCRAM SLEEP [17] : 272 288KB OCRAM SLEEP [16] : 256 272KB OCRAM SLEEP [15] : 240 256KB OCRAM SLEEP [14] : 224 240KB OCRAM SLEEP [13] : 208 224KB OCRAM SLEEP [12] : 192 208KB OCRAM SLEEP [11] : 176 192KB OCRAM SLEEP [10] : 160 176KB OCRAM SLEEP [9] : 144 160KB OCRAM SLEEP [8] : 128 144KB OCRAM SLEEP [7] : 112 128KB OCRAM SLEEP [6] : 96 112KB OCRAM SLEEP [5] : 80 96KB OCRAM SLEEP [4] : 64 80KB OCRAM SLEEP [3] : 48 64KB OCRAM SLEEP [2] : 32 48KB OCRAM SLEEP [1] : 16 32KB OCRAM SLEEP [0] : 0 16KB OCRAM SLEEP

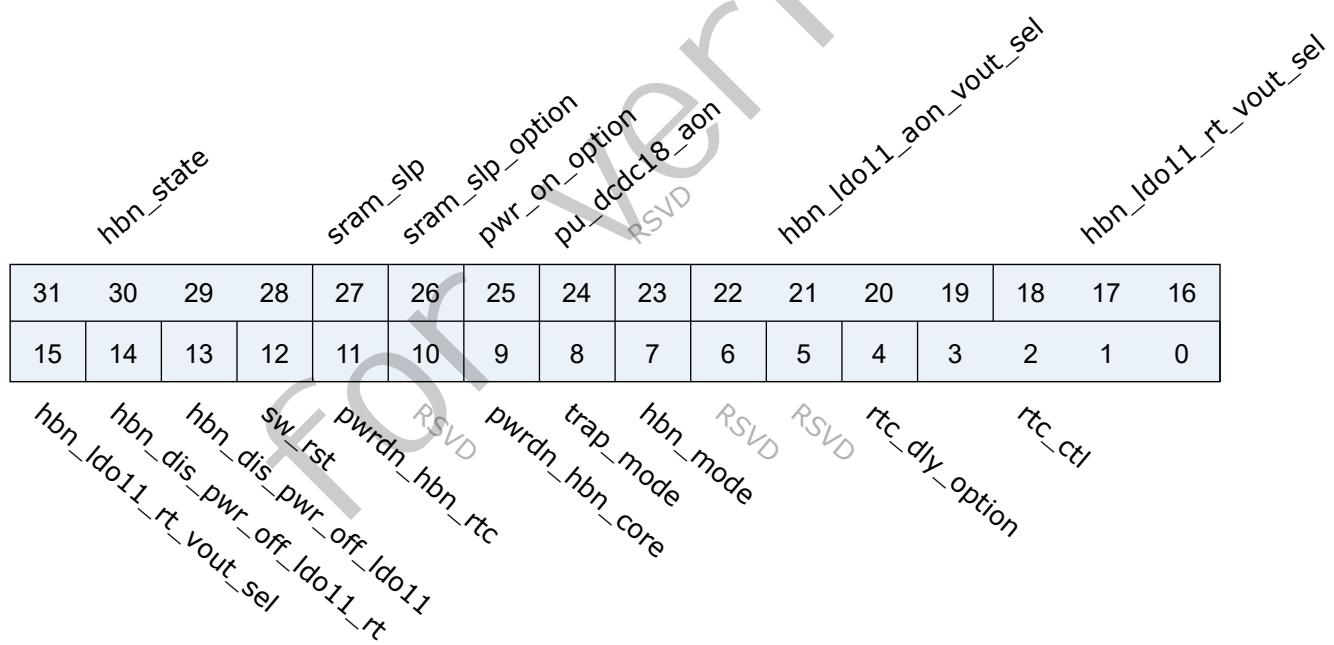
## 25.5 寄存器描述

名称	描述
HBN_CTL	
HBN_TIME_L	
HBN_TIME_H	
RTC_TIME_L	
RTC_TIME_H	
HBN_IRQ_MODE	
HBN_IRQ_STAT	
HBN_IRQ_CLR	
HBN_PIR_CFG	
HBN_PIR_VTH	

名称	描述
HBN_PIR_INTERVAL	
HBN_BOR_CFG	
HBN_GLB	
HBN_SRAM	
HBN_PAD_CTRL_0	
HBN_PAD_CTRL_1	
vbat_ldo	
rc32k_ctrl0	
xtal32k	

### 25.5.1 HBN\_CTL

地址: 0x2000f000



位	名称	权限	复位值	描述
31:28	hbn_state	r	4'h0	SW polling until 0 (default 4'h3 @pwron)
27	sram_slp	r	1'b0	SW polling until 0 (default 1'b1 @pwron)
26	sram_slp_option	r/w	0	

位	名称	权限	复位值	描述
25	pwr_on_option	r/w	0	
24	pu_dcdc18_aon	r/w	1	Power On DCDC18 during Power On Sequence (require 400 800us)
23	RSVD			
22:19	hbn_ldo11_aon_vout_sel	r/w	4'hA	VDD11_AON Voltage Out Select @ Enter hibernate
18:15	hbn_ldo11_rt_vout_sel	r/w	4'hA	VDD11_RT Voltage Out Select @ Enter hibernate
14	hbn_dis_pwr_off_ldo11_rt	r/w	0	Set 1 to disable power off VDDCORE_RT at HBN mode (for low power)
13	hbn_dis_pwr_off_ldo11	r/w	0	Set 1 to disable power off VDDCORE at HBN mode (for debug)
12	sw_RST	r/w	0	soft reset
11	pwrdn_hbn_RTC	r/w	0	Power Off HBN RTC @ Enter hibernate
10	RSVD			
9	pwrdn_hbn_core	r/w	0	Power Off HBN Core @ Enter hibernate
8	trap_mode	r	0	Boot Strap 0: Flash, 1: UART or SDIO
7	hbn_mode	w	0	Enter hibernate
6:5	RSVD			
4	rtc_dly_option	r/w	0	
3:0	rtc_ctl	r/w	4'h0	[6:4] Slow LED, x/0.25/0.5/1/2/4/8/16 seconds (move to 0x38) [3] rtc long time 0 353days (bit 39 13 compare) [2] rtc short time 0 488s (bit 23 0 compare) [1] rtc time 0 353days (bit 39 0 compare) [0] rtc enable

### 25.5.2 HBN\_TIME\_L

地址: 0x2000f004

hbn\_time\_l

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

hbn\_time\_l

位	名称	权限	复位值	描述
31:0	hbn_time_l	r/w	32'h0	RTC timer compare bit 31:0

### 25.5.3 HBN\_TIME\_H

地址: 0x2000f008

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
hbn_time_h															
RSVD															
RSVD															
hbn_time_h															

### 25.5.4 RTC\_TIME\_L

地址: 0x2000f00c

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
rtc_time_latch_l															
RSVD															
rtc_time_latch_l															
hbn_time_h															

位	名称	权限	复位值	描述
31:0	rtc_time_latch_l	r	32'h0	RTC time latched value bit 31:0

### 25.5.5 RTC\_TIME\_H

地址: 0x2000f010

rtc_time_latch_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

位	名称	权限	复位值	描述
31	rtc_time_latch	w	0	RTC time latch for SW read
30:8	RSVD			
7:0	rtc_time_latch_h	r	8'h0	RTC time latched value bit 39:32

### 25.5.6 HBN\_IRQ\_MODE

地址: 0x2000f014

hbn_pin_wakeup_mode															
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

位	名称	权限	复位值	描述
31:28	RSVD			
27	pin_wakeup_en	r/w	0	Pin wakeup delay enable

位	名称	权限	复位值	描述
26:24	pin_wakeup_sel	r/w	3'd3	Pin wakeup delay 1 7 sec
23:22	irq_acomp1_en	r/w	0	enable acomp1 interrupt [20] posedge [21] negedge
21:20	irq_acomp0_en	r/w	0	enable acomp0 interrupt [20] posedge [21] negedge
19	RSVD			
18	irq_bor_en	r/w	0	enable brown-out interrupt
17	RSVD			
16	reg_en_hw_pu_pd	r/w	1	1: Pull GPIO17 @ pwr_on and pwr_RST 0 : no pull
15:8	RSVD			
7:4	hbn_pin_wakeup_mask	r/w	4'b0	mask hbn_pin_wakeup_event
3:0	hbn_pin_wakeup_mode	r/w	4'b0101	hbn_pin_wakeup mode 0000 : sync falling edge trigger 0001 : sync rising edge trigger 0010 : sync low level trigger 0011 : sync high level trigger 01xx : sync rising & falling edge trigger 1000 : async falling edge trigger 1001 : async rising edge trigger 1010 : async low level trigger 1011 : async high level trigger

### 25.5.7 HBN\_IRQ\_STAT

地址: 0x2000f018

irq_stat															
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
irq_stat															

位	名称	权限	复位值	描述
31:0	irq_stat	r	0	[22] acomp1 [20] acomp0 [18] brown-out [17] irq_pir state [16] irq_RTC state [3:0] hbn_pin_wakeup state (GPIO19/18/17/16)

### 25.5.8 HBN\_IRQ\_CLR

地址: 0x2000f01c

irq\_clr

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

irq\_clr

位	名称	权限	复位值	描述
31:0	irq_clr	w	0	[22] irq_acomp1 clear [20] irq_acomp0 clear [18] irq_bor clear [17] irq_pir clear [16] irq_RTC clear [3:0] hbn_pin_wakeup state (GPIO19/18/17/16)

### 25.5.9 HBN\_PIR\_CFG

地址: 0x2000f020

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	pir_en	RSVD	pir_dis	RSVD	pir_lpf_sel	pir_hpf_sel										
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	

位	名称	权限	复位值	描述
31:9	RSVD			
8	gpadc_cs	r/w	0	GPADC clock source select signal 0: 32MHz clock 1: PIR clock (f32k_clk)
7	pir_en	r/w	0	pir enable
6	RSVD			

位	名称	权限	复位值	描述
5:4	pir_dis	r/w	0	pir disable [4] low -> high won't trigger interrupt [5] high -> low won't trigger interrupt
3	RSVD			
2	pir_lpf_sel	r/w	0	0: 1. 1:/2
1:0	pir_hpf_sel	r/w	0	0: 1-z-1, 1: 1-z-2, 0: 2-z-3

### 25.5.10 HBN\_PIR\_VTH

地址: 0x2000f024

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

pir\_vth

位	名称	权限	复位值	描述
31:14	RSVD			
13:0	pir_vth	r/w	14'h3ff	PIR compare threshold

### 25.5.11 HBN\_PIR\_INTERVAL

地址: 0x2000f028

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

pir\_interval

位	名称	权限	复位值	描述
31:12	RSVD			

位	名称	权限	复位值	描述
11:0	pir_interval	r/w	12'd2621	pir_lpf_sel = 0: 32768 / (pir_interval+1) Hz, default 12.5Hz ( 80ms) pir_lpf_sel = 1: 32768 / (pir_interval*2+1) Hz, default 6.25Hz ( 160ms)

### 25.5.12 HBN\_BOR\_CFG

地址: 0x2000f02c

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	

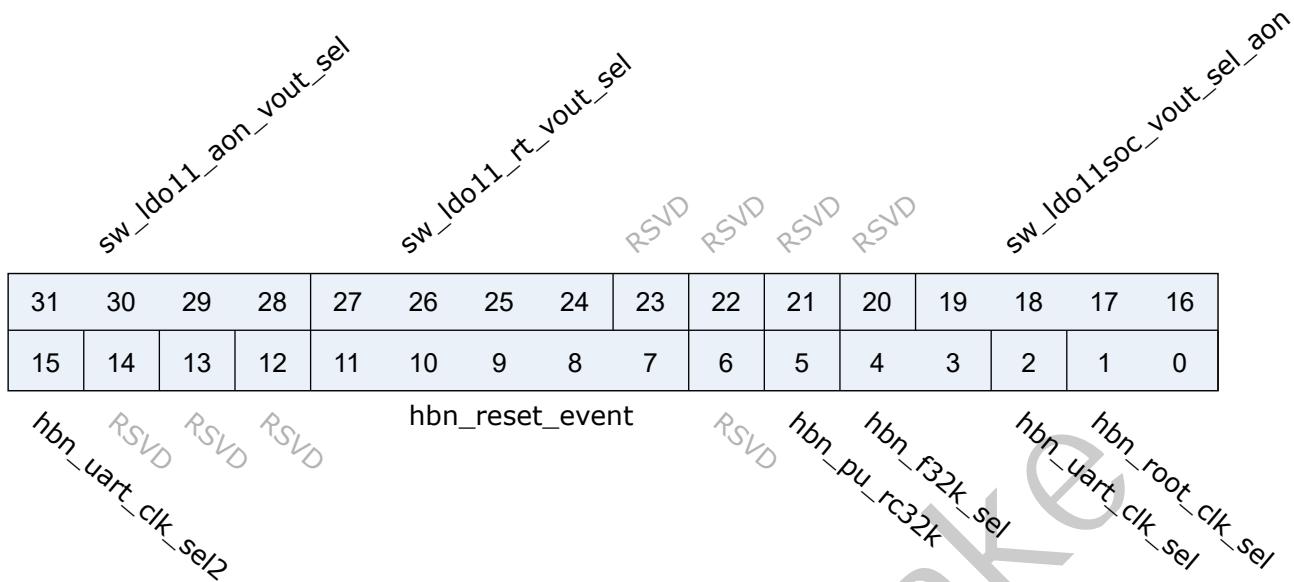
RSVD RSVD

*r\_bod\_out* *pu\_bod* *bod\_vth* *bod\_sel*

位	名称	权限	复位值	描述
31:6	RSVD			
5	r_bod_out	r	1'h0	
4	pu_bod	r/w	1'h0	Power up Brown Out Reset
3:1	bod_vth	r/w	3'h5	bod threshold 000: 2.05V, 001: 2.10V, 010: 2.15V, 011: 2.20V, 100: 2.25V, 101: 2.30V, 110: 2.35V, 111: 2.40V
0	bod_sel	r/w	1'h0	0: POR is independent of BOD, 1: POR is relevant to BOD

### 25.5.13 HBN\_GLB

地址: 0x2000f030



位	名称	权限	复位值	描述
31:28	sw_ldo11_aon_vout_sel	r/w	4'ha	aon ldo output voltage external control: 0:0.70V, 1:0.70V, 2:0.70V, 3:0.75V, 4:0.80V, 5:0.85V, 6:0.9V, 7:0.95V 8:1.0V, 9:1.05V, 10:1.1V, 11:1.15V, 12:1.2V, 13:1.25V, 14:1.3V, 15:1.35V
27:24	sw_ldo11_rt_vout_sel	r/w	4'ha	ldo output voltage external control: 0:0.70V, 1:0.70V, 2:0.70V, 3:0.75V, 4:0.80V, 5:0.85V, 6:0.9V, 7:0.95V 8:1.0V, 9:1.05V, 10:1.1V, 11:1.15V, 12:1.2V, 13:1.25V, 14:1.3V, 15:1.35V
23:20	RSVD			
19:16	sw_ldo11soc_vout_sel_aon	r/w	4'hA	vdd11soc output voltage selection
15	hbn_uart_clk_sel2	r/w	0	UART clock selection2 from HBN (0 : result of hbn_uart_clk_sel (bclk or MUX 160MHz), 1: XCLK (XTAL or RC32M))
14:12	RSVD			
11:7	hbn_reset_event	r	5'b0	[4] : bor_out_event [3] : pwr_RST_N event [2] : sw_RST event [1] : por_OUT event [0] : watch dog reset

位	名称	权限	复位值	描述
6	RSVD			
5	hbn_pu_rc32k	r/w	1	0: Turn off rc32k during rtc power domain off, 1: Don't turn off rc32k (move to RTC Domain)
4:3	hbn_f32k_sel	r/w	0	32KHz clock source selection (0: RC32K 1: XTAL 32K 3: DIG 32K)
2	hbn_uart_clk_sel	r/w	0	UART clock selection from HBN (0:bclk 1:muxpll_160m_clk)
1:0	hbn_root_clk_sel	r/w	0	root clock source selection (0: RC32M 1: XTAL 2/3: PLL)

### 25.5.14 HBN\_SRAM

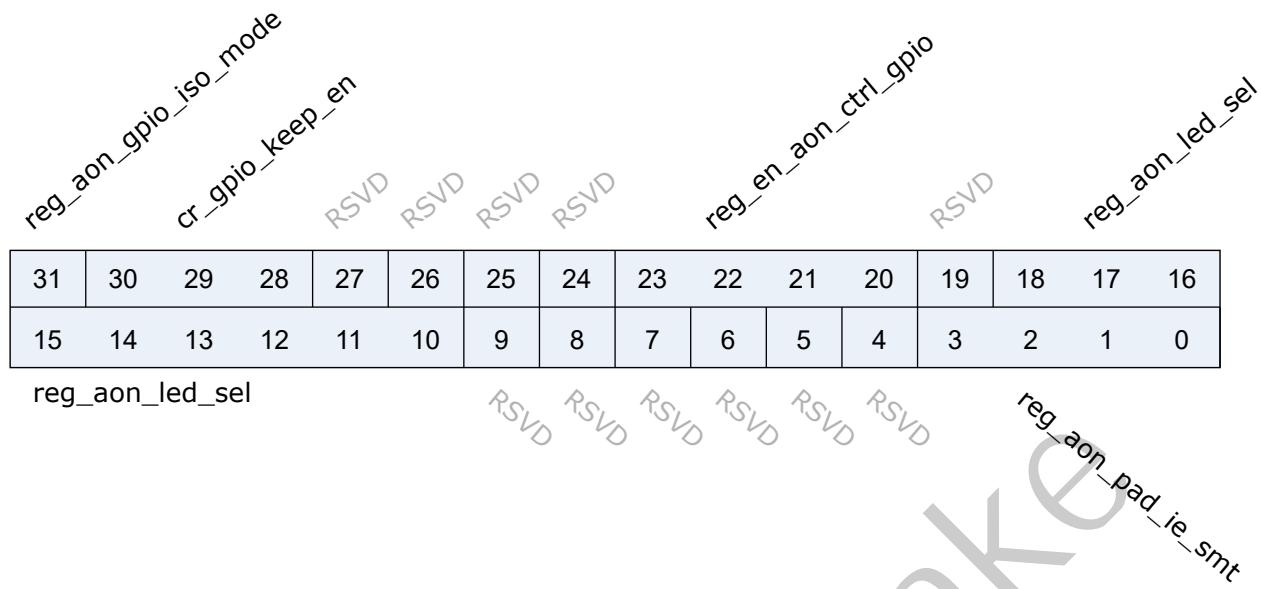
地址: 0x2000f034

| 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    |      |

位	名称	权限	复位值	描述
31:8	RSVD			
7	retram_slp	r/w	0	
6	retram_ret	r/w	0	
5:0	RSVD			

### 25.5.15 HBN\_PAD\_CTRL\_0

地址: 0x2000f038



位	名称	权限	复位值	描述
31	reg_aon_gpio_iso_mode	r/w	0	1: HW Keep GPIO @ PDS or HBN Mode 0 : No Keep GPIO @ PDS or HBN Mode
30:28	cr_gpio_keep_en	r/w	0	if cr_aon_ctrl_gpio_latch=1, can use bit to enable or disable IO latch function at enter HBN Mode [0] : GPIO0 15 [1] : GPIO20 36 [2] : GPIO16 19
27:24	RSVD			
23:20	reg_en_aon_ctrl_gpio		4'h0	AON GPIO16 19 Control by AON HW : [23] :GPIO19 [22]: GPIO18 [21]: GPIO17(can be muxed to be XTAL32K) [20]: GPIO16(can be muxed to be XTAL32K)
19	RSVD			

位	名称	权限	复位值	描述
18:10	reg_aon_led_sel		9'h000	(if corresponding AON GPIO controlled by AON HW) Always on PAD Output Slow LED, x/0.25/0.5/1/2/4/8/16 seconds [12:10] (for GPIO16) : reg_aon_led1_sel [15:13] (for GPIO17) : reg_aon_led2_sel [18:16] (for GPIO18/19) : reg_aon_led3_sel 1 : 0.25sec 2 : 0.5sec 3: 1 sec 4: 2sec 5: 4 sec 6 : 8sec 7 :16sec
9:4	RSVD			
3:0	reg_aon_pad_ie_smt		4'h0	Always on PAD IE/SMT (if corresponding AON GPIO controlled by AON HW) [3] : GPIO19 [2] : GPIO18 [1] : GPIO17 [0] : GPIO16

### 25.5.16 HBN\_PAD\_CTRL\_1

地址: 0x2000f03c

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															
reg_aon_pad_pd															

位	名称	权限	复位值	描述
31:24	RSVD			

位	名称	权限	复位值	描述
23:20	reg_aon_pad_pu		4'h0	Always on PAD PU (if corresponding AON GPIO controlled by AON HW) [23] : GPIO19 [22] : GPIO18 [21] : GPIO17 [20] : GPIO16
19:14	RSVD			
13:10	reg_aon_pad_pd		4'h0	Always on PAD PD (if corresponding AON GPIO controlled by AON HW) [13] : GPIO19 [12] : GPIO18 [11] : GPIO17 [10] : GPIO16
9:4	RSVD			
3:0	reg_aon_pad_oe		4'h0	Always on PAD OE (if corresponding AON GPIO controlled by AON HW) [3] : GPIO19 [2] : GPIO18 [1] : GPIO17 [0] : GPIO16

### 25.5.17 vbat\_ldo

地址: 0x2000f040

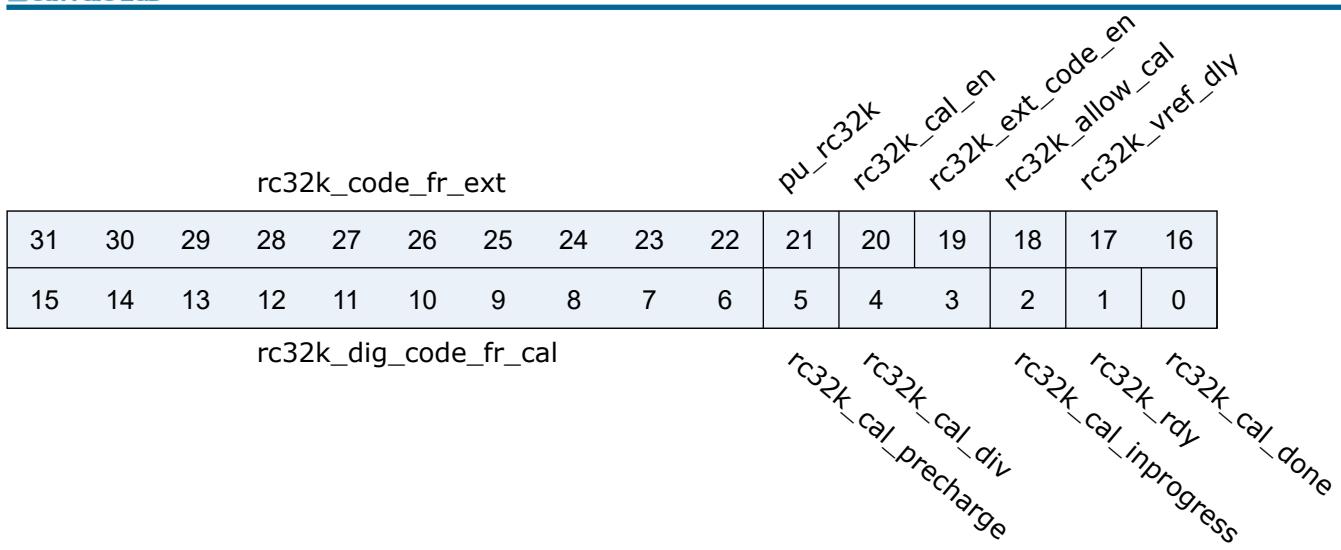
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ldo33_otp_sd_aon	ldo33_otp_th_aon	ten_ldo33_aon	ldo33_ocp_out_aon	ldo33_otp_out_aon	ldo33_otp_en_aon	ldo33_vout_trim_aon	ldo33_vout_sel_aon								
ldo33_sstart_en_aon	ldo33_sstart_delay_aon	RSVD	ldo33_ocp_th_aon	ldo33_ocp_en_aon	ldo33_cc_aon	RSVD	RSVD	ldo33_bm_aon							

位	名称	权限	复位值	描述
31	ldo33_otp_sd_aon	r/w	1'h0	0: do not pulldown ldo33 when OTP happens, 1: pulldown ldo33 when OTP happens
30:28	ldo33_otp_th_aon	r/w	3'h7	OTP temperature threshold selection: 0: 125C, 1: 150C, 2: 175C, 3: 200C, 4: 200C, . 7: 200C
27	ten_ldo33_aon	r/w	1'h0	
26	ldo33_ocp_out_aon	r	1'h0	OCPI signal
25	ldo33_otp_out_aon	r	1'h0	OTP signal
24	ldo33_otp_en_aon	r/w	1'h1	enable Over Temperature Protection circuit in vbat_top
23:20	ldo33_vout_trim_aon	r/w	4'h8	output voltage trim: 1
19:16	ldo33_vout_sel_aon	r/w	4'hb	avdd33_out voltage selection: 0: 2.10, 1: 2.20, 2: 2.30, 3: 2.40, 4: 2.50, 5: 2.70, 6: 2.90, 7: 3.00, 8: 3.10, 9: 3.20, a: 3.25, b: 3.30, c: 3.35, d: 3.40, e: 3.50, f: 3.60
15	ldo33_sstart_en_aon	r/w	1'h1	enable Soft-start circuit in vbat_top

位	名称	权限	复位值	描述
14:12	ldo33_sstart_delay_aon	r/w	3'h3	Sstart time selection: 0: 200us, 1: 290us, 2: 390us, 3: 480us, 4: 580us, 5: 670us, 6: 770us, 7: 860us
11	RSVD			
10:8	ldo33_ocp_th_aon	r/w	3'h5	OCP current threshold selection: 0: 0.5A, 1: 0.6A, 2: 0.8A, 3: 1.0A, 4: 1.2A, 5: 1.5A, 6: 2.0A, 7: 2.0A
7	ldo33_ocp_en_aon	r/w	1'h1	enable Over Current Protection circuit in vbat_top
6:4	ldo33_cc_aon	r/w	3'h1	Compensation Capacitance selection
3:2	RSVD			
1:0	ldo33_bm_aon	r/w	2'h1	Bias mode selection

### 25.5.18 rc32k\_ctrl0

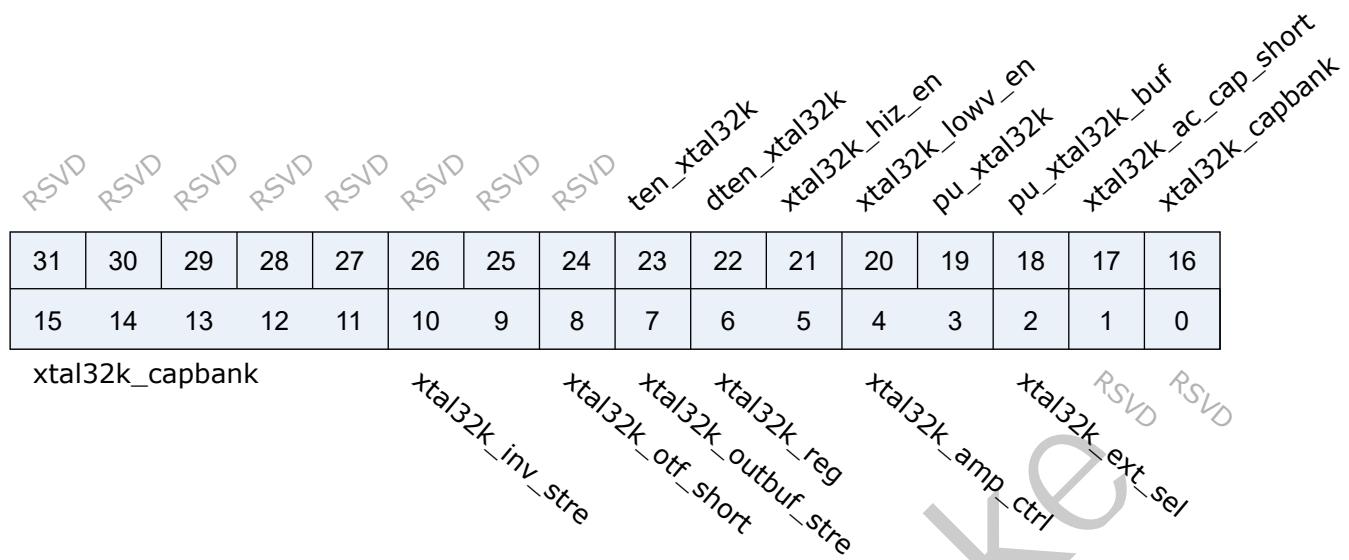
地址: 0x2000f200



位	名称	权限	复位值	描述
31:22	rc32k_code_fr_ext	r/w	10'h12C	External code In for frequency setting
21	pu_rc32k	r/w	1	Power up 32k oscillator (Useless in 616)
20	rc32k_cal_en	r/w	0	Enable calibration of 32k oscillator
19	rc32k_ext_code_en	r/w	1	Allow external code in to go into the ckt
18	rc32k_allow_cal	r/w	0	Allow calibration to be performed (monitor system clock)
17:16	rc32k_vref_dly	r/w	0	reference power up delay
15:6	rc32k_dig_code_fr_cal	r	10'h200	After calibration hold calibrated code
5	rc32k_cal_precharge	r	0	Initial a new cal
4:3	rc32k_cal_div	r/w	2'h3	Divider for the clock step during calibration
2	rc32k_cal_inprogress	r	0	Asserts high when calibration is in progress
1	rc32k_rdy	r	1	Asserts high when 32k clock is ready upon pwrup
0	rc32k_cal_done	r	1	Asserts high when calibration is done

## 25.5.19 xtal32k

地址: 0x2000f204



位	名称	权限	复位值	描述
31:24	RSVD			
23	ten_xtal32k	r/w	1'h0	
22	dten_xtal32k	r/w	1'h0	
21	xtal32k_hiz_en	r/w	1	high-z xtal32k input/output for share gpio usage
20	xtal32k_lowv_en	r/w	1'h0	xtal32k low voltage mode enable
19	pu_xtal32k	r/w	0	power up signal for 32K crystal oscillator
18	pu_xtal32k_buf	r/w	1	1: power up XTAL32k level shifter buffer
17	xtal32k_ac_cap_short	r/w	0	
16:11	xtal32k_capbank	r/w	6'h20	32K crystal load cap control word (also copy from efuse)
10:9	xtal32k_inv_stre	r/w	2'h1	32K crystal inverter amplify strength
8	xtal32k_otf_short	r/w	0	32K crystal over tone filter short
7	xtal32k_outbuf_stre	r/w	0	32K crystal output buffer strength
6:5	xtal32k_reg	r/w	2'h1	32K crystal voltage regulator level
4:3	xtal32k_amp_ctrl	r/w	2'h1	32K crystal oscillation amplitude control
2	xtal32k_ext_sel	r/w	0	External 32K clock enable
1:0	RSVD			

## 26.1 简介

### 26.1.1 AES 简介

高级加密标准 (AES, Advanced Encryption Standard) 为最常见的对称加密算法。在密码学中又称 Rijndael 加密法，是美国联邦政府采用的一种区块加密标准。

### 26.1.2 SHA 简介

SHA256 是 SHA-2 下细分出的一种算法，SHA-2，名称来自于安全散列算法 2 (英语：Secure Hash Algorithm 2) 的缩写，一种密码散列函数算法标准，由美国国家安全局研发，属于 SHA 算法之一，是 SHA-1 的后继者。SHA-2 下又可再分为六个不同的算法标准，包括了：SHA-224、SHA-256、SHA-384、SHA-512、SHA-512/224、SHA-512/256。其中 SHA-512/224 指结果取 SHA-512 的前 224 个 bit，SHA-512/256 指结果取 SHA-512 的前 256 个 bit。

### 26.1.3 CRC 简介

循环冗余校验 (Cyclic Redundancy Check, CRC) 是一种根据网络数据包或计算机文件等数据产生简短固定位数校验码的一种信道编码技术，主要用来检测或校验数据传输或者保存后可能出现的错误。它是利用除法及余数的原理来作错误侦测的。

### 26.1.4 GMAC 简介

GMAC 就是利用伽罗华域 (Galois Field, GF, 有限域) 乘法运算来计算消息的 MAC 值。

## 26.2 主要特征

- 支持 AES-128, AES-192, AES-256 加解密
- 支持 SHA-256, SHA-512
- 支持 CRC-16, CRC-32
- 支持 GMAC
- 支持真随机数发生器 TRNG

## 26.3 原理描述

### 26.3.1 AES 加解密流程

AES 对称加密算法也就是加密和解密用相同的密钥，加密流程如下图：

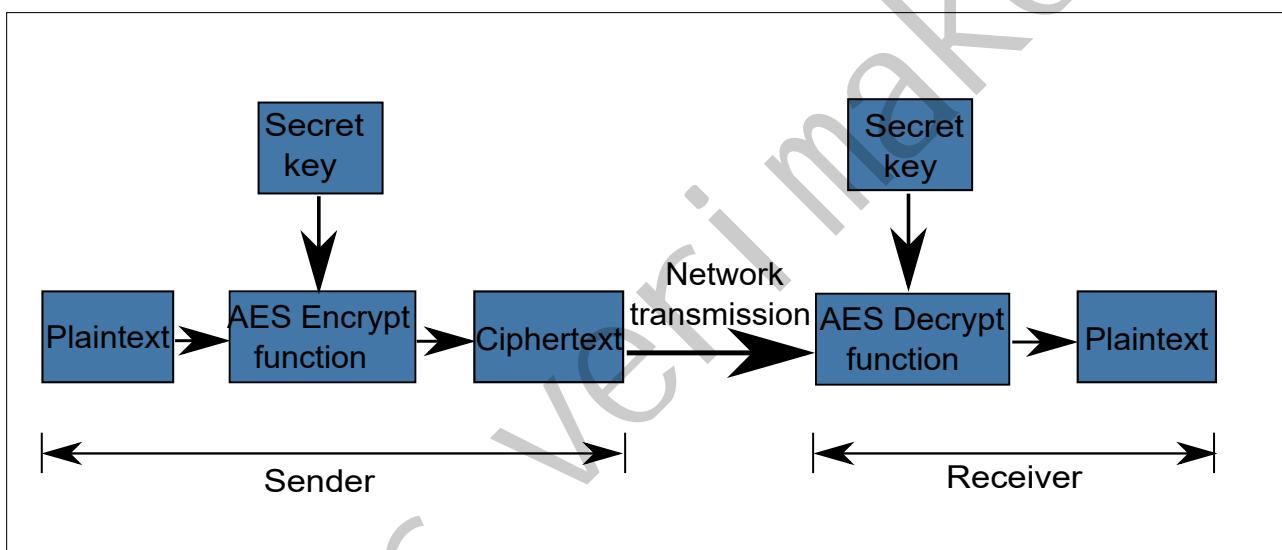


图 26.1: AES 加密流程图

下面简单介绍下各个部分的作用与意义：

- 明文 **P**: 没有经过加密的数据。
- 密钥 **K**: 用来加密明文的密码，在对称加密算法中，加密与解密的密钥是相同的。密钥为接收方与发送方协商产生，但不可以直接在网络上传输，否则会导致密钥泄漏，通常是通过非对称加密算法加密密钥，然后再通过网络传输给对方，或者直接面对面商量密钥。密钥是绝对不可以泄漏的，否则会被攻击者还原密文，窃取机密数据。
- AES 加密函数：设 AES 加密函数为  $E$ ，则  $C = E(K, P)$ ，其中  $P$  为明文， $K$  为密钥， $C$  为密文。也就是说，把明文  $P$  和密钥  $K$  作为加密函数的参数输入，则加密函数  $E$  会输出密文  $C$ 。
- 密文 **C**: 经加密函数处理后的数据。
- AES 解密函数: 设 AES 解密函数为  $D$ ，则  $P = D(K, C)$ ，其中  $C$  为密文， $K$  为密钥， $P$  为明文。也就是说，把密文

C 和密钥 K 作为解密函数的参数输入，则解密函数会输出明文 P。

### 26.3.2 SHA256 的实现

SHA256 算法中用到了 8 个哈希初值以及 64 个哈希常量。

其中，SHA256 算法的 8 个哈希初值如下：

- h0 = 0x6a09e667
- h1 = 0xbb67ae85
- h2 = 0x3c6ef372
- h3 = 0xa54ff53a
- h4 = 0x510e527f
- h5 = 0x9b05688c
- h6 = 0x1f83d9ab
- h7 = 0x5be0cd19

这些初值是对自然数中前 8 个质数（2,3,5,7,11,13,17,19）的平方根的小数部分取前 32bit 而来。

在 SHA256 算法中，用到的 64 个常量如下：

- 428a2f98 71374491 b5c0fbcf e9b5dba5
- 3956c25b 59f111f1 923f82a4 ab1c5ed5
- d807aa98 12835b01 243185be 550c7dc3
- 72be5d74 80deb1fe 9bdc06a7 c19bf174
- e49b69c1 efbe4786 0fc19dc6 240ca1cc
- 2de92c6f 4a7484aa 5cb0a9dc 76f988da
- 983e5152 a831c66d b00327c8 bf597fc7
- c6e00bf3 d5a79147 06ca6351 14292967
- 27b70a85 2e1b2138 4d2c6dfc 53380d13
- 650a7354 766a0abb 81c2c92e 92722c85
- a2bfe8a1 a81a664b c24b8b70 c76c51a3
- d192e819 d6990624 f40e3585 106aa070
- 19a4c116 1e376c08 2748774c 34b0bcb5

- 391c0cb3 4ed8aa4a 5b9cca4f 682e6ff3
- 748f82ee 78a5636f 84c87814 8cc70208
- 90beffa a4506ceb bef9a3f7 c67178f2

和 8 个哈希初值类似，这些常量是对自然数中前 64 个质数 (2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67,71, 73, 79,83,89,97...) 的立方根的小数部分取前 32bit 而来。

SHA256 散列函数：

- $Ch(x,y,z) = (x \square y) \square (\neg x \square z)$
- $Ma(x,y,z) = (x \square y) \square (x \square z) \square (y \square z)$
- $\Sigma 0(x) = S^2(x) \square S^{13}(x) \square S^{22}(x)$
- $\Sigma 1(x) = S^6(x) \square S^{11}(x) \square S^{25}(x)$
- $\sigma 0(x) = S^7(x) \square S^{18}(x) \square R^3(x)$
- $\sigma 1(x) = S^{17}(x) \square S^{19}(x) \square R^{10}(x)$

其中：

- $\square$ ：按位“与”
- $\neg$ ：按位“补”
- $\square$ ：按位“异或”
- $S^n$ ：循环右移 n 个 bit
- $R^n$ ：右移 n 个 bit

### 26.3.3 GMAC 的原理

消息认证实际上是对消息本身产生的一个冗余的信息，即消息验证码（MAC）。消息认证码（Message authentication code）是一种确认完整性并进行认证的一种技术，简称 MAC。密码学中，消息认证码指的是通信实体双方使用的一种验证机制，保证消息数据完整性的一种工具。消息认证码是一种带密钥的哈希函数，它本质上是一个哈希函数，那为什么要带密钥呢？是因为消息在传输过程中是可以被篡改，哈希值也可以被篡改，因此为了保证这个哈希值的有效性，通过加密的方式将哈希值保护起来，这样在接收方接收到消息后就可以通过这个哈希值来判断整条消息的完整性，从而达到信息传递的目的。

消息认证码步骤如下图所示：

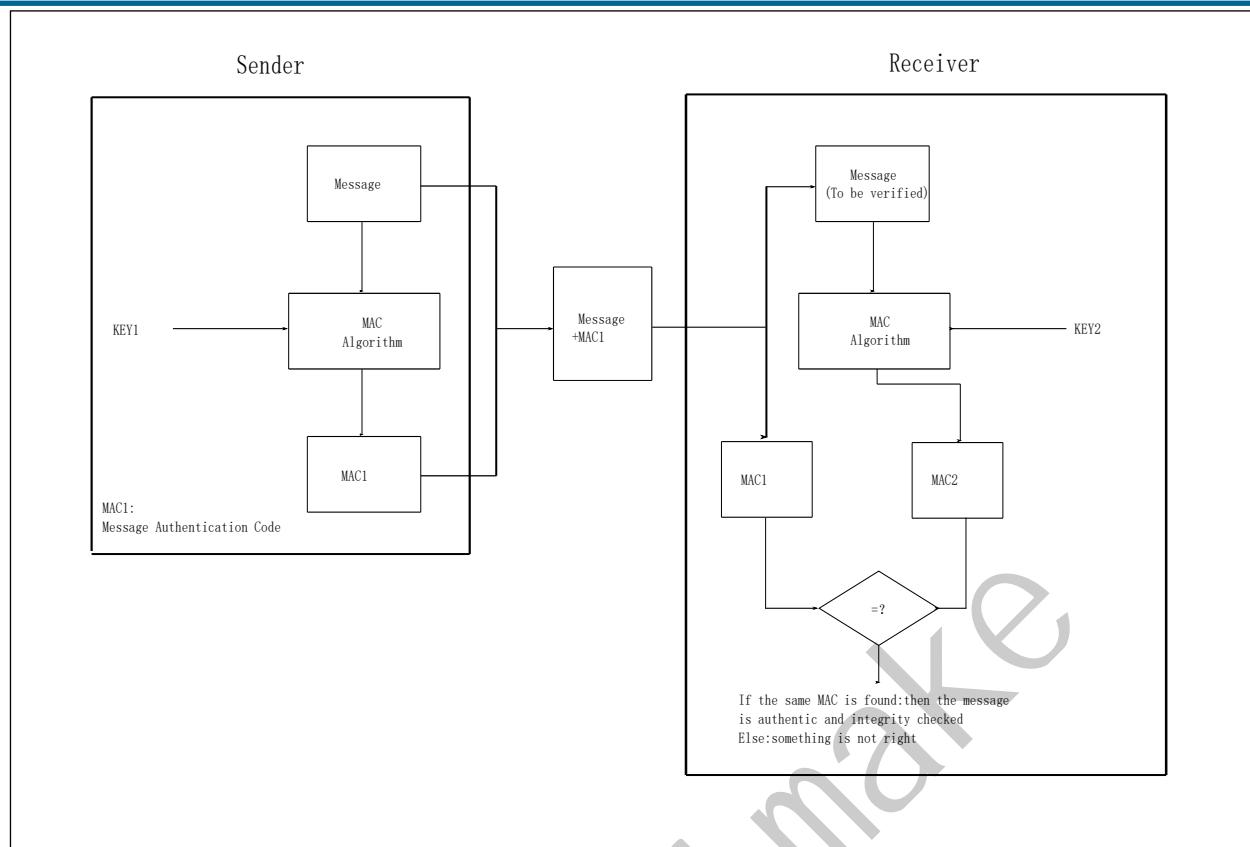


图 26.2: 消息认证码流程图

流程如下：

- 发送者与接收者事先共享密钥 **K** (上图中的 **KEY1** 与 **KEY2** 值保持一致)。
- 发送者根据消息计算 **MAC** 值 (使用密钥 **KEY1** 对原始消息计算 **MAC1**)。
- 发送者将原始消息和 **MAC1** 发送给接收者。
- 接收者根据收到的原始消息计算 **MAC2** (使用密钥 **KEY2**)。
- 接收者将自己计算出的 **MAC2** 与从发送者收到的 **MAC1** 比对。
- 如果 **MAC** 一致, 接收者可以判定消息的确来自接收者 (认证成功) 且没有被篡改或者出现传输出错的情况; 如果不一致, 可判断消息不是来自发送方 (认证失败)。

注意：建议发送方和接收方将密钥 **KEY** 存放于硬件安全模块中，计算 **MAC** 值的过程最好也放到硬件安全模块中完成，这样可以保证密钥的安全，例如放到加密芯片中。

**GMAC** 就是利用伽罗华域 (Galois Field, GF, 有限域) 乘法运算来计算消息的 **MAC** 值。

## 26.4 功能描述

### 26.4.1 AES 加速器

1 AES 加速器支持 AES-128/192/256 加解密 6 种运算。

- 配置寄存器 se\_aes\_0\_ctrl 中的 se\_aes\_0\_mode 和 se\_aes\_0\_dec\_en, 如下图所示:

amode	adec en	运算
0	0	AES-128 加密
1	0	AES-256 加密
2	0	AES-192 加密
0	1	AES-128 解密
1	1	AES-256 解密
2	1	AES-192 解密

图 26.3: AES 运算模式图

配置寄存器 se\_aes\_0\_ctrl 中的 se\_aes\_0\_block\_mode 选择不同的加密方式，目前支持 ECB、CTR、CBC、XTS 模式。

#### 2 密钥、明文、密文、初始化向量

- 寄存器 se\_aes\_0\_msa 存放明文或者密文的地址。
- 寄存器 se\_aes\_0\_mda 存放密文或者明文的地址。
- 寄存器 se\_aes\_0\_iv\_0~se\_aes\_0\_iv\_3 存放 IV。
- 寄存器 se\_aes\_0\_key\_0~se\_aes\_0\_key\_7 存放密钥。

#### 3 软硬件加密流程

- 配置寄存器 se\_aes\_0\_endian, 其中包括 se\_aes\_0\_dout\_endian、se\_aes\_0\_din\_endian、se\_aes\_0\_key\_endian、se\_aes\_0\_iv\_endian、se\_aes\_0\_twk\_endian, 若值为 0 表示 little-endian, 值为 1 表示 big-endian
- 配置寄存器 se\_aes\_0\_ctrl 中的 se\_aes\_0\_block\_mode
- 配置寄存器 se\_aes\_0\_ctrl 中的 se\_aes\_0\_mode
- 配置寄存器 se\_aes\_0\_ctrl 中的 se\_aes\_0\_dec\_en
- 配置寄存器 se\_aes\_0\_ctrl 中的 se\_aes\_0\_dec\_key\_sel, 0 表示使用新的 key, 1 表示使用跟上一次相同的 key
- 配置寄存器 se\_aes\_0\_ctrl 中的 se\_aes\_0\_iv\_sel, 0 表示使用新的 iv, 1 表示使用跟上一次相同的 iv

- 配置寄存器 `se_aes_0_ctrl` 中的 `se_aes_0_en` 使能 AES
- 配置寄存器 `se_aes_0_iv_0~se_aes_0_iv_3` 设置 IV, MSB 时填写顺序为 `se_aes_0_iv_0~se_aes_0_iv_3`, LSB 时填写顺序为 `se_aes_0_iv_3~se_aes_0_iv_0`
- 配置寄存器 `se_aes_0_key_0~se_aes_0_key_7` 设置 key, MSB 时填写顺序为 `se_aes_0_key_0~se_aes_0_key_7`, LSB 时填写顺序为 `se_aes_0_key_7~se_aes_0_key_0`。AES-128 取前 4 个, AES-196 取前 6 个, AES-256 取 8 个
- 配置寄存器 `se_aes_0_msa` 设置待处理数据的源地址
- 配置寄存器 `se_aes_0_mda` 设置处理结果存放的目的地址
- 配置寄存器 `se_aes_0_ctrl` 中的 `se_aes_0_msg_len` 设置待处理数据的长度, 以 128-bit 为单位
- 配置寄存器 `se_aes_0_ctrl` 中的 `se_aes_0_trig_1t` 触发 AES 运行
- 结果输出存储在寄存器 `se_aes_0_mda` 对应的地址中

#### 26.4.2 SHA 加速器

1 SHA 加速器支持 7 种标准运算: SHA-1、SHA-224、SHA-256、SHA-512、SHA-384、SHA-512/224、SHA-512/256, 同时还支持 MD5、CRC16、CRC32。

寄存器 `se_sha_0_ctrl` 中的 `se_sha_0_mode` 用于选择 SHA 的具体模式: 0:SHA-256 1:SHA-224 2:SHA-1 3:SHA-1 4:SHA-512 5:SHA-384 6:SHA-512/224 7:SHA-512/256

寄存器 `se_sha_0_ctrl` 中的 `se_sha_0_mode_ext` 由于选择扩展模式, 如果不使用扩展模式应设为 0: 0:SHA 1:MD5 2:CRC-16 3:CRC-32

配置寄存器 `se_sha_0_ctrl` 中的 `se_sha_0_mode` 选择不同的 SHA 运算, 配置寄存器 `se_sha_0_ctrl` 中的 `se_sha_0_mode_ext` 可选择 MD5、CRC16、CRC32。

- 当 `se_sha_0_mode_ext` 为 0 时, `se_sha_0_mode` 有效。
- 当 `se_sha_0_mode_ext` 不为 0 时, `se_sha_0_mode` 无效。

#### 2 明文、密文

- 寄存器 `se_sha_0_msa` 存放明文地址。
- 寄存器 `se_sha_0_hash_l_0~se_sha_0_hash_l_7` 存放密文。

获取顺序备注: - MSB: `se_sha_0_hash_l_0~se_sha_0_hash_l_7`。- LSB: `se_sha_0_hash_l_7~se_sha_0_hash_l_0`。

#### 3 运算流程

- 配置寄存器 `se_sha_0_ctrl` 中的 `se_sha_0_mode` 设置 SHA 的具体模式
- 配置寄存器 `se_sha_0_ctrl` 中的 `se_sha_0_en` 使能 SHA
- 配置寄存器 `se_sha_0_ctrl` 中的 `se_sha_0_hash_sel`, 0 表示开始新的 HASH 计算, 1 表示沿用上一次的结果进行

## HASH 计算

- 配置寄存器 `se_sha_0_msa` 设置待处理数据的源地址
- 配置寄存器 `se_sha_0_ctrl` 中的 `se_sha_0_msg_len` 待处理数据的长度，SHA-1、SHA-224、SHA-256 以 512-bit 为单位，SHA-512、SHA-384、SHA-512/224、SHA-512/256 以 1024-bit 为单位
- 配置寄存器 `se_sha_0_ctrl` 中的 `se_sha_0_trig_1t` 触发 SHA 运行
- 结果输出存储在 `se_sha_0_hash_l_0~se_sha_0_hash_l_7` 中，MSB:`se_sha_0_hash_l_0~se_sha_0_hash_l_7`, LSB:`se_sha_0_hash_l_7~se_sha_0_hash_l_0`

### 26.4.3 GMAC(link 模式)

#### 1. GMAC\_link\_Table 结构体定义

- Word0:
  - [9]:`se_gmac_0_int_clr_1t`
  - [10]:`se_gmac_0_int_set_1t`
  - [31:16]:`se_gmac_0_msg_len`
- Word1:`se_gmac_0_msa`
- Word2、Word3、Word4、Word5: `se_gmac_0_h`
- Word6、Word7、Word8、Word9: `se_gmac_0_tag`

#### 2. 使用流程

- 配置寄存器 `se_gmac_0_ctrl_0` 中的 `se_gmac_0_x_endian`、`se_gmac_0_h_endian`、`se_gmac_0_t_endian` 0:little-endian 1:big-endian
- 将 GMAC\_link\_Table 结构体的起始地址写到寄存器 `se_gmac_0_lca` 中
- 将原始消息地址赋给 GMAC\_link\_Table 结构体中的 Word1 中的 `se_gmac_0_msa`
- 将原始消息长度赋给 GMAC\_link\_Table 结构体中的 Word0 中的 `se_gmac_0_msg_len`, 128bit 消息长度对应 `se_gmac_0_msg_len` 中的 1
- 配置寄存器 `se_gmac_0_ctrl_0` 中的 `se_gmac_0_trig_1t` 触发 GMAC 运行
- 结果输出存储在 GMAC\_link\_Table 结构体中的 Word6~Word9 中

## 26.4.4 真随机数发生器

1. 内置一个真随机数发生器，其生成的随机数可作为加密等操作的基础。
  - 真随机数：真正的随机数是使用物理现象产生的：比如掷钱币、骰子、转轮、使用电子元件的噪音、核裂变等等，这样的随机数发生器叫做物理性随机数发生器，它们的缺点是技术要求比较高。
  - 伪随机数：真正意义上的随机数（或者随机事件）在某次产生过程中是按照实验过程中表现的分布概率随机产生的，其结果是不可预测的，是不可见的。而计算机中的随机函数是按照一定算法模拟产生的，其结果是确定的，是可见的。我们可以认为这个可预见的结果其出现的概率是 100%。所以用计算机随机函数所产生的“随机数”并不随机，是伪随机数。
2. 输出
  - 寄存器 SE\_TRNG\_0\_DOUT\_0~SE\_TRNG\_0\_DOUT\_7 存放输出的随机数。
3. 使用流程
  - 配置寄存器 se\_trng\_0\_ctrl\_0 中的 se\_trng\_0\_en 使能 TRNG
  - 配置寄存器 se\_trng\_0\_ctrl\_0 中的 se\_trng\_0\_trig\_1t 触发 TRNG 运行
  - 结果输出存储在 se\_trng\_0\_dout\_0~se\_trng\_0\_dout\_7 中

表 27.1: 文档版本修改信息

日期	版本	修改内容
2021/9/22	0.9	初版
2022/8/22	0.91	添加寄存器描述