

WM_W800_Registers Manual

V2.1

Beijing Lian Sheng De Microelectronics Co.,

Ltd (winner micro) Address: 18/F, Yindu

Building, No. 67 Fucheng Road, Haidian

District, Beijing, China Tel: +86-10-62161900

Company website: www.winnermicro.com

Document Modification Record

releases	revision time	revised record	author	audits
V0.1	2019-09-25	establish	chengsheng	
V0.2	2020-05-13	Revision of legacy description errors;	Chengsheng	
V1.0	2020-7-20	Updated digital function interface	Ray.	
V2.0	2020-8-20	Improvement of high-speed interface function	Ray.	
V2.1	2020-09-08	Add SD_ADC Module Description	chengsheng	

catalogs

Document Modification Record	2
Catalog	3
Figure Catalog	22
Table of Contents	23
1 Introduction	34
1.1 Purpose of preparation	34
1.2 References	34
2 Features	35
3 Overview	39
4 Chip Structure	41
4.1 Chip Structure	41
4.2 Bus Structures	43
4.3 Clock Architecture	47
4.4 Address Space	48
4.4.1 SRAM	53
4.4.2 Flash	53
4.4.3 PSRAM	54
4.5 Startup Configuration	54
5 Clock and Reset Module	56
5.1 Function Overview	56
5.2 Main Characteristics	56
5.3 Functional Description	56

5.3.1	Clock Gating.....	56
5.3.2	Clock Adaptive Shutdown.....	57
5.3.3	Function reset.....	57
5.3.4	Clock crossover.....	57
5.3.5	Debug Function Control.....	59
5.4	Register Description	60
5.4.1	Register List	60
5.4.2	Software Clock Gating Enable Register	60
5.4.3	Software Clock Mask Register.....	64
5.4.4	Software Reset Control Register	65
5.4.5	Clock Division Configuration Register	71
5.4.6	Debug Control Register	73
5.4.7	I2S Clock Control Register.....	I2S Clock Control Register
5.4.8	Reset Status Register	76
6	DMA Module.....	77
6.1	Function Overview.....	77
6.2	Main Characteristics	77
6.3	Functional Description.....	77
6.3.1	DMA Channel	77
6.3.2	DMA Data Stream	78
6.3.3	DMA Cyclic Mode.....	79
6.3.5	DMA Peripheral Selection.....	79
6.3.6	DMA Chained Table Mode.....	80

6.3.7	DMA Interrupt.....	80
64	Register Description	80
64.1	Register List	80
64.2	Interrupt Mask Register.....	82
64.3	Interrupt Status Register	83
64.4	UART Select Register.....	84
64.5	DMA Source Address Register.....	85
64.6	DMA Destination Address Register	85
64.7	DMA Cycle Source Start Address Register.....	85
64.8	DMA Cycle Destination Start Address Register.....	86
64.9	DMA Cycle Length Register	86
64.10	DMA Channel Control Register.....	86
64.11	DMA Mode Selection Register.....	87
64.12	DMA Data Flow Control Register.....	88
64.13	DMA Transfer Byte Count Register	90
64.14	DMA Link Table Entry Address Register.....	90
64.15	DMA Current Destination Address Register	90
7	Universal Hardware Encryption Module	92
7.1	Function Overview.....	92
7.3	Functional Description	92
7.3.1	SHA1 Encryption	92
7.3.2	MD5 Encryption.....	92
7.3.3	RC4 Encryption.....	93

7.34	DES Encryption.....	93
7.35	3DES Encryption.....	93
7.36	AES encryption.....	93
7.37	CRC Encryption.....	93
7.38	TRNG Random Number Generator.....	94
7.4	Register Description.....	95
7.41	Register List	95
7.42	Configuration Registers.....	96
7.43	TRNG Control Register.....	99
7.44	Control registers.....	100
7.45	Status Register.....	101
8	RSA Encryption Module.....	102
8.1	Function Overview	102
8.2	Main features	102
8.3	Functional Description.....	102
8.31	Modulo Functions.....	102
8.4	Register Description.....	102
8.4.2	Data X Register.....	103
8.4.3	Data Y Register.....	103
8.4.4	Data M Register.....	103
8.4.5	Data D Register	103
8.4.6	RSA Control Register	104
8.4.7	Parameter MC Register	105

8.4.8	Parameter N Register.....	105
9	GPIO Module.....	106
9.1	Function Overview	106
9.2	Main Characteristics	106
9.3	Functional Description.....	106
9.4	Register Description.....	107
9.4.1	Register List	107
9.4.2	GPIO Data Register	109
9.4.3	GPIO Data Enable Register.....	110
9.4.4	GPIO Direction Control Register.....	110
9.4.5	GPIO Pull-Up and Pull-Down Control Registers	111
9.4.6	GPIO Multiplexing Select Register.....	112
9.4.7	GPIO Multiplexing Select Register 1.....	114
9.4.8	GPIO Multiplexing Select Register 0.....	114
9.4.9	GPIO Interrupt Trigger Mode Configuration Register	115
9.4.11	GPIO Interrupt Up and Down Edge Trigger Configuration Registers	116
9.4.12	GPIO Interrupt Enable Configuration Register	117
9.4.13	GPIO Bare Interrupt Status Register	118
9.4.14	GPIO Masked Interrupt Status Register.....	118
9.4.15	GPIO Interrupt Clear Control Register	119
10	High Speed SPI Device Controller.....	120
10.1	Function Overview	120
10.2	Main Characteristics	120

103	Functional Description.....	120
103.1	Introduction to SPI Protocol	120
103.2	SPI Workflow.....	121
104	Register Description.....	121
104.1	List of registers operated inside the HSPI chip	121
104.2	Host Side Access HSPI Controller Register List	125
104.3	High-Speed SPI Device Controller Interface Timing	130
11	SDIO Device Controller	141
11.1	Functional Overview	141
11.2	Main Characteristics	141
11.3	Functional Description.....	141
113.1	SDIO Bus.....	141
113.2	SDIO Commands.....	142
11.4	Register Description.....	144
114.1	Register List	144
114.2	SDIO Fn0 Register.....	144
114.3	SDIO Fn1 register.....	157
12	HSPI/SDIO Wrapper Controller	168
12.1	Function Overview	168
12.2	Key Features.....	168
12.3	Functional Description.....	169
123.1	Uplink data reception function	169
123.2	Downlink Data Migration Function	170

124	Register Description.....	170
124.1	Register List	170
124.2	WRAPPER Interrupt Status Register.....	172
124.3	WRAPPER Interrupt Configuration Registers.....	172
124.4	WRAPPER Uplink Command Ready Register	172
124.5	WRAPPER Down Command buf Ready Register.....	173
124.6	SDIO TX Link Enable Register.....	173
124.7	SDIO TX Link Address Register	174
124.8	SDIO TX Enable Register.....	174
124.9	SDIO TX Status Register.....	174
124.10	SDIO RX Link Enable Register	175
124.12	SDIO RX Enable Register.....	176
124.13	SDIO RX Status Register.....	176
124.14	WRAPPER CMD BUF Base Address Register	177
124.15	WRAPPER CMD BUF SIZE Register	177
13	SDIO HOST Device Controller.....	178
13.1	Function Overview	178
13.2	Main Features.....	178
13.3	Functional Description.....	178
13.4	Register Description.....	179
134.1	Register List	179
14	SPI Controller.....	200
14.1	Function Overview	200

14.2	Main Characteristics	200
14.3	Functional Description.....	200
14.3.1	Master-slave configurable.....	200
14.3.2	Multi-mode support.....	201
14.3.3	Efficient data transfer	201
14.4	Register Description.....	201
14.4.1	Register List	201
14.4.2	Channel Configuration Register	202
14.4.3	SPI Configuration Register	206
14.4.5	Mode Configuration Register.....	210
14.4.6	Interrupt Control Register.....	211
14.4.7	Interrupt Status Register.....	213
14.4.8	SPI Status Register	215
14.4.9	SPI Timeout Register.....	216
14.4.10	Data Transmit Register	216
14.4.11	Transmission Mode Register.....	217
14.4.12	Data Length Register.....	219
14.4.13	Data Receive Register.....	220
15	I2C Controller	221
15.1	Functional Overview	221
15.2	Main Features.....	221
15.3	Functional Description.....	221
15.3.1	Transmission Rate Selection	221

15.32	Interrupt and start-stop controllable.....	222
15.33	Fast output and detection signals	222
15.4	Register Description.....	222
15.4.1	Register List	222
15.4.2	Clock Divider Register_1.....	223
15.4.3	Clock Divider Register_2.....	223
15.4.4	Control Registers.....	224
15.4.6	Transceiver Control Register	225
15.4.7	TXR Readout register.....	227
15.4.8	CR Read Register	227
16	I2S Controller	229
16.1	Function Overview	229
16.2	Key Features.....	229
16.3	Functional Description.....	229
16.3.1	Multi-mode support.....	229
16.3.2	Zero Cross Detection.....	230
16.3.3	Efficient data transfer	230
16.4	I2S/PCM Timing Diagram	230
16.5	FIFO Storage Structure Diagram	232
16.6	I2S Module Operating Clock Configuration.....	234
16.7	Other function descriptions:	237
16.7.1	Over-zero detection:.....	237
16.7.2	Mute function.....	238

16.7.3	Interruptions.....	238
16.7.4	FIFO Status Query.....	238
16.8	Data Transfer Process.....	239
16.8.1	Master sends audio data.....	239
16.8.2	Receiving audio data from the slave	239
16.8.4	Sending audio data from the slave.....	241
16.8.5	Full Duplex Mode	241
16.9	Register Description.....	242
16.9.1	Register List	242
16.9.2	Control Registers.....	243
16.9.3	Interrupt Mask Register.....	248
16.9.4	Interrupt Flag Register.....	250
16.9.5	Status Register.....	254
16.9.6	Data Transmit Register.....	255
16.9.7	Data Receive Register.....	255
17	UART Module	256
17.1	Function Overview	256
17.2	Main Features.....	256
17.3	Functional Description.....	256
17.3.1	UART Baud Rate.....	256
17.3.2	UART Data Format	257
17.3.3	UART Hardware Flow Control.....	259
17.3.4	UART DMA Transfer.....	259

17.35	UART Interrupt.....	260
17.4	Register Description.....	260
17.4.1	Register List	260
17.4.3	Automatic Hardware Flow Control Register.....	262
17.4.4	DMA Setup Register	263
17.4.5	FIFO Control Register	264
17.4.6	Baud Rate Control Register.....	265
17.4.7	Interrupt Mask Register.....	265
17.4.8	Interrupt Status Register	266
17.4.9	FIFO Status Register	268
17.4.10	TX Starting address register	268
17.4.11	RX Starting address register	269
18	UART&7816 Module.....	270
18.1	Function Overview	270
18.2	Key Features.....	270
18.3	UART Functional Description.....	271
18.4	7816 Functional Description	271
18.4.1	7816 Introduction	271
18.4.2	7816 Interface.....	271
18.4.3	7816 Configuration	272
18.4.4	7816 Clock Configuration	272
18.4.5	7816 Rate Setting	273
18.4.6	7816 Power-on reset	274

184.7 7816 Thermal reset.....	275
18.4.9 7816 Data transmission.....	276
18.4.10 UART&7816 DMA Transmission.....	276
18.4.11 UART&7816 Interrupt.....	277
185 Register Description.....	277
185.1 Register List	277
185.2 Data Flow Control Register	278
185.3 Automatic Hardware Flow Control Register.....	281
185.4 DMA Setup Register	282
185.5 FIFO Control Registers.....	283
185.6 Baud Rate Control Register.....	284
185.7 Interrupt Mask Register.....	285
185.8 Interrupt Status Register.....	285
185.9 FIFO Status Register	287
185.10 TX Starting address register	288
185.11 RX Starting address register.....	288
185.12 7816 Protection Time Register	289
185.13 7816 Timeout Time Register	289
19 Timer Module.....	290
19.1 Functional Overview	290
19.2 Key Features.....	290
19.3 Functional Description.....	290
19.3.2 Delay Function.....	291

194 Register Description.....	291
194.1 Register List	291
194.2 Standard us configuration registers	292
194.3 Timer Control Register	292
194.4 Timer 1 Timing Value Configuration Register.....	294
194.5 Timer 2 Timer Value Configuration Registers.....	294
194.6 Timer 3 Timing Value Configuration Registers.....	294
194.7 Timer 4 Timing Value Configuration Registers.....	294
194.8 Timer 5 Timer Value Configuration Registers.....	295
194.9 Timer 6 Timer Value Configuration Registers.....	295
194.10 Timer 1 Current Count Register	295
194.11 Timer 2 Current Count Register	295
194.12 Timer 3 Current Count Register	295
194.13 Timer 4 Current Count Value Register.....	296
194.14 Timer 5 Current Count Value Register.....	296
194.15 Timer 6 Current Count Value Register.....	296
20 Power Management Modules.....	298
20.1 Function Overview	298
20.2 Main Characteristics	298
20.3 Functional Description.....	298
20.3.2 Low Power Mode	299
20.3.3 Wake-up mode	299
20.3.4 Timer0 Timer.....	300

20.3.5 Real-time clock function	300
20.3.6 32K Clock Source Switching and Calibration.....	300
204 Register Description.....	301
204.1 Register List	301
204.2 PMU Control Register	301
204.3 PMU Timer 0	304
204.4 PMU Interrupt Source Register.....	305
21 Real Time Clock Modules	307
21.1 Function Overview	307
21.2 Main Features.....	307
21.3 Functional Description.....	307
21.3.1 Timing function	307
21.3.2 Timer Function	308
21.4 Register Description.....	308
21.4.1 Register List	308
21.4.2 RTC Configuration Register 1	308
21.4.3 RTC Configuration Register 2	309
22 Watchdog Modules.....	310
22.2 Main Characteristics	310
22.3 Functional Description.....	310
22.3.1 Timer function	310
22.3.2 Reset Function	310
22.4 Register Description.....	311

224.1 Register List	311
224.2 WDG Timer Value Load Register	311
224.3 WDG Current Value Register	312
224.4 WDG Control Register	312
224.5 WDG Interrupt Clear Register	312
224.6 WDG Interrupt Source Register	313
224.7 WDG Interrupt Status Register	313
23 PWM Controller	314
23.1 Functional Overview	314
23.2 Main characteristics	314
23.3 Functional Description	315
23.3.1 Input Signal Capture	315
23.3.2 Number of DMA transfer captures	315
23.3.3 Supports single and automatic loading modes	315
23.3.4 Multiple output modes	315
23.4 Register Description	316
23.4.2 Clock Divider Register_01	317
23.4.3 Clock Divider Register_23	317
23.4.4 Control Registers	318
23.4.5 Cycle Registers	321
23.4.6 Cycle Number Register	323
23.4.7 Comparison Registers	323
23.4.8 Deadband Control Register	325

23.4.9	Interrupt Control Register	326
23.4.10	Interrupt Status Register	327
23.4.11	Channel 0 Capture Register.....	329
23.4.12	Brake Control Register.....	329
23.4.13	Clock Divider Register_4	330
23.4.14	Channel 4 Control Register_1	331
23.4.15	Channel 4 Capture Register.....	333
23.4.16	Channel 4 Control Register_2	334
24	QFLASH Controller.....	338
24.1	Function Overview	338
24.2	Key Features.....	338
24.3	Functional Description.....	338
24.3.1	Bus Access.....	338
24.3.2	Register Access.....	338
24.4	Register Description.....	341
24.4.1	Register List	341
24.4.2	Command Message Register	341
24.4.3	Command start register	342
24.5	Common Instructions for QFLASH	343
25	PSRAM Interface Controller.....	345
25.1	Function Overview	345
25.2	Main Characteristics	345
25.3	Functional Description.....	345

25.3.1	Pin Description	345
25.3.2	Access Mode Settings	346
25.3.3	PSRAM Initialization	346
25.3.4	PSRAM Access Methods	347
25.3.5	BURST Function	347
25.4	Register Description	348
25.4.1	Register List	348
25.4.2	Command Message Register	348
25.4.3	Timeout Control Register	349
26	Touch Sensor	365
26.1	Module Functionality Overview	365
26.2	Functional Usage Instructions	365
26.3	Register list:	366
26.3.1	Touch Sensor Control Register	367
26.3.2	Touch Key Single Control Register	368
26.3.3	Interrupt Control Register	368
27	W800 Security Architecture Design	370
27.1	Function Overview	370
27.1.1	SRAM Security Access Controller (SASC)	370
27.1.2	Trusted IP Controller (TIPC)	371
27.2	Security Architecture Block Diagram	371
27.3	Register Description	371
27.3.1	SASC Register List	372

27.32 TIPC Register.....	380
27.4 Instructions for use	382
27.4.1 Memory Safe Access (SASC)	382
27.4.2 Trusted Access to Peripherals.....	385
28 Appendix 1. Chip Pin Definitions	386
28.1 Chip Pinout	386
28.2 Chip Pin Multiplexing Relationships.....	388
Statement.....	390

catalogo

of

diagrams

ms

Figure 1 W800 Chip Architecture	41
Figure 2 W800 bus structure	43
Figure 3 W800 clock structure	47
Figure 5 System Clock Frequency Division Relationship	57
Fig. 6 Upper unit SPI send/receive data format	131
Figure 7 HSPI register read operation (big-end mode)	131
Figure 8 HSPI Register Write Operation (Big-Ended Mode)	132
Fig. 9 Register read operation (small end mode)	132
Figure 10 Register Write Operation (Small End Mode)	132
Figure 11 Port Read Operation (Big End Mode)	132
Figure 12 Port Write Operation (Big End Mode)	133
Figure 13 Port Read Operation (Small End Mode)	133
Figure 14 Port Write Operation (Small End Mode)	133
Figure 15 CPOL=0,CPHA=0	134
Figure 16 CPOL=0,CPHA=1	134
Figure 17 CPOL=1,CPHA=0	0
Figure 18 CPOL=1,CPHA=1	135
Figure 19 Main SPI Handling Interrupt Flow	136
Figure 20 Downstream data flow diagram	137
Figure 21 Downstream Command Flowchart	138
Figure 22 Uplink Data (Command) Flowchart	139

Figure 23 SDIO Internal Storage Mapping	143
Figure 24 CCCR Register Storage Structure.....	144
Figure 25 FBR1 Register Structure.....	145
Figure 26 CIS Storage Space Structure.....	145
Figure 27 SDIO Receive BD Descriptor.....	169
Figure 28 SDIO Transmit BD Descriptor.....	170
Figure 29 UART Data Length.....	257
Figure 30 UART Stop Bits	258
Figure 31 UART Parity Bits.....	258
Figure 32 UART Hardware Flow Control Connections.....	259
Figure 33 7816 Connection Schematic.....	272
Figure 34 7816 Power-On Reset Timing.....	274
Figure 35 7816 Thermal Reset.....	275
Figure 36 7816 Inactivation process	275
Figure 37 7816 Data Transfer.....	276
Figure 38 W800 Chip Pinout.....	386

table of contents

Table 1 AHB-1 Bus Master Device List.....	44
Table 2 AHB-1 Bus Slave Device List.....	44
Table 3 AHB-2 Bus Master Device List.....	45
Table 4 AHB-2 Bus Slave Device List.....	46
Table 5 Detailed Division of Address Space for Bus Devices.....	49

Table 6 Startup Configuration.....	54
Table 8 Clock Reset Module Register List.....	60
Table 9 Software Clock Gating Enable Registers	60
Table 10 Software Clock Mask Registers.....	64
Table 11 Software Reset Control Registers	65
Table 12 Clock Division Configuration Registers.....	71
Table 13 Clock Selection Registers.....	73
Table 14 I2S Clock Control Registers.....	Table 14 I2S Clock Control Registers
Table 14 Reset Status Registers	76
Table 15 DMA Address Assignments	78
Table 16 DMA Register List	80
Table 17 DMA Interrupt Mask Registers.....	82
Table 18 DMA Interrupt Status Registers	83
Table 19 UART Select Registers.....	84
Table 20 DMA Source Address Registers	85
Table 21 DMA Destination Address Registers.....	Table 21 DMA Destination Address Registers
Table 22 DMA Cycle Source Start Address Registers	85
Table 23 DMA Cycle Destination Start Address Registers.....	86
Table 24 DMA Cycle Length Registers	86
Table 25 DMA Channel Control Registers.....	86
Table 26 DMA Mode Selection Registers	87
Table 27 DMA Data Flow Control Registers.....	88

Table 28 DMA Transfer Byte Count Registers	90
Table 29 DMA Link Table Entry Address Registers	90
Table 30 DMA Current Destination Address Registers	90
Table 31 Encryption Module Register List	95
Table 32 Cryptographic Module Configuration Registers	96
Table 33 TRNG Module Control Registers	99
Table 33 Encryption Module Control Registers	100
Table 34 Encryption Module Status Registers	101
Table 35 RSA Register List	102
Table 36 RSA Data X Register	103
Table 37 RSA Data Y Register	103
Table 38 RSA Data M Register	103
Table 39 RSA Data D Register	104
Table 40 RSA Control Registers	104
Table 41 RSA Parameters MC Register	105
Table 42 RSA Parameters N Register	105
Table 43 GPIOA Register List	107
Table 44 GPIOB Register List	108
Table 45 GPIOA Data Registers	109
Table 46 GPIOB Data Registers	109
Table 47 GPIOA Data Enable Registers	110
Table 48 GPIOB Data Enable Registers	110

Table 49 GPIOA Direction Control Registers.....	110
Table 50 GPIOB Direction Control Registers	111
Table 51 GPIOA Pull-Up Control Registers.....	111
Table 52 GPIOB Pull-Up and Pull-Down Control Registers.....	112
Table 53 GPIOA Multiplexing Selection Registers.....	112
Table 54 GPIOB Multiplexing Selection Registers.....	113
Table 55 GPIOA Multiplexing Select Register 1	114
Table 56 GPIOB Multiplexing Select Register 1	114
Table 57 GPIOA Multiplexing Select Register 0	114
Table 58 GPIOB Multiplexing Select Register 0	115
Table 59 GPIOA Interrupt Trigger Mode Configuration Registers.....	115
Table 60 GPIOB Interrupt Trigger Mode Configuration Registers	115
Table 61 GPIOA Interrupt Edge Trigger Mode Configuration Registers.....	116
Table 62 GPIOB Interrupt Edge Trigger Mode Configuration Registers	116
Table 63 GPIOA Interrupt Top and Bottom Edge Trigger Configuration Registers.....	116
Table 64 GPIOB Interrupt Top and Bottom Edge Trigger Configuration Registers.....	117
Table 65 GPIOA Interrupt Enable Configuration Registers	117
Table 66 GPIOB Interrupt Enable Configuration Registers.....	117
Table 67 GPIOA Bare Interrupt Status Registers	118
Table 68 GPIOB Bare Interrupt Status Registers	118
Table 69 GPIOA Interrupt Status Register after Masking.....	118
Table 70 GPIOB Interrupt Status Register after Masking	118

Table 71 GPIOA Interrupt Clear Control Registers	119
Table 72 GPIOB Interrupt Clear Control Registers	119
Table 73 HSPI Internal Access Registers	121
Table 74 HSPI FIFO Flush Registers	122
Table 75 HSPI Configuration Registers	123
Table 76 HSPI Mode Configuration Registers	123
Table 77 HSPI Interrupt Configuration Registers	124
Table 78 HSPI Interrupt Status Registers	124
Table 79 HSPI Data Upload Length Registers	125
Table 80 HSPI Interface Configuration Registers (Master Device Access)	125
Table 81 HSPI Get Data Length Registers	127
Table 82 HSPI Downstream Data Flag Registers	127
Table 83 HSPI Interrupt Configuration Registers	128
Table 84 HSPI Interrupt Status Registers	128
Table 85 HSPI Data Ports 0	128
Table 86 HSPI Data Ports 1	129
Table 87 HSPI Command Ports 0	129
Table 88 HSPI Command Ports 1	130
Table 89 List of SDIO CCCR Registers and FBR1 Registers	145
Table 90 SDIO Fn1 Address Mapping Relationships	157
Table 91 SDIO Fn1 partial registers (for HOST access)	158
Table 92 SDIO AHB Bus Registers	160

Table 93 WRAPPER Controller Registers	170
Table 94 WRAPPER Interrupt Status Registers	172
Table 95 WRAPPER Interrupt Configuration Registers	172
Table 96 WRAPPER Uplink Command Ready Registers	172
Table 97 WRAPPER Downstream C o m m a n d buf Ready Registers	173
Table 98 SDIO TX Link Enable Registers	173
Table 99 SDIO TX Link Address Registers	174
Table 100 SDIO TX Enable Registers	174
Table 101 SDIO TX Status Registers	174
Table 102 SDIO RX Link Enable Registers	175
Table 103 SDIO RX Link Address Registers	175
Table 104 SDIO RX Enable Registers	176
Table 105 SDIO RX Status Registers	176
Table 106 WRAPPER CMD BUF Base Address Registers	177
Table 107 WRAPPER CMD BUF SIZE registers	177
Table 108 SPI Register List	201
Table 109 SPI Channel Configuration Registers	202
Table 110 SPI Configuration Registers	206
Table 111 SPI Clock Configuration Registers	209
Table 112 SPI Mode Configuration Registers	210
Table 113 SPI Interrupt Control Registers	211
Table 114 SPI Interrupt Status Registers	213

Table 115 SPI Status Register.....	215
Table 116 SPI Timeout Register.....	216
Table 117 SPI Data Transmission Registers	216
Table 118 SPI Transfer Mode Registers	217
Table 119 SPI Data Length Register	219
Table 120 SPI Data Receive Registers.....	220
Table 121 I2C Register List.....	222
Table 122 I2C Clock Division Register_1.....	223
Table 123 I2C Clock Divider Register_2	223
Table 124 I2C Control Registers	224
Table 125 I2C Data Registers.....	224
Table 126 I2C Transceiver Control Registers.....	225
Table 127 I2C TXR Readout Registers	227
Table 128 I2C CR Readout Registers	227
Table 129 I2S Register List.....	242
Table 130 I2S Control Registers	243
Table 131 I2S Interrupt Mask Registers	248
Table 132 I2S Interrupt Flag Registers.....	250
Table 133 I2S Status Registers.....	254
Table 134 I2S Data Transmission Registers.....	255
Table 135 I2S Data Receive Registers	255
Table 136 UART Register List.....	260

Table 137 UART Data Flow Control Registers	261
Table 138 UART Automatic Hardware Flow Control Registers	262
Table 139 UART DMA Setup Registers.....	263
Table 140 UART FIFO Control Registers	264
Table 141 UART Baud Rate Control Registers.....	265
Table 142 UART Interrupt Mask Registers.....	265
Table 143 UART Interrupt Status Registers	266
Table 144 UART FIFO Status Registers.....	268
Table 145 UART TX start address registers	268
Table 146 UART RX start address registers.....	269
Table 147 7816 Rate Settings	273
Table 148 UART&7816 Register List.....	277
Table 149 UART&7816 Data Flow Control Registers.....	278
Table 150 UART&7816 Automatic Hardware Flow Control Registers.....	281
Table 151 UART&7816 DMA Setup Registers.....	282
Table 152 UART&7816 FIFO Control Registers.....	283
Table 153 UART&7816 Baud Rate Control Registers	284
Table 154 UART&7816 Interrupt Mask Registers.....	285
Table 155 UART&7816 Interrupt Status Registers.....	285
Table 156 UART&7816 FIFO Status Registers	287
Table 157 UART&7816 TX start address registers	288
Table 158 UART&7816 RX start address registers	288

Table 159 7816 Protection Time Registers	289
Table 160 7816 Timeout Time Registers	289
Table 161 Timer Register List.....	291
Table 162 Timer Standard us Configuration Registers.....	292
Table 163 Timer Timer Control Registers	292
Table 164 Timer 1 Timer Value Configuration Registers	294
Table 165 Timer 2 Timer Value Configuration Registers	294
Table 166 Timer 3 Timer Value Configuration Registers	294
Table 167 Timer 4 Timer Value Configuration Registers	294
Table 168 Timer 5 Timer Value Configuration Registers.....	295 Timer Value Configuration Register
Table 169 Timer 6 Timer Value Configuration Registers	295 Timer 6 Timer Value Configuration Register
Table 170 PMU Register List.....	301
Table 171 PMU Control Registers.....	301
Table 172 PMU T i m e r 0 Registers.....	304
Table 173 PMU Interrupt Source Registers.....	305
Table 174 RTC Register List.....	308
Table 175 RTC Configuration Register 1	308
Table 176 RTC Configuration Register 2	309
Table 177 WDG Register List	311
Table 178 WDG Timing Value Load Registers.....	311
Table 179 WDG Current Value Registers.....	312
Table 180 WDG Control Registers	312

Table 181 WDG Interrupt Clear Registers.....	312
Table 182 WDG Interrupt Source Registers	313
Table 183 WDG Interrupt Status Registers.....	313
Table 184 PWM Register List	316
Table 185 PWM Clock Division Register_01	317
Table 186 PWM Clock Divider Register_23.....	317
Table 187 PWM Control Registers.....	318
Table 188 PWM Cycle Registers.....	321
Table 189 PWM Cycle Number Registers.....	323
Table 190 PWM Comparison Registers	323
Table 191 PWM Deadband Control Registers.....	325
Table 192 PWM Interrupt Control Registers	326
Table 193 PWM Interrupt Status Registers	327
Table 194 PWM Channel 0 Capture Registers	329
Table 195 PWM Brake Control Registers	329
Table 196 PWM Clock Division Register_4	330
Table 197 PWM Channel 4 Control Register_1	331
Table 198 PWM C h a n n e l _ 4 Capture Registers.....	333
Table 199 PWM Channel 4 Control Register_2.....	334
Table 200 QFLASH Controller Register List.....	341
Table 201 QFLASH Command Message Store	341
Table 202 QFLASH Command Startup Registers.....	342

Table 203 QFALSH Frequently Used Commands.....	343
Table 200 PSRAM Controller Register List.....	348
Table 201 PSRAM Control Setup Registers.....	348
Table 201 CS Timeout Control Register.....	349
Table 200 Touch Sensor Controller Register List.....	366
Table 201 Touch Sensor Control Setting Registers.....	367
Table 201 Touch Key Single Setting Registers.....	368
Table 201 Touch Key Interrupt Control Registers.....	368
Table 204 Chip Pin-Multiplexing Relationships.....	388

1

INTRO

DUCTI Purpose of preparation

ON

W800 is an embedded Wi-Fi SoC chip from UniSolar Microelectronics. It is highly integrated, requires fewer peripheral devices, and is cost-effective. It is suitable for all kinds of smart products in IoT (Smart Home) field. Highly integrated Wi-Fi and Bluetooth 4.2 Combo is its main function; in addition, the chip integrates XT804 core, built-in QFlash, SDIO, SPI, UART, GPIO, I²C, PWM, I²S, 7816, LCD, Touch Sensor and other interfaces, and supports a variety of hardware encryption and decryption algorithms. In addition, the chip MCU contains a security kernel that supports code security authority settings, system-wide support for firmware encryption storage, firmware signing, secure debugging, secure upgrading and other security measures to enhance product security features.

This document mainly describes the internal structure of the W800 chip, the information of each function module and the detailed register usage information; it is the main reference material for developers to develop drivers and applications. There are open source implementations of various functions in the SDK provided by UniSolar Microelectronics. Developers can refer to the corresponding driver and application samples to increase their understanding of the chip's functions and register descriptions. There is no register description for the Wi-Fi/BT part in this document.

1.2 bibliography

Please refer to the "W800 Chip Datasheet" for information on the W800 chip package parameters, electrical characteristics, and RF parameters;

The W800 chip integrates a ROM program, which provides functions such as downloading firmware, reading and writing MAC address, reading and writing Wi-Fi parameters, etc. For detailed information,

1

INTRO refer to "WM_W800_ROM Functions";

DUCTI

ON

The W800 chip has a built-in 2Mbytes QFlash memory for code and parameter storage. This document provides basic QFlash operation information. For needs beyond the scope of this document, you need to refer to the QFlash manual;

W800 chip adopts Hangzhou Pingtou Brother XT804 core804 related to the introduction of the function, development information, etc. can refer to Pingtou Brother company released information;

For more information, please refer to the website of UniSolar Microelectronics (<http://www.winnermicro.com/>).

2 characterization

- chip package

- QFN32 package, 4mm x 4mm.

- Chip integration

- Integrated XT804 processor up to 240MHz
 - Integrated 288KB SRAM
 - Integrated 2MB FLASH
 - Integrated 8-channel DMA controller, supports 16 hardware requests, supports software chain table management
 - Integrated PA/LNA/TR-Switch
 - Integrated 32.768KHz clock oscillator
 - Integrated Voltage Detection Circuitry
 - Integrated LDOs
 - Integrated power-on reset circuit

- Chip Interface

- Integrated 1 SDIO2.0 Device Controller, supports SDIO 1-bit/4-bit/SPI three modes of operation,

operating clock range 0~50MHz

- Integrated 1 SDIO 2.0 HOST controller, support SDIO and SD card operation, working clock range 0~50MHz
- Integrated 1 QSPI PSRAM interface, support up to 64MB PSRAM capacity, maximum operating clock frequency 80MHz;
- Integrated 5 UART interfaces, support RTS/CTS, baud rate range 1200bps~2Mbps
- Integrated 1 high-speed SPI slave interface, operating clock range 0~50MHz
- Integrated 1 SPI master/slave interface, master device operates at up to 20MHz, slave device supports up to 6Mbps data transfer rate
- Integrated an I2C controller supporting 100/400Kbps rates
- Integrated PWM controller, supports 5 PWMs
 - Individual outputs or 2 PWM inputs. Maximum output frequency 20MHz, maximum input frequency 20MHz.
- Integrated duplex I2S controller supporting 32KHz to 192KHz I2S interface codecs
- Integrated 1 7816 interface, UART compatible interface supports ISO-7816-3 T=0.T=1 mode; supports EVM2000 protocol
- Support multiple hardware encryption and decryption modes,
 - including RSA/AES/RC4/DES/3DES/RC4/SHA1/MD5/CRC8/CRC16/CRC32/TRNG
- Integrated 1 differential, or 2 single-ended 16bit ADC interfaces;
- Integrated 11-way Touch Sensor;
- Supports up to 17 GPIO ports, each with
 - rich multiplexing relationships. Input and
 - output configuration options are
 - available.

● WIFI protocol and features

- Integrated 1 SDIO 2.0 HOST controller, support SDIO and SD card operation,
work Support Gigabit LAN, IEEE802.11 b/g/n;
- Support WMM/WMM-PS/WPA/WPA2/WPS
- WiFi Direct support;

- Supports the EDCA channel access method;
 - Supports 20/40M bandwidth operating mode;
 - Supports STBC, GreenField, Short-GI, and reverse transfer;
 - RIFS frame spacing is supported;
 - AMPDU, AMSDU are supported;
 - Supports 802.11n MCS 0~7, MCS32 Physical Layer transmission rate slots up to 150Mbps.
 - HT-immediate Compressed BlockAck, normal ACK, and no ACK response methods are supported;
 - Support CTS to self;
 - Supports AP function; AP and STA are used simultaneously;
 - In the BSS network, multiple multicast networks are supported, and different encryption methods are supported for each multicast network, up to a total of 32 multicast networks and incoming STA encryption;
 - The BSS network supports a total of 32 sites and groups when used as an AP;
 - Reception sensitivity:
 - 20MHz MCS7@-71dBm@10%PER;
 - 40MHz MCS7@-67dBm@10%PER;
 - 54Mbps@-73dBm@10%PER;
 - 11Mbps@-86dBm@8%PER;
 - 1Mbps@-96dBm@8%PER;
 - A number of different receive frame filtering options are supported;
 - Supports the listening function;
-
- Bluetooth Protocol and Features

- Integrated Bluetooth baseband processor/co-processor with BT/BLE4.2 protocol support
 - Supports DR/EDR at various rates;
 - Supports BLE 1Mbps rates;
-
- Power supply and power consumption
 - 3.3V single power supply;
 - Supports Wi-Fi power saving mode power management;
 - Supports work, sleep, standby, and shutdown operating modes;
 - Standby power consumption is less than 15uA;

3

Overview

This chip is a multi-interface, multi-protocol wireless LAN 802.11n (1T1R) SOC chip. The SOC integrates RF Transceiver, CMOS PA Power Amplifier, Baseband Processor/Media Access Control, SDIO, SPI, UART, GPIO and other interfaces for low power WLAN.

W800 supports GB15629.11-2006, IEEE802.11 b/g/n protocols, and STBC, Green Field, Short-GI, Reverse Transmission, RIFS Frame Interval, AMPDU, AMSDU, T-immediate Compressed Block Ack, normal ACK, no ACK, CTS to self, and other rich protocols and operations.

The W800 chip integrates an RF transceiver front-end, A/D and D/A converters on-chip. It supports DSSS (Direct Sequence Spread Spectrum) and OFDM.

(The W800 chip also includes a built-in enhanced signal monitor that largely eliminates the effects of multipath effects. The transceiver's analog front-end is equipped with a transceiver AGC function for optimal system-on-chip performance, and the W800 chip includes a built-in signal booster monitor that largely eliminates the effects of multipath.

In terms of security, the W800 chip supports not only the national standard WAPI encryption, but also the international standard WEP, TKIP, and CCMP encryption. These hardware components enable the data transmission system based on the chip to obtain similar data transmission performance as non-encrypted communication when carrying out confidential communication.

In addition to supporting the energy-saving operations stipulated by IEEE802.11 and Wi-Fi protocols, the W800 chip also supports user-defined energy-saving schemes. The chip supports four operating modes: work, sleep, standby, and shutdown, which enables the whole system to realize low

3

Power consumption and makes it easy for users to define different energy-saving schemes according to their own usage scenarios.

The W800 chip integrates a high-performance 32-bit embedded processor, a large amount of memory resources, and a wealth of peripheral interfaces, making it easy to use. It is easy to apply the chip to secondary development work for specific products.

The W800 chip supports AP function, which can realize the simultaneous formation of 5 SSID networks and the function of 5 independent APs. Support the function of establishing multicast network. It can realize the function of joining other networks as STA and establishing BSS network as AP at the same time.

The W800 chip supports the WPS method, which allows users to realize the encrypted full network with one-button operation to ensure the security of information.

The versatility and high integration of the W800 chip ensures that WLAN systems do not require excessive off-chip circuitry and external memory.

4 chip structure

4.1 chip structure

The following figure describes the overall structure of the W800 chip. The core part includes the XT804 CPU, 288KB SRAM and 20KB ROM memory. The PMU part, as the constant power supply module of the chip, provides power-on timing management, start-up clock, real-time clock function, and so on. Provides rich peripherals

The Wi-Fi section integrates MAC, BB, and RF.

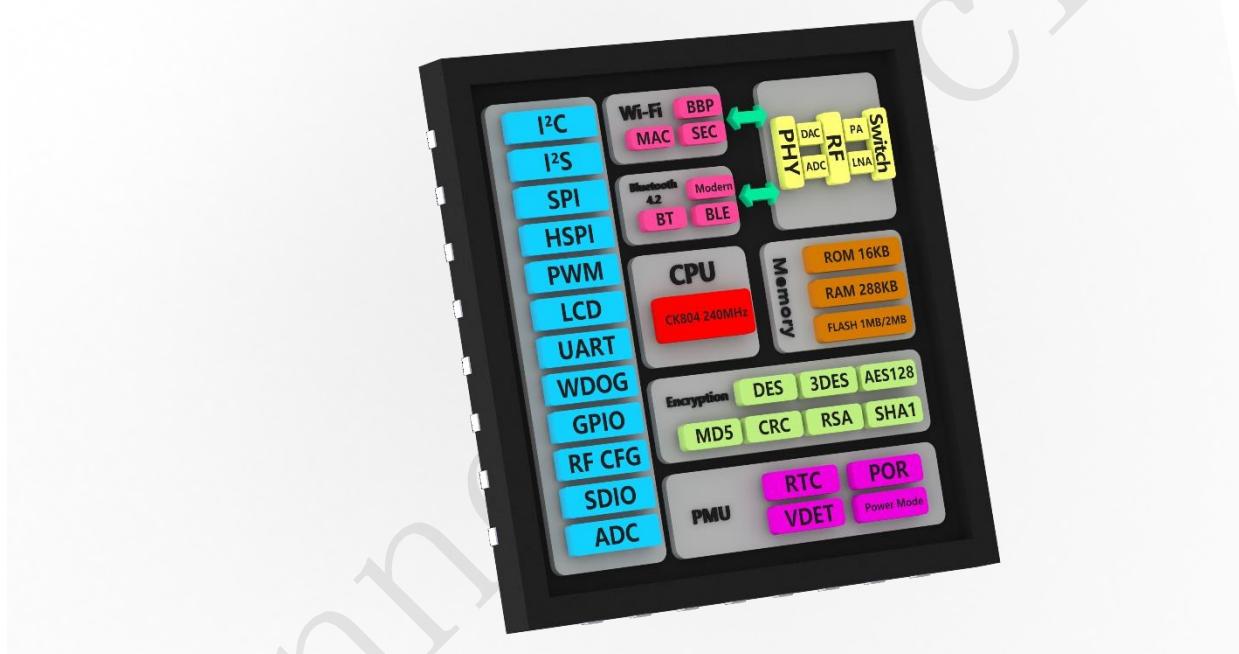


Figure 1 W800 Chip
Structure

Winner Micro

4.2 bus structure

The W800 chip consists of two levels of buses as shown below

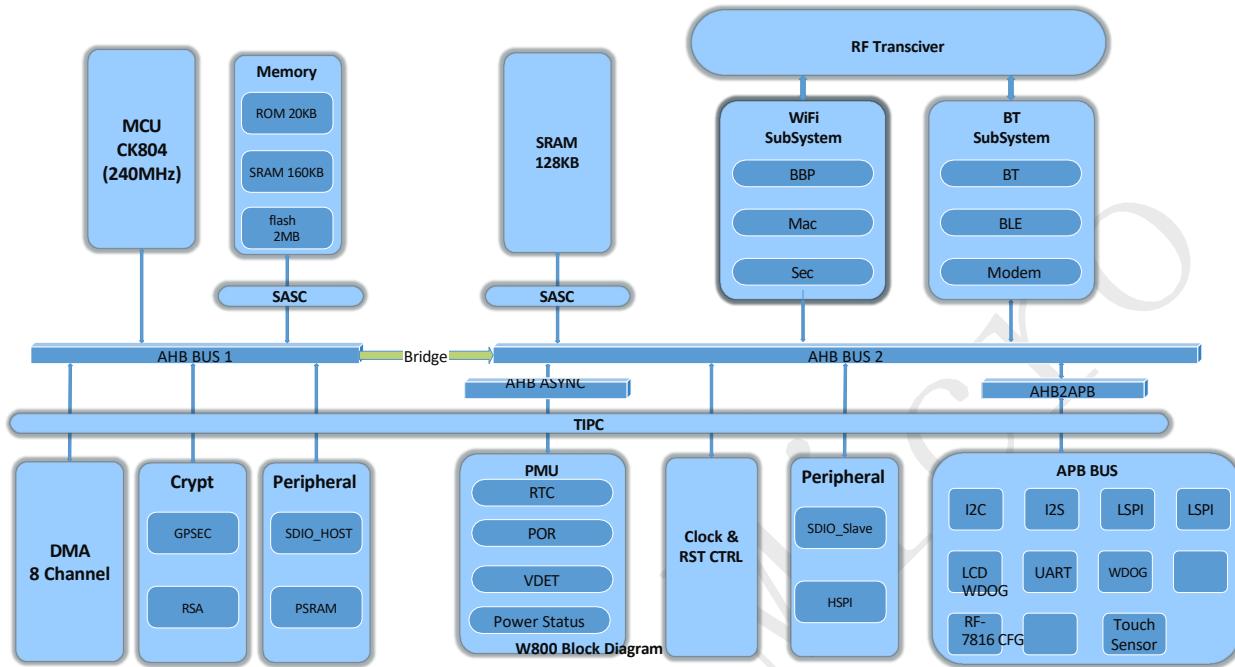


Figure 2 W800 Bus Structure

(1) AHB-1 Bus

This stage of the bus has four master devices-i.e., XT804, DMA, GPSEC-and five slave devices.

The XT804 is a 32-bit power-efficient embedded CPU core for control applications. It utilizes a 16/32-bit mixed-code instruction system and is designed with a lean and efficient 3-stage pipeline.

The XT804 provides a variety of configurable features, including hardware floating-point unit, on-chip cache, DSP acceleration unit, trusted protection technology, on-chip tightly coupled IP, etc., which can be configured according to the application needs. In addition, the XT804 provides a multi-bus interface that supports flexible configuration of the system bus, instruction bus, and data bus, and the XT804 has special acceleration for interrupt response, which requires only 13 cycles of interrupt response delay. The bus clock operates at 240MHz at the fastest and can be

configured to 240/160/120/80/40MHz or lower.

Table 1 AHB-1 Bus Master Device List

main equipment	functionality
CPU	Complete chip register configuration, memory management and usage, and full 802.11MAC protocol. Up to Running frequency 240MHz
DMA	Stand-alone 8-channel DMA module supporting a chained table structure with 16 on-chip hardware DMA request sources.
GPSEC	Universal encryption module, supports DES/3DESSHA1/AES/MD5/RC4/CRC/PRDN. auto-complete. Encrypts and writes back a block of data in the specified memory space.

Table 2 AHB-1 Bus Slave Device List

slave equipment	functionality
ROM	ROM is used to store the initialization firmware after the CPU is powered up. It mainly accomplishes the initial configuration of the chip register space. After completing the above work, the CPU control is handed over to the firmware stored in FLASH.
AHB2AHB	Completes the conversion of the CPU bus clock domain to the BusMatrix2 bus clock domain for master device access. Requires that the clock domains must be homologated and that the ratio of the CPU clock to the BusMatrix2 clock frequency is M:1, M

	is an integer greater than or equal to 1.
flash	Stores firmware code as well as operating parameters.
SRAM 160KB	It can be used to store instructions or data, and the firmware can use this memory as needed.
RSA	Supports RSA encryption and decryption operations up to 2048bit.

GPSEC	General-purpose encryption/decryption module supporting SHA1/AES/MD5/RC4/CRC/TRNG. auto-completion finger Encrypt/decrypt and write back a block of data in a fixed memory space.
SDIO_HOST	SDIO HOST controller for the SDIO 2.0 standard; SDIO interface peripherals can be accessed through this interface. The SDIO interface clock is divided by the bus clock and supports up to 50MHz.
PSRAM_CTRL	QSPI interface PSRAM controller. External PSRAM can be accessed through this controller. The port clock is divided by the bus clock and supports up to 80MHz clock.

(2) AHB-2 Bus

This bus has 4 masters and 3 slaves, and uses a crossbar connection structure to enable simultaneous access from different masters to different slaves, thus increasing the bandwidth. The bus clock operates at a maximum frequency of 40MHz and can be configured to lower frequencies if required.

Table 3 AHB-2 Bus Master Device List

main equipment	functionality
MAC	802.11MAC control protocol processing module. The main operations on the bus include sending reads and sending reads. data, receiving write data, and sending write-backs of completion descriptors, among other operations.
SEC	The security module accomplishes encryption, decryption, and migration of sent and received data. When transmitting, the transmit data and MAC descriptor are moved to the specified location and

	<p>encryption is completed; when receiving, the receive data and the MAC descriptor are moved to the specified location and encryption is completed.</p> <p>The MAC receive descriptor is moved to the specified location and decryption is completed.</p>
AHB2AHB	Conversion of the AHB-1 bus to the AHB-2 bus for bus master device access.
SDIO/HSPI	<p>The host will be able to access the chip via the SDIO2.0 device controller or high-speed SPI from the device controller.</p> <p>Accesses are converted to AHB bus signals and access to content memory and register space.</p>

Each master device has a fixed priority, with decreasing priority from top to bottom.

Table 4 AHB-2 Bus Slave Device List

slave equipment	functionality
SRAM 128KB	Used to store the uplink and downlink data buffer, the SDIO/SPI/UART interfaces use this RAM for data cache
Configuration	Register configuration space, where the high-speed module configuration registers are uniformly addressed.
APB	All low-speed modules access the space and the various low-speed modules are connected using the APB bus.
BT_CORE	Bluetooth controller.

4.3 clock structure

The W800 uses 24/40MHz crystals as the SoC clock source, and has a built-in DPLL output of 480MHz, which is used by the CPU, system bus, data bus, and WiFi system, and a 32.768KHZ RC oscillator, which is used by the PMU and LCD module. The clock structure is summarized in the following figure.

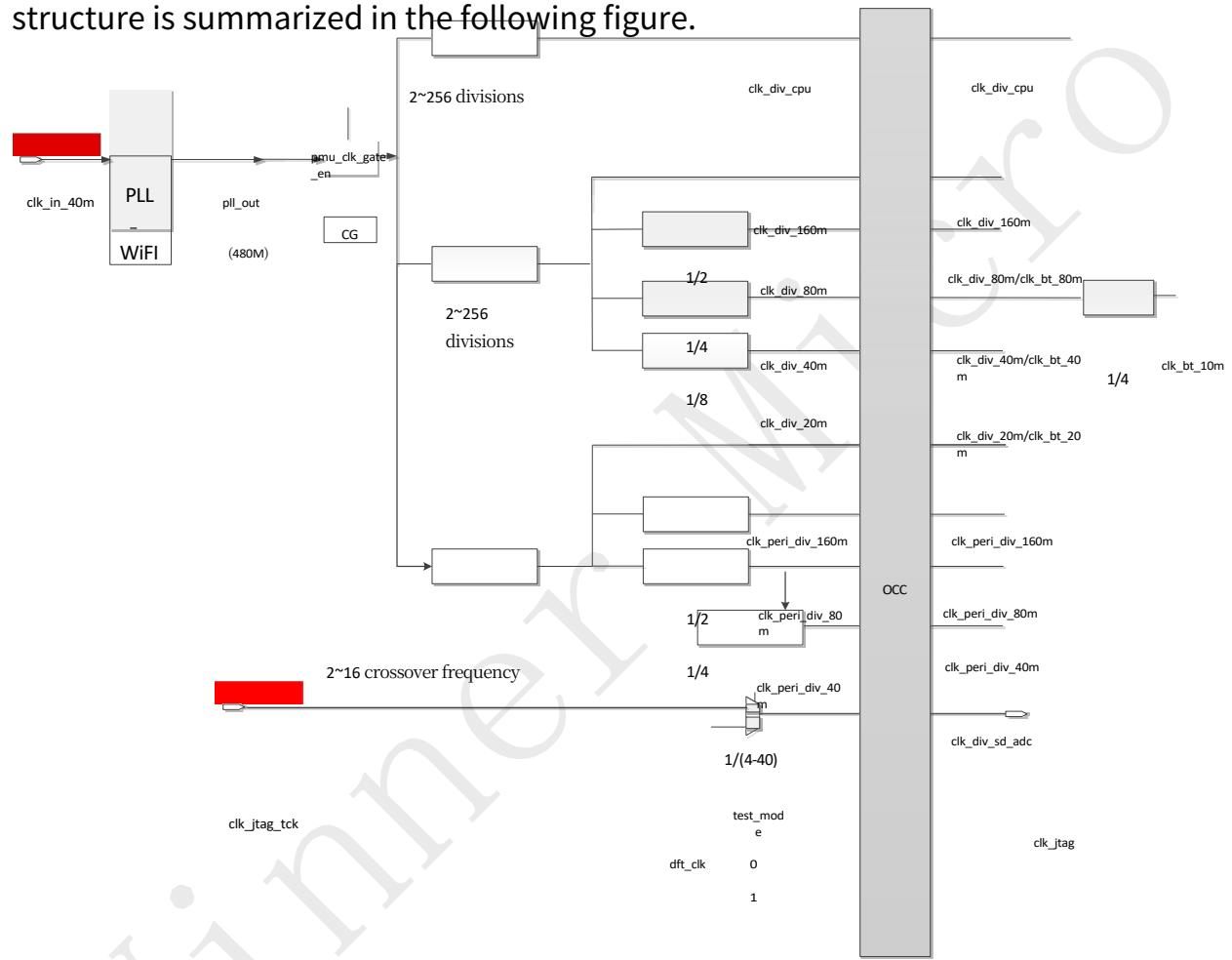
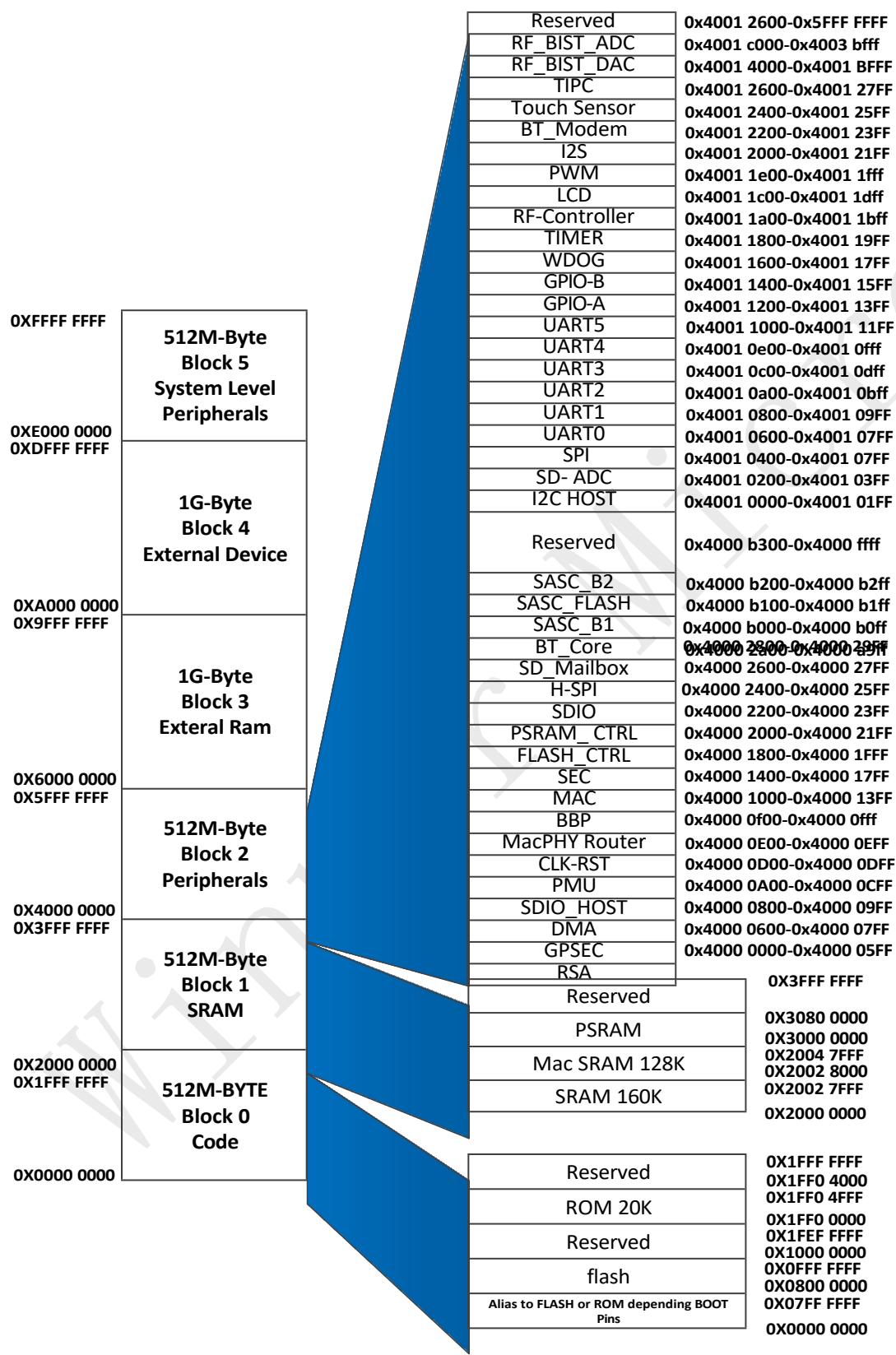


Figure 3 W800 clock structure

4.4 address space



W800 Address

Mapping

The XT804 supports 4G memory space, which is divided into 6 blocks as shown in the above figure, namely code area, memory area, on-chip peripheral, off-chip memory area, off-chip peripheral and system peripheral area. According to the requirement, the w800 on-chip storage space is mapped to the first three zones as shown in Figure III.

Table 5 Detailed Division of Address Space for Bus Devices

Bus installations	BootMode=0	address space subdivision	note
ROM	0x0000 0000 ~ 0x0004 FFFF		Stores solidified firmware code
flash	0x0800 0000 ~ 0x0FFF FFFF		As a dedicated instruction store Ware.
SRAM	0x2000 0000 ~ 0x2002 7FFF		Firmware memory and instruction storage surname Ou
Mac RAM	0x2002 8000 ~ 0x2004 7FFF		SDIO/H-SPI/UART data cache
PSRAM	0x3000 000~0x30800000		Peripheral Memory
CONFIG	0x4000 0000 ~ 0x4000 2FFF	0x4000 0000~ 0x4000 05FF	RSA Configuration Space
		0x4000 0600~ 0x4000 07FF	GPSEC Configuration Space

	0x4000 0800~ 0x4000 09FF	DMA Configuration Space
	0x4000 0A00~ 0x4000 0CFF	SDIO_HOST Configuration empty section of a room or lateral space between two pairs of pillars
	0x4000 0D00~ 0x4000 0DFF	PMU Configuration Space

	0x4000 0E00~ 0x4000 0EFF	Clock and Reset Configuration spatial
	0x4000 0F00~ 0x4000 0FFF	MacPHY Router with installation space
	0x4000 1000~ 0x4000 13FF	BBP Configuration Space
	0x4000 1400~ 0x4000 17FF	MAC Configuration Space
	0x4000 1800~ 0x4000 1FFF	SEC Configuration Space
	0x4000 2000~ 0x4000 21FF	FLASH Controller Configuration space
	0x4000 2200~ 0x4000 23FF	PSRAM_CTRL Configuration spatial
	0x4000 2400~ 0x4000 25FF	SDIO Slave Configuration Empty section of a room or lateral space between two pairs of pillars
	0x4000 2600~ 0x4000 27FF	H-SPI Configuration Space
	0x4000 2800~ 0x4000 29FF	SD Wrapper Configuration space
	0x4000 2A00~ 0x4000 A9FF	BT Core Configuration Space

	0x4000 B000~ 0x4000 B0FF	SASC-B1 First-level bus memory security configuration module
	0x4000 B100~ 0x4000 B1FF	SASC-Flash Flash Security Configuration Module

		0x4000 B200~ 0x4000 B2FF	SASC-B2 Secondary Bus Memory Security Configuration Module
APB	0x4001 0000~ 0x 4001 C000	0x4001 0000~ 0x4001 01FF	I2C master
		0x4001 0200~ 0x4001 03FF	Sigma ADC
		0x4001 0400~ 0x4001 07FF	SPI master
		0x4001 0600~ 0x4001 07FF	UART0
		0x4001 0800~ 0x4001 09FF	UART1
		0x4001 0A00~ 0x4001 0BFF	UART2
		0x4001 0C00~ 0x4001 0DFF	UART3
		0x4001 0E00~ 0x4001 0FFF	UART4
		0x4001 1000~ 0x4001 11FF	UART5
		0x4001 1200~ 0x4001 13FF	GPIO-A
		0x4001 1400~ 0x4001 15FF	GPIO-B
		0x4001 1600~ 0x4001 17FF	WatchDog
		0x4001 1800~ 0x4001 19FF	Timer
		0x4001 1A00~ 0x4001 1BFF	RF_Controller
		0x4001 1C00~ 0x4001 1DFF	LCD

	0x4001 1E00~ 0x4001 1FFF	PWM
--	--------------------------	-----

	0x4001 2000~ 0x4001 22FF	I2S
	0x4001 2200~ 0x4001 23FF	BT-modem
	0x4001 2400~ 0x4001 25FF	Touch Sensor
	0x4001 2600~ 0x4001 25FF	TIPC Interface Security Settings
	0x4001 4000~ 0x4000 BFFF	RF_BIST DAC Transmit random access memory (RAM)
	0x4001 C000~ 0x4003 BFFF	RF_BIST ADC Receive random access memory (RAM)
	0x4001 3C00~ 0x5FFF FFFF	RSV

4.4.1 SRAM

The W800 has a built-in 288KB of SRAM, of which 160KB is mounted on the primary AHB bus and 128KB on the secondary AHB bus. Primary bus devices such as the CPU can access all memory areas, but devices on the secondary bus can only access 128KB of memory on the secondary bus.

4.4.2 Flash

4.4.2.1 QFlash

W800 integrates 2MBytes of QFlash inside the chip, and the 32KB cache inside the chip realizes the XIP method of executing the program on QFlash. During program execution, the CPU first reads the instruction from the Cache, and when it cannot get the instruction, it reads the instruction from the QFlash in 8Bytes rows and stores it in the Cache. Therefore, when the size of continuously running code is less than 32K, the CPU does not need to read instructions from QFlash, and then the CPU can run at a higher frequency. The above method is a read instruction operation method, and the entire RO section of the image is operated in this way. There is no need for user intervention in this process.

QFlash can also store data. When the user program needs to read or write the data in QFlash, it needs to operate through the built-in QFlash controller, which provides the corresponding address, instruction and other registers to help realize the operation that the user wants. Please refer to the corresponding chapter of QFlash controller for detailed description.

Users should note that when the program reads or writes data, there is no need to judge the status, wait, etc., because the QFlash controller itself will judge it. When the QFlash controller returns, it

indicates that the reading or writing has been completed.

4.4.2.2 SPI Flash

In addition to the 6PIN QFlash interface (built-in PIN, unpackaged) the W800 chip also supports low-speed SPI interface access. The SPI interface can operate at a maximum frequency of 20MHz and supports master-slave functionality, as described in detail in the SPI Interface chapter.

4.4.3

PSRAM

The W800 has a built-in SPI/QSPI PSRAM controller, which supports external PSRAM devices with a maximum capacity of 64Mb, and provides PSRAM read, write, and erase operations in bus mode, with a maximum read/write speed of 80MHz. The maximum read/write speed is 80MHz, and when the storage capacity needs to be expanded, off-chip PSRAM can be used to expand the code storage space or data storage space, PSRAM also supports XIP execution of programs, and CPU Cache also supports caching of data in PSRAM.

4.5 Startup Configuration

When the W800 chip is powered on, the CPU will start to execute the firmware in the ROM and load the user image at the specified address in the Flash. When the ROM firmware starts to run, it will read the BootMode (PA0) pin and enter the boot state according to the signal of the pin:

Table 6 Startup Configuration

BootMode	priming conditions	activation mode
your (honorific)		Normal startup process
lower (one's head)	Duration <30ms, not valid in fast test mode	Normal startup process
	Duration >= 30ms	Entering Function Mode

4.4.3

PSRAM

Notes:

Test Mode: Chip test function, not user operable.

Function Mode: Enter the basic functions realized by ROM, such as: download firmware, burn MAC address, etc., detailed information refer to

WM_W800_ROM Functional
Brief.pdf

Typically, the BootMode pin should be used for production or debugging. In the production phase, users can quickly burn Flash by pulling the BootMode pin low for more than 30ms continuously to enter the function mode.

In rework or repair scenarios, if the chip is not at the "highest security level" (for a description of the security level, see

When the "WM_W800_ROM Function Summary" is displayed, you can use this pin to enter the function mode to erase the old image and write a new one.

During the debugging phase, regardless of any faults in the firmware, it is possible to burn new firmware by pulling the BootMode pin continuously low for more than 30ms to access the serial port download function.

5 Clock and Reset Module

5.1 Functional overview

The clock and reset module accomplishes the software control of the chip clock and reset system.

Clock control includes clock frequency conversion, clock shutdown and adaptive gating; reset

control includes soft reset control of the system and sub-modules.

5.2 Main characteristics

- Supports clock shutdown for each module
- Supports adaptive shutdown of some module clocks
- Supports software reset for each module
- Supports CPU frequency setting
- Supports ADC/DAC loopback testing
- Supports I2S clock settings

5.3 Functional Description

5.3.1 Clock gating

By configuring the clock gating enable register CLK_GATE_EN you can control the clock shutdown of a specified function to shut down a module function.

In order to provide the firmware with flexibility in controlling the power consumption of the system, the clock and reset module provides clock gating for each module within the system. When the clock of the corresponding module is turned off, the digital logic and clock tree of that module will

5 Clock and

Reset Module can reduce the dynamic power consumption of the system.

Detailed description of the switch corresponding to register SW_CLKG_EN for each specific module.

5.3.2 Clock Adaptive Shutdown

The chip adaptively shuts down the clock of certain function modules based on the migration of certain internal states. Users are advised not to change the configuration as this may cause system abnormalities when configuring PMU functions.

5.3.3 Function reset

The chip provides the soft reset function for each subsystem, and the subsystem can be reset by setting the corresponding BIT of SW_RST_CTRL to 0. However, the reset state will not be cleared automatically. However, the reset state will not be cleared automatically, and the corresponding BIT of SW_RST_CTRL needs to be set to 1 in order to resume normal operation.

The soft reset function does not reset the CPU and WatchDog.

Detailed description of the switch corresponding to register SW_CLKG_EN for each specific module.
 The reset operation of APB/BUS1/BUS2 (corresponding to the APB bus, system bus, and data bus) in this register is not recommended and can cause the system to

Unified access device exception.

5.3.4 clock crossover

The W800 system uses a 40MHz/24MHz crystal as the system clock source, and the system has a built-in DPLL that outputs a fixed 480MHz clock as the system clock source.

System-wide clock source (below)

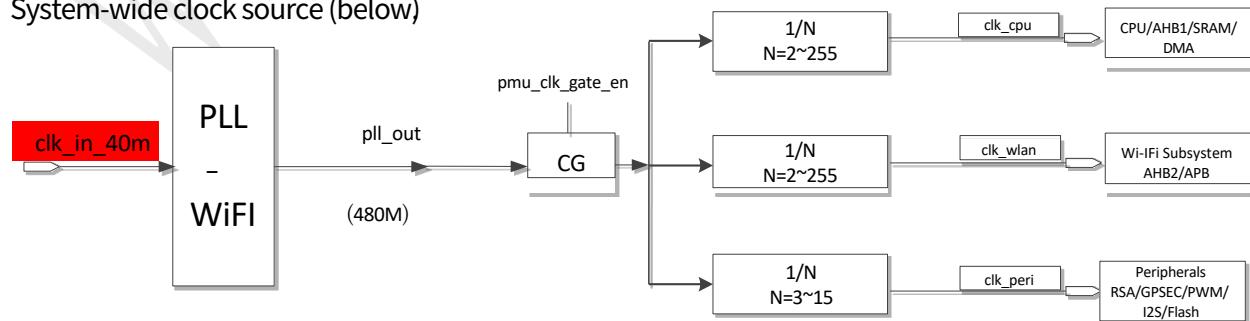


Fig. 4 System clock frequency division relationship

The system bus clock is the same as the CPU clock, and the data bus clock is fixed to 1/4 of the WLAN root clock.

The WLAN root clock is also the clock source for the entire WLAN system.

This module provides the function to set the CPU clock and WLAN root clock for firmware to adjust the system performance and power consumption.

Setting BIT[7:0] of SYS_CLK_DIV register can adjust the CPU clock dividing factor, the source clock of CPU clock dividing is the output of DPLL, which is fixed at 480MHz, and the default value of CPU clock dividing factor is 6, i.e., the default operating frequency of CPU is 6 divisions of 480MHz, i.e., 80MHz, and you can reconfigure this parameter when you need to adjust the required clock of CPU. When you need to adjust the CPU clock, you can reconfigure this parameter.

The CLK_PERI clock provides the root clock for the running clock of the cryptographic module in the SoC system, as well as the root clock for the running clocks of certain interfaces, such as the PWM interface, the I2S interface, and the Flash interface clock. This clock is also derived from the 480MHz frequency division of the DPLL output. Under normal operation, the clock should be fixed at 3 divisions, resulting in a CLK_PERI root clock of 160MHz. 2 divisions and 4 divisions of the CLK_PERI root clock result in 80MHz and 40MHz, which are provided to the encryption module and the interface module.

Setting BIT[15:8] of SYS_CLK_DIV register can adjust the WLAN clock division factor. The default dividing factor is 3, i.e., the 480MHz output of DPLL is divided by 3 to get 160MHz clock, which is sent to WLAN as the root node clock (WLAN will then continue to divide the clock to get a more detailed low-frequency clock for the use of WLAN system).

Note: If you want the WLAN system to work properly, the WLAN root clock needs to be kept at

160MHz or the WLAN system will fail. When the WLAN system is not required to work, the WLAN root clock can be lowered to reduce the dynamic power consumption of the system.

When changing the system clock configuration, it is important to note that the ratio of the system bus to the data bus needs to be maintained at M:1, where M is an integer with a minimum of 1. When changing the system clock configuration, it is also necessary to update BIT [23:16] of register SYS_CLK_DIV at the same time, to set the correct ratio factor. Otherwise, accessing the data bus will result in abnormal data.

The [15:8] of SYS_CLK_SEL provides the crossover factor to set the SAR_ADC operating frequency to 40M as the clock source. Crossover Factor The number is the assigned crossover frequency value.

BIT[4] of SYS_CLK_SEL is to configure the clock frequency selection for the core operation of RSA module, which can select 80MHz or 160MHz. BIT[5] is to configure the clock frequency selection for the core operation of GPSEC module, which can select 80MHz or 160MHz.

BIT[6] is to configure the clock frequency selection for the external bus of the FLASH module, either 40MHz or 80MHz can be selected.

When it is necessary to reconfigure cpu_clk_divider, wlan_clk_divider, bus2_syncdn_factor, and sdadc_fdiv, it is necessary to set BIT[31] of SYS_CLK_DIV, and the hardware automatically updates the above four parameters to the divider, and then clears BIT[31]. I2S_CLK_CTRL provides clock configuration for the I2S module.

5.3.5 Debug Function Control

Users can set the value of DEBUG_CTRL (SYS_CLK_SEL- BIT[16]) to enable and disable the JTAG function.

5.4 register description

5.4.1 register list

Table 7 Clock Reset Module Register List

offset address	name (of a thing)	abridge	inter views	descriptive	reset value
0X0000	Software Clock Gating Enable Register	SW_CLKG_EN	RW	Whether the software configuration module turns off the clock	0X0000_7FFF
0X0004	Software Clock Mask Register	SW_CLK_MASK	RW	When the software configuration module is or is not adaptive shutdown goblet	0X0000_007E
0X0008	reservations				
0X000C	Software Reset Control Register	SW_RST_CTRL	RW	Software Configuration Reset Module	0X01FF_FFFF
0X0010	Clock Divider Configuration Register	SYS_CLK_DIV	RW	Configuring the Clock Division Ratio	0X0000_2212
0X0014	Debug Control Register	DEBUG_CTRL	RW	Configuring ADC/DAC Loopback Tests	0X0000_0000
0X0018	I2S Clock Control Register	I2S_CLK_CTRL	RW	Configuring the I2S Clock	0X0000_0000
0X001C	Reset Status Register	RESET_STATUS	RW	View CPU Soft Reset with Watchdog reset state	0x0000_0000

5.4.2 Software Clock Gating Enable Register

Table 8 Software Clock Gating Enable Registers

classifier for honorifi	intervi ews	Operating Instructions	reset value

c people			
[31:22]	RO	reservations	
[21]	RW	soft_touch_gate_en Configure the gating of the touch_sensor module clock, default touch_sensor module gating is on 0: touch_sensor module clock is off	1'b1

		1: touch_sensor Clock on	
[20]	RW	<p>soft_bt_gate_en</p> <p>Configure gating of the BT./BLE module clock, default</p> <p>BT/BLE module gating on 0: BT/BLE module clock off</p> <p>1: BT/BLE clock on</p>	1'b1
[19]	RW	<p>soft_qsramp_gate_en</p> <p>Configure the gating of the qspi_ram module clock, default</p> <p>qspi_ram module gating is on 0: qspi_ram module clock is off</p> <p>1: qspi_ram clock on</p>	1'b1
[18]	RW	<p>soft_sdio_m_gate_en</p> <p>Configure the gating of the sdio_master module clock, default</p> <p>sdio_master module gating is on 0: sdio_master module clock is off</p> <p>1: sdio_master clock on</p>	1'b1
[17]	RW	<p>soft_gpsec_gate_en</p> <p>Configure the gating of the gpsec module clock, default gpsec module gating on 0: gpsec module clock off</p> <p>1: gpsec clock on</p>	1'b1
[16]	RW	<p>soft_rsa_gate_en</p> <p>Configure gating of the RSA clock, default RSA gating on 0: RSA module clock off</p>	1'b1

		1: RSA clock on	
[15]	RW	soft_i2s_gate_en	1'b1

		Configure the gating of the i2s clock, default i2s gating on 0: i2s clock off 1: i2s clock on	
[14]	RW	soft_lcd_gate_en Configure the gating of the lcd clock, default lcd gating on 0: lcd clock off 1: lcd clock on	1'b1
[13]	RW	Soft_pwm_gate_en Configure the gating of the pwm clock, default pwm gating on 0: pwm clock off 1: pwm clock on	1'b1
[12]	RW	soft_sd_adc_gate_en Configure gating of sd_adc_clock, default sd_adc_gating on 0: sd_adc_clock off 1: sd_adc_clock on	1'b1
[11]	RW	soft_gpio_gate_en Configure GPIO clock gating, default GPIO gating on 0: GPIO clock off 1: GPIO clock on	1'b1
[10]	RW	soft_timer_gate_en Configure the gating of the timer clock, default timer gating is on 0: timer clock is	1'b1

		off	
--	--	-----	--

		1: timer Clock on	
[9]	RW	<p>soft_rf_cfg_gate_en: for internal use, do not modify Configure gate control for rf_cfg clock, default rf_cfg gate control is</p> <p>on 1'b0: rf_cfg clock is off</p> <p>1'b1: rf_cfg clock on</p>	1'b1
[8]	RW	<p>soft_dma_gate_en</p> <p>Indicates whether the clock supplying the dma clock domain is</p> <p>turned off 1'b0: dma clock off</p> <p>1'b1: dma clock on</p>	1'b1
[7]	RW	<p>soft_ls_spi_gate_en</p> <p>Configure gating for low-speed spi clock, default low-speed spi gating on 1'b0: low-speed spi clock off</p> <p>1'b1: low speed spi clock on</p>	1'b1
[6]	RW	<p>soft_uart5_gate_en</p> <p>Configure door control for uart5, default uart5 on 0: uart5 off</p> <p>1: uart5 open</p>	1'b1
[5]	RW	<p>soft_uart4_gate_en</p> <p>Configure door control for uart4, default uart4 on 0: uart4 off</p> <p>1: uart4 open</p>	1'b1

[4]	RW	soft_uart3_gate_en	1'b1
-----	----	--------------------	------

		Configure door control for uart3, default uart3 on 0: uart3 off 1: uart3 open	
[3]	RW	soft_uart2_gate_en Configure door control for uart2, default uart2 on 0: uart2 off 1: uart2 on	1'b1
[2]	RW	soft_uart1_gate_en Configure the gating of the uart1 clock, default uart1 gating on 1'b0: uart1 clock off 1'b1: uart1 clock on	1'b1
[1]	RW	soft_uart0_gate_en Configure the gating of the uart0 clock, default uart0 gating on 1'b0: uart0 clock off 1'b1: uart0 clock on	1'b1
[0]	RW	soft_i2c_gate_en Configure i2c clock gating, default i2c gating on 1'b0: i2c clock off 1'b1: i2c clock on	1'b1

5.4.3 Software Clock Mask Register

Table 9 Software Clock Mask Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[6]	RW	<p>soft_cpu_clk_gt_mask</p> <p>Indicates whether or not the clock supplied to the CPU clock domain (including CPU, bus1, ROM, and SRAM) can be turned off adaptively (w h e n the CPU needs to enter the WFI state, don't set the adaptive off) 1'b0: Allow adaptive off and on</p> <p>1'b1: Adaptive switching off and on not allowed</p>	1'b1
[5 : 2]	RW	Reserved for internal use.	
[1]	RW	<p>soft_sdioahb_clk_gt_mask</p> <p>Indicates whether the clock supplying the sdio ahb clock domain can be turned off adaptively 1'b0:</p> <p>Adaptive turn-off and turn-on allowed</p> <p>1'b1: Adaptive switching off and on not allowed</p>	1'b1
[0]	RW	<p>soft_pmu_clk_gt_mask</p> <p>The clock output from pll is followed by a gating unit that is configured using this register to indicate whether it is allowed to be turned off by the PMU. 1'b0:</p> <p>Allows the PMU to turn off this gating unit, thus turning off the clock</p> <p>1'b1: the PMU is not allowed to switch off this gating unit</p>	1'b0

5.4.4 Software Reset

Control Register

Table 10 Software Reset Control Registers

classifier for	intervi ews	Operating Instructions	reset value
-------------------	----------------	------------------------	-------------

honorific people			
[31]	RW	soft_touch_RST_N Software reset touch_sensor module 0: reset	1'b1

		1: Reset release	
[30]	RW	<p>soft_rst_flash_n</p> <p>Software Reset Flash</p> <p>Controller Module 0: Reset</p> <p>1: Reset release</p>	1'b1
[29]	RW	<p>soft_rst_bt_n</p> <p>Software reset</p> <p>BT module 0:</p> <p>Reset</p> <p>1: Reset release</p>	1'b1
[28]	RW	<p>soft_rst_qspi_ram_n</p> <p>Software reset</p> <p>qspi_ram module 0:</p> <p>reset</p> <p>1: Reset release</p>	1'b1
[27]	RW	<p>soft_rst_sdio_m_n</p> <p>Software reset</p> <p>sdio_master module 0:</p> <p>reset</p> <p>1: Reset release</p>	1'b1
[26]	RW	<p>soft_rst_gpsec_n</p> <p>Software reset</p> <p>gpsec module 1'b0:</p> <p>reset</p>	1'b1

		1'b1: Reset release	
[25]	RW	soft_rst_rsa_n	1'b1

		Software reset RSA module 1'b0: reset 1'b1: Reset release	
[24]	RW	soft_RST_I2S_N Software reset I2S module 1'b0: reset 1'b1: Reset release	1'b1
[23]	RW	soft_RST_LCD_N Software reset LCD module 1'b0: reset 1'b1: Reset release	1'b1
[22]	RW	soft_RST_PWM_N Software reset PWM module 1'b0: reset 1'b1: Reset release	1'b1
[21]	RW	soft_RST_SAR_ADC_N Software reset SAR_ADC module 1'b0: reset 1'b1: Reset release	1'b1

[20]	RW	soft_rst_timer_n Software reset timer module 1'b0: reset	1'b1
------	----	---	------

		1'b1: Reset release	
[19]	RW	<p>soft_rst_gpio_n</p> <p>Software reset</p> <p>gpio module 1'b0:</p> <p>reset</p> <p>1'b1: Reset release</p>	1'b1
[18]	RW	<p>soft_rst_rf_cfg_n</p> <p>Software reset configures the RF's register module</p> <p>(internal use, do not modify) 1'b0: reset</p> <p>1'b1: Reset release</p>	1'b1
[17]	RW	<p>soft_rst_spis_n</p> <p>Software reset high</p> <p>speed spi module</p> <p>1'b0: reset</p> <p>1'b1: Reset release</p>	1'b1
[16]	RW	<p>soft_rst_spim_n</p> <p>Software reset low</p> <p>speed spi module</p> <p>1'b0: reset</p> <p>1'b1: Reset release</p>	1'b1
[15]	RW	<p>soft_rst_uart5_n</p> <p>Software reset on-</p> <p>chip uart5 module</p> <p>1'b0: reset</p>	1'b1

		1'b1: Reset release	
[14]	RW	soft_rst_uart4_n	1'b1

		Software reset on-chip uart4 module 1'b0: reset 1'b1: Reset release	
[13]	RW	soft_RST_uart3_n Software reset on-chip uart3 module 1'b0: reset 1'b1: Reset release	1'b1
[12]	RW	soft_RST_uart2_n Software reset on-chip uart2 module 1'b0: reset 1'b1: Reset release	1'b1
[11]	RW	soft_RST_uart1_n Software reset on-chip uart1 module 1'b0: reset 1'b1: Reset release	1'b1
[10]	RW	soft_RST_uart0_n Software reset on-chip uart0 module 1'b0: reset 1'b1: Reset release	1'b1

[9]	RW	soft_rst_i2c_n Software reset on- chip i2c module 1'b0: reset	1'b1
-----	----	--	------

		1'b1: Reset release	
[8]	RW	<p>soft_rst_bus2_n</p> <p>Software reset on-chip bus2 module</p> <p>1'b0: reset</p> <p>1'b1: Reset release</p>	1'b1
[7]	RW	<p>soft_rst_bus1_n</p> <p>Software reset on-chip bus1 module</p> <p>1'b0: reset</p> <p>1'b1: Reset release</p>	1'b1
[6]	RW	<p>soft_rst_apb_n</p> <p>Software reset apb bridge module 1'b0:</p> <p>reset</p> <p>1'b1: Reset release</p>	1'b1
[5]	RW	<p>soft_rst_mem_mng_n</p> <p>Software reset mem_mng module (internal use, do not modify) 1'b0: reset</p> <p>1'b1: Reset release</p>	1'b1
[4]	RW	<p>soft_rst_dma_n</p> <p>Software reset dma module 1'b0:</p> <p>reset</p>	1'b1

		1'b1: Reset release	
[3]	RW	soft_rst_sdio_ahb_n	1'b1

		software reset sdio ahb clock domain module 1'b0: reset 1'b1: Reset release	
[2]	RW	soft_rst_sec_n Software reset safety module (internal use, do not modify) 1'b0: reset 1'b1: Reset release	1'b1
[1]	RW	soft_rst_mac_n Software reset mac module (internal use, do not modify) 1'b0: reset 1'b1: Reset release	1'b1
[0]	RW	soft_rst_bbp_n Software reset bbp module (internal use, do not modify) 1'b0: reset 1'b1: Reset release	1'b1

5.4.5 Clock Divider

Configuration Registers

Table 11 Clock Division Configuration Registers

classifier for honori- fic people	intervi- ews	Operating Instructions	reset value

[31]	RW	divide_freq_en When it is necessary to reconfigure cpu_clk_divider, wlan_clk_divider, bus2_syncdn_factor, sdadc_fdiv, set this register, the hardware will automatically update the above four parameters to the divider, and then clear this register.	1'b0
------	----	--	------

		<p>Registers.</p> <p>1'b0: Crossover coefficient in effect</p> <p>1'b1: Request hardware to update crossover parameters</p> <p>Note: When configuring the dividing factors here, all factors must already be valid when Divide_freq_en is active</p>	
[30:28]		reservations	
[27:24]	RW	<p>Peripheral_divider</p> <p>160M clock Divider</p> <p>factor:</p> <p>DPLL is used as the clock source for frequency division. The crossover coefficient is the assigned crossover value. The crossover output should be 160MHz. The DPLL output is 480MHz and should be configured to 3.</p>	4'h3
[23:16]	RW	<p>bus2_syncdn_factor</p> <p>The clock ratio between bus1 and bus2 should be N:1.</p> <p>Where N is an integer, in the actual adjustment, it mainly depends on the ratio of CPU's operating frequency and bus2's clock frequency. Since the default CPU adopts 80MHz clock and the bus2 adopts 40MHz clock, then N=2</p>	8'h2

[15 : 8]	RW	wlan_clk_divider The clock from the PLL is divided and sent to the wlan system. This register is the frequency division factor, which is >=2. The default dividing factor is 3, i.e. the 480MHz output of pll is divided by 3 to get a 160MHz clock, which is given to wlan as the root node clock (wlan continues to divide the clock to get a more detailed low-frequency clock); Note 1: The clock should be fixed at 160MHz if the WLAN system is working properly, or the clock can be downclocked to save power consumption if the WLAN system is turned off. The clock must not be configured higher than 160MHz. Note 2: The secondary bus clock and APB clock are quadrature of this clock;	8'h3
----------	----	--	------

[7 : 0]	RW	<p>cpu_clk_divider</p> <p>The clock from the PLL is divided and sent to the CPU. this register is the division factor, which is ≥ 2.</p> <p>The default dividing factor is 6, i.e., after reset release, the 480MHz clock output from PLL will be divided into 6, and the clock sent to cpu will be 80MHz. This parameter can be reconfigured when you need to adjust the clock required by the cpu.</p>	8'h6
---------	----	--	------

5.4.6 Debug Control Register

Table 12 Clock Selection Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[16]	RW	<p>JTAG Enable</p> <p>1'b0: Disable JTAG debug function</p> <p>1'b1: Enable JTAG debug function</p>	1'b0
[15:8]	RW	<p>sd_adc_div</p> <p>sigma-delta ADC clock Crossover factor:</p> <p>The frequency division is performed using 40MHz as the clock source. The frequency division coefficient is the assigned frequency division value.</p> <p>Divide_freq_en in register clk_divider must be configured after configuring this register to take effect;</p>	8'd10
[7]	RW	RSV	1'b0

[6]	RW	qflash_clk_sel QSPI_FLASH clock selection 1: 80MHz is used;	1'b0
-----	----	--	------

		0: Use 40MHz;	
[5]	RW	gpsec_sel GPSEC Clock Selection 1: Use 160MHz; 0: 80MHz is used;	1'b0
[4]	RW	rsa_sel RSA Clock Option 1: Use 160MHz; 0: 80MHz is used;	1'b0
[3:0]	RW	Reserved.	4'd0

5.4.7 I2S Clock Control Register

Table 13 I2S Clock Control Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:18]		reservations	

[17: 8]	RW	<p>BCLKDIV</p> <p>BCLK allocator: $F_{BCLK} = F_{I2SCLK} / BCLKDIV$</p> <p>Note: If EXTAL_EN is not selected and the internal PLL is used then $F_{I2SCLK} = F_{CPU}$ (same frequency as CPU).</p> <p>Assuming $F_{CPU} = 160\text{MHz}$ and $F_{I2SCLK} = \text{external crystal frequency}$ when WXTAL_EN is enabled.</p> <p>BCLKDIV= round (F_I2SCLK/(Fs*W*F))</p> <p>where F_s is the sampling frequency of the audio data and W is the word width;</p>	10'b0
---------	----	--	-------

		<p>F = 1 when data is mono; F = 2 when data is stereo.</p> <p>For example, if the internal PLL is used and the data width is 24 bits, the format is stereo with a sampling frequency of 128KHz, BCLKDIV should be configured as $(160 * 10e6 / 128 * 10e3 * 24 * 2) = 10'h1a$.</p>	
[7 : 2]	RW	<p>MCLKDIV</p> <p>This MCLK divider is used to generate the appropriate MCLK frequency if an external clock is selected. $F_{mclk} = F_{I2SCLK} / (2 * MCLKDIV)$;</p> <p>When $MCLKDIV = 0$ F_{I2SCLK} is the external clock; when $MCLKDIV \geq 1$ $F_{mclk} = F_{I2SCLK}$;</p> <p>Note: F_{mclk} should be configured to $256 * fs$, where fs is the sampling frequency.</p>	6'b0
[1]	RW	<p>MCLKEN</p> <p>MCLK enable switch 1'b0:</p> <p>0: disable MCLK</p> <p>1: Enable MCLK</p>	1'b0

[0]	RW	<p>EXTAL_EN</p> <p>External clock selection, select whether to use internal I2S block clock or external clock 1'b0:</p> <p>internal clock</p> <p>1'b1: External clock</p> <p>Note: When using an external clock, the external clock must be $2 * N * 256$ fs, where fs is the sampling frequency and N must be an integer.</p>	1'b0
-----	----	---	------

5.4.8 Reset Status Register

Table 14 Reset Status Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:18]		reservations	
[17: 8]	WO	CPU soft reset state clear Write 1 Clears the CPU soft Reset Status.	1'b0
[7 : 2]	WO	Wdog soft reset state clear Write 1 Clears the Wdog Reset Status status.	1'b0
[1]	RO	CPU soft reset Status 1: The CPU has generated a soft reset; 0: The CPU did not generate a soft reset;	1'b0
[0]	RO	Wdog reset state 1: Wdog generates a Reset; 0: Wdpg does not generate a Reset	1'b0

6 DMA

Module

6.1 Functional overview

DMA is used to provide high-speed data transfers between peripherals and memory and between memory and memory. Data can be moved quickly through DMA without any CPU operation.

This saves CPU resources and does not affect the CPU's ability to perform other instructions.

DMA is mounted on the AHB bus and supports up to 8 channels, 16 hardware peripheral request sources, and a linked table structure with register control.

6.2 Main characteristics

- Amba2.0 standard bus interface, 8 DMA channels
- Supports DMA operations based on the memory chain table structure
- Supports 16 hardware peripheral request sources
- Supports 1, 4-burst mode of operation
- Supports byte, half-word, and word transfer operations.
- Supports constant or sequential incremental source and destination addresses or configurable cyclic operation over a predefined address range
- Supports memory-to-memory, memory-to-peripheral, and peripheral-to-memory data transfers

6.3 Functional Description

6.3.1 DMA Channel

W800 supports a total of 8 DMA channels, and the DMA channels do not interfere with each other and can operate at the same time. Different DMA channels can be selected for different data streams.

6 DMA Module

Each DMA channel is assigned to a different register address offset segment, so you can directly select the address segment of the corresponding channel to configure it for use. No

The registers for the same channel
are configured in exactly the same
way.

Table 15 DMA Address Assignments

DMA base address	0x4000 0800
DMA_CH0	Offset (0x10~0x38)
DMA_CH1	Offset (0x40~0x68)
DMA_CH2	Offset (0x70~0x98)
...	...
DMA_CH7	Offset (0x160~0x188)

6.3.2 DMA data stream

The 8 DMA channels enable a unidirectional data transfer link between source and destination.

The DMA source and destination addresses can be set to constant, incremental, or cyclic modes after each DMA operation:

- DMA_CTRL[2:1] controls the way the source address changes after each DMA operation;
- DMA_CTRL[4:3] controls how the destination address changes after each DMA operation.

The DMA can set the byte, half-word, and word carry units, and the final amount of data to be carried is an integer multiple of the carry unit, which is set via DMA_CTRL[6:5].

DMA can set how many units of data to be carried at a time by burst, and select 1 or 4 units of data to be carried at a time by DMA_CTRL[7]. If DMA_CTRL[6:5] is set to word, and burst is set to 4, then 4 words of data will be carried at a time.

DMA can set the number of Bytes to be transferred each time DMA is initiated, up to a maximum of 65535 Bytes, via DMA_CTRL[23:8].

6.3.3 DMA cyclic mode

DMA Cyclic Address Mode means that after setting the source and destination addresses of DMA, the data transfer will jump to the start address of the cycle when it reaches the set cycle boundary, and so on until it reaches the set transfer byte.

The source and destination addresses for the cyclic address mode need to be set using the SRC_WRAP_ADDR and DEST_WRAP_ADDR registers, and the length value of the cycle is set using WRAP_SIZE.

6.3.4 DMA transfer mode

DMA supports three transfer modes:

- Memory to Memory

Both source and destination addresses are configured as memory addresses to be transferred, and DMA_MODE[0] is set to 0. Software mode.

- Memory to Peripheral

The source address is set to the memory address, the destination address is set to the peripheral address, DMA_MODE[0] is set to 1 for hardware mode, and DMA_MODE[5:2] selects the peripheral used.

- Peripheral to Memory

The source address is set to the peripheral address, the destination address is set to the memory address, DMA_MODE[0] is set to 1, hardware mode, and DMA_MODE[5:2] selects the peripheral used.

6.3.5 DMA Peripheral Selection

When using peripheral-to-memory or memory-to-peripheral transfers, DMA_MODE[5:2] needs to be selected for the corresponding peripheral in addition to the corresponding peripheral being set to DMA TX or RX.

Note: Since there are 3 UART ports, when the UART uses DMA, you also need to select the corresponding UART port via UART_CH[1:0].

UART.

6.3.6 DMA chained table mode

DMA supports the chained table mode of operation. Through the chained table mode, when DMA moves the memory data of the current chained table, we can fill the data to the next chained table in advance, and after DMA moves the current chained table, it can directly move the data of the next chained table when it judges that the next chained table is valid. Through the way of the chain table can effectively improve the efficiency of DMA and CPU cooperation.

Chained Table Operation Mode: Set DMA to chained table operation mode through DMA_MODE[1] register, then set DESC_ADDR register as the start address of the chained table structure, and then enable DMA through CHNL_CTRL register. When DMA finishes processing the current memory move, the software notifies DMA that there is still valid data in the chained table by setting the valid flag, and DMA processes the next data to be moved according to the valid flag of the chained table. When the DMA finishes processing the current memory move, the software informs the DMA that there is still valid data in the chain table by setting the valid flag.

6.3.7 DMA interrupt

DMA transmission completion or burst can generate an interrupt, INT_MASK register can block the interrupt corresponding to the DMA channel.

When the DMA corresponding interrupt is generated, the status of the current interrupt can be queried through the INT_SRC register, indicating what the current interrupt is generated, and the corresponding status bit needs to be cleared by software writing 1.

6.4 register description

6.4.1 register list

Table 16 DMA Register List

offset address	name (of a thing)	abridge	inter views	descriptive	reset value
0X0000	Interrupt Mask Register	INT_MASK	RW	Setting DMA interrupts to be masked	0X0000_FFFF
0X0004	Interrupt Status Register	INT_SRC	RW	Indicates the current DMA interrupt status	0X0000_0000
0X0008	DMA channel selection register	DMA_CH	RW	Which UART is selected for the UART peripheral?	0X0000_0000

0X000C	reservations				
DMA CHNL0 registers					
0X0010	DMA Source Address Register	SRC_ADDR	RW	Source address for DMA transfers	0X0000_0000
0X0014	DMA Destination Address Register	DEST_ADDR	RW	Destination address for DMA transfers	0X0000_0000
0X0018	DMA Cycle Source Start Address Register	SRC_WRAP_ADDR	RW	Cyclic Mode DMA Transmit Source Address	0X0000_0000
0X001C	DMA Cycle Destination Start Address Register tool	DEST_WRAP_ADD R	RW	DMA transfer destination in cyclic mode sites	0X0000_0000
0X0020	DMA Cycle Length Register	WRAP_SIZE	RW	DMA Cycle Boundary in Cycle Mode	0X0000_0000
0X0024	DMA Channel Control Register	CHNL_CTRL	RW	Current channel DMA start and stop	0X0000_0000
0X0028	DMA Mode Selection Register	DMA_MODE	RW	Setting the DMA mode of operation	0X0000_0000
0X002C	DMA Data Flow Control Register	DMA_CTRL	RW	Setting the DMA transfer data stream	0X0000_0000
0X0030	DMA Transfer Byte Count Register	DMA_STATUS	RO	Get the current number of bytes transferred	0X0000_0000
0X0034	DMA Link Table Entry Address Register	DESC_ADDR	RW	DMA Link Table Address Entry Address Setting	0X0000_0000
0X0038	DMA Current Destination Address Register	CUR_DEST_ADDR	RO	Address of current DMA operation	0X0000_0000
DMA CHNL1 registers					
0X0040 - 0X0068	Same as DMA CHNL0 registers				
DMA CHNL2 registers					

0X0070 - 0X0098	Same as DMA CHNL0 registers
DMA CHNL3 registers	
0X00A0 -	Same as DMA CHNL0 registers

0X00C8	
DMA CHNL4 registers	
0X00D0 - 0X00F8	Same as DMA CHNL0 registers
DMA CHNL5 registers	
0X0100 - 0X0128	Same as DMA CHNL0 registers
DMA CHNL6 registers	
0X0130 - 0X0158	Same as DMA CHNL0 registers
DMA CHNL7 registers	
0X0160 - 0X0188	Same as DMA CHNL0 registers

6.4.2 Interrupt Mask Register

Table 17 DMA Interrupt Mask Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:16]		reservations	
[15]	RW	channel7 transfer_done Interrupt mask, valid high.	1'b1
[14]	RW	channel7 burst_done Interrupt mask, active high.	1'b1

[13]	RW	channel6 transfer_done Interrupt mask, high active.	1'b1
[12]	RW	channel6 burst_done Interrupt Mask, high active.	1'b1

[11]	RW	channel5 transfer_done Interrupt mask, high active.	1'b1
[10]	RW	channel5 burst_done Interrupt mask, high active.	1'b1
[9]	RW	channel4 transfer_done Interrupt mask, valid high.	1'b1
[8]	RW	channel4 burst_done Interrupt mask, high active.	1'b1
[7]	RW	channel3 transfer_done Interrupt mask, valid high.	1'b1
[6]	RW	channel3 burst_done Interrupt mask, active high.	1'b1
[5]	RW	channel2 transfer_done Interrupt mask, high active.	1'b1
[4]	RW	channel2 burst_done Interrupt mask, active high.	1'b1
[3]	RW	channel1 transfer_done Interrupt mask, high active.	1'b1
[2]	RW	channel1 burst_done Interrupt Mask, high active.	1'b1
[1]	RW	channel0 transfer_done Interrupt mask, active high.	1'b1
[0]	RW	channel0 burst_done Interrupt mask, active high.	1'b1

6.4.3 Interrupt Status Register

Table 18 DMA Interrupt Status Registers

classifier for honori- fic people	intervi- ews	Operating Instructions	reset value
[31:16]		reservations	
[15]	RW	channel7 transfer_done Interrupt status, write 1 to clear 0. DMA transfer completion generates an interrupt.	1'b0
[14]	RW	channel7 burst_done Interrupt status, write 1 to clear 0. DMA burst completed to generate interrupt.	1'b0
[13]	RW	channel6 transfer_done Interrupt status, write 1 to clear 0. DMA transfer	1'b0

		completion generates an interrupt.	
[12]	RW	channel6 burst_done Interrupt status, write 1 to clear 0. DMA burst completed to generate interrupt.	1'b0
[11]	RW	channel5 transfer_done Interrupt status, write 1 to clear 0. DMA transfer completion generates an interrupt.	1'b0

[10]	RW	channel5 burst_done Interrupt status, write 1 to clear 0. DMA burst completed to generate interrupt.	1'b0
[9]	RW	channel4 transfer_done Interrupt status, write 1 to clear 0. DMA transfer completion generates an interrupt.	1'b0
[8]	RW	channel4 burst_done Interrupt status, write 1 to clear 0. DMA burst completed to generate interrupt.	1'b0
[7]	RW	channel3 transfer_done Interrupt status, write 1 to clear 0. DMA transfer completion generates an interrupt.	1'b0
[6]	RW	channel3 burst_done Interrupt status, write 1 to clear 0. DMA burst completed to generate interrupt.	1'b0
[5]	RW	channel2 transfer_done Interrupt status, write 1 to clear 0. DMA transfer completion generates an interrupt.	1'b0
[4]	RW	channel2 burst_done Interrupt status, write 1 to clear 0. DMA burst completed to generate interrupt.	1'b0
[3]	RW	channel1 transfer_done Interrupt status, write 1 to clear 0. DMA transfer completion generates an interrupt.	1'b0
[2]	RW	channel1 burst_done Interrupt status, write 1 to clear 0. DMA burst completed to generate interrupt.	1'b0
[1]	RW	channel0 transfer_done Interrupt status, write 1 to clear 0. DMA transfer completion generates interrupt.	1'b0
[0]	RW	channel0 burst_done Interrupt status, write 1 to clear 0. DMA burst completed to generate interrupt.	1'b0

6.4.4 UART Select Register

Table 19 UART Selection Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:24]		reservations	

[23:8]	RW	<p>dma req clear.</p> <p>Write 1 to each bit to clear the corresponding dma req request. Self-clearing.</p> <p>For example, writing 1 to bit 23 clears the 15th corresponding dma request in dma_sel; writing 1 to bit 8 clears the 0th dma request in dma_sel-uart_rx_req;</p>	16'd0
[2 : 0]	RW	<p>Uart dma channel selection:</p> <p>3'd0: uart0 module dma channel Access dma</p>	3'h0

		3'd1: uart1 module dma channel access dma 3'd2: uart2/7816 module dma channel Access dma 3'd3: uart3 module dma channel Access dma 3'd4: uart4 module dma channel Access dma 3'd5: uart5 module dma channel Access dma	
--	--	---	--

6.4.5 DMA Source Address Register

Table 20 DMA Source Address Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:0]	RW	Source address, peripheral address, or memory address for DMA carry in acyclic mode	32'h0

6.4.6 DMA Destination Address Register

Table 21 DMA Destination Address Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:0]	RW	Destination address, peripheral address, or memory address for DMA carry in acyclic mode	32'h0

6.4.7 DMA Cycle Source Start Address Register

Table 22 DMA Cycle Source Start Address Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:0]	RW	Starting address of the source address, peripheral address or memory address for DMA carry in cyclic mode	32'h0

6.4.8 DMA Cycle Destination Start Address Register

Table 23 DMA Cycle Destination Start Address Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:0]	RW	Starting address of the destination address of the DMA carry, peripheral address or memory address in cyclic mode	32'h0

6.4.9 DMA Cycle Length Register

Table 24 DMA Cycle Length Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:16]	RW	The DMA destination address cycle length in cyclic mode. The DMA handles data incrementally from the start address, and when the number of bytes of data handled reaches this set value, it will jump to the start address of the cycle and continue to handle data from the start address.	16'h0
[15:0]	RW	DMA source address cycle length in cyclic mode.	16'h0

6.4.10 DMA Channel Control Register

Table 25 DMA Channel Control Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value

[31: 2]		reservations	
[1]	RW	<p>dma_stop</p> <p>Stop dma operation, high effective.</p> <p>The DMA will stop after the current burst operation is completed and chnl_on is cleared. software should determine that the dma has completely stopped based on chnl_on being 0.</p>	1'b0
[0]	RW	chnl_on	1'b0

		<p>Initiates the current channel DMA conversion, active high.</p> <p>Dma is automatically cleared to 0 when conversion is completed or setting is stopped.</p>	
--	--	--	--

6.4.11 DMA Mode Selection Register

Table 26 DMA Mode Selection Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31: 7]		reservations	
[6]	RW	<p>chain_link_en</p> <p>Valid in chained table mode, indicates whether the dma will continue to read and process the subsequent chained tables after the first one is processed.</p> <p>If 1, update next_desc_addr in the chain table and continue reading the next chain table until the chain table</p> <p>vld in vld is 0; if it is 0, the processing stops when it finishes with the current linked table.</p>	1'b0

[5 : 2]	RW	<p>dma_sel</p> <p>16 dma_req selections.</p> <p>4'd0: uart rx dma req</p> <p>4'd1: uart tx dma req</p> <p>4'd2: pwm_cap0_req</p> <p>4'd3: pwm_cap1_req 4'</p> <p>d4: ls_spi rx dma req</p> <p>4'd5: ls_spi tx dma req</p> <p>4'd6: sd_adc chnl0 req</p> <p>4'd7: SD_ADC chnl1 req</p>	4'h0
---------	----	---	------

		4'd8: SD_ADC chnl2 req 4'd9: SD_ADC chnl3 req 4'd10: I2S RX req 4'd11: I2S TX req. 4'd12: SDIO_HOST req	
[1]	RW	chain_mode 1'b0: use normal mode 1'b1: Use of the chained table model	1'b0
[0]	RW	dma_mode 1'b0: software mode. 1'b1: Hardware approach.	1'b0

6.4.12 DMA Data Flow Control Register

Table 27 DMA Data Flow Control Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:24]		reservations	
[23: 8]	RW	total_byte The total number of bytes to be operated. The number of bytes to be operated should be consistent with the data_size configuration, i.e., an integer multiple of 4 for word operations, or an integer multiple of 2 for half-word operations.	16'h0

[7]	RW	burst_size Sets how many units of data the DMA will carry at a time 1'b0: burst is 1	1'b0
-----	----	---	------

		<p>1'b1: burst is 4</p> <p>When the last burst size exceeds the number of remaining transfers, use burst size for the size of the remaining data.</p>	
[6 : 5]	RW	<p>data_size</p> <p>Set the DMA handling unit 2'h0:</p> <p>byte 2'h1: half_word</p> <p>2'h2: word</p> <p>2'h3: Reserved</p>	2'h0
[4 : 3]	RW	<p>dest_addr_inc</p> <p>2'h0: the destination address remains unchanged after each operation;</p> <p>2'h1: The destination address is automatically accumulated after each operation.</p> <p>2'h2: Reserved</p> <p>2'h3: Loop operation, after each operation, the destination address is accumulated automatically, and the defined loop boundary is reached to jump to the start address of the loop.</p>	2'h0

[2 : 1]	RW	<p>src_addr_inc 2'h0: source address unchanged after each operation; 2'h1: The source address is automatically accumulated after each operation. 2'h2: Reserved 2'h3: Loop operation, the source address is accumulated automatically after each operation, and jumps to the start address of the loop when it reaches the defined loop boundary.</p>	2'h0
[0]	RW	auto_reload	1'b0

		Upon completion of the current DMA carry, the next DMA carry is automatically re-run in the current DMA configuration.	
--	--	--	--

6.4.13 DMA Transfer Byte Count Register

Table 28 DMA Transfer Byte Count Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:16]		reservations	
[15:0]	RW	<p>transfer_cnt</p> <p>The number of bytes currently being transferred.</p> <p>Each time the dma is turned back on (chnl_on set to 1) clear 0 and restart counting.</p>	16'h0

6.4.14 DMA Link Table Entry Address Register

Table 29 DMA Link Table Entry Address Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:0]	RW	<p>desc_addr</p> <p>Used as the entry address of the chain table when the chain table is enabled.</p> <p>After each transfer completes the chain table, the base address of the next chain table is updated to this register.</p>	32'h0

6.4.15 DMA Current Destination Address Register

Table 30 DMA Current Destination Address Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:0]	RO	current_dest_addr Current DMA operation destination address.	32'h0

		When software stops dma, you can check this register to learn the destination address where dma will operate.	
--	--	---	--

7 Universal Hardware Encryption Module

7.1 Functional overview

The encryption module automatically completes the encryption of the specified length of source address space data, and automatically writes back the encrypted data to the specified destination address space after completion; supports SHA1/MD5/RC4/DES/3DES/AES/CRC/TRNG.

7.2 key feature

- Support SHA1/MD5/RC4/DES/3DES/AES/CRC/TRNG encryption algorithms.
- DES/3DES supports both ECB and CBC modes.
- AES supports ECB, CBC and CTR modes.
- CRC supports CRC8, CRC16_MODBUS, CRC16_CCITT and CRC32 modes.
- CRC supports input/output reversal
- SHA1/MD5/CRC Support continuous multi-packet encryption
- Built-in true random number generator, also supports seed seeds to generate pseudo-random numbers

7.3 Functional Description

7.3.1 SHA1 encryption

Hardware SHA1 calculations can be performed on consecutive packets of data, the results of the calculations are stored in registers, and the encryption result of the previous packet can be used as the initial value of the next packet.

7.3.2 MD5 encryption

Hardware MD5 calculations can be performed on consecutive packets of data, the results of which

are stored in a register, and the encryption result of the previous packet can be used as the initial value of the next packet.

7.3.3 RC4 encryption

Supports RC4 encryption and decryption.

7.3.4 DES encryption

Supports DES encryption and decryption, ECB and CBC modes.

7.3.5 3DES encryption

Supports 3DES encryption and decryption, ECB and CBC modes.

7.3.6 AES encryption

Supports AES encryption and decryption, ECB, CBC and CTR modes.

7.3.7 CRC encryption

Hardware CRC calculation can be performed on consecutive multi-packet data, the calculation result exists in the register, the encryption result of the previous packet can be used as the initial value of the next packet, it supports four modes: CRC8, CRC16_MODBUS, CRC16_CCITT and CRC32, and it supports input/output reversal. The CRC32 calculation formula is as follows:

1. CRC-32: 0x04C11DB7

$$X^{32}+X^{26}+X^{23}+X^{22}+X^{16}+X^{12}+X^{11}+X^{10}+X^8+X^7+X^5+X^4+X^2+X+1$$

Commonly used for protocols such as ZIP, RAR, IEEE 802

LAN/FDDI, IEEE 1394, PPP-FCS and others.

2、CRC-16: supports two kinds of polynomials

21 : 0X1021

X16 + X12 + X5 1+

Commonly used in protocols such as ISO HDLC, ITU X.25, V.34/V.41/V.42, PPP-FCS CCITT.

22 : 0X8005

X16 + X15 + X2 1+

Commonly used in USB, ANSI X3.28, SIA

DC-07 and other protocols. 3、CRC-8 : 0X207

x8+x2+x1+1

7.3.8 TRNG Random Number Generator

A true random number generator module is integrated into the W800 system. It is divided into an analog module and a digital post-processing module. The analog module outputs the random clock ad_trng_clks and the random number ad_trng_dout, and the digital post-processing module is used to eliminate the bias and autocorrelation of the random number.

The related control registers are in the GPSEC register list.

The basic operation process is as follows:

1. Enable TRNG_EN and set TRNG_SEL to 1 to make GPSEC register 0x48 display the output value of TRNG. At this point, the analog module starts to output random clock and random signal. The signals obtained from the first 8 clock samples are used as the initial state of the LFSR to initialize the LFSR chain, and the data obtained from each of the subsequent random clock samples are shifted and stored in TRNG_RANDOM after post-processing in the XOR CHAIN and LFSR registers.
2. The software can read the random value through GPSEC register 0x48. When register TRNG_DIG_BYPASS is set to 1, the digital post-processing module stops working and stores the output value of the analog module directly into the result register TRNG_RANDOM.

7.4 register description

7.4.1 register list

Table 31 Encryption Module Register List

displacement addresses	name (of a thing)	abridge	inter views	descriptive	reset value
0X0000	Source Address Register	SRC_ADDR	RW	rc4/sha1/aes/des/3des/crc/md 5 Multiplexing source addresses	0X0000_0000
0X0004	Destination Address Register	DEST_ADDR	RW	RC4/AES/DES/3DES multiplexing destination address	0X0000_0000
0X0008	configuration register	GPSEC_CFG	RW	General Purpose Hardware Encryption Module Configuration Registers	0X0000_0000
0X000C	control register	GPSEC_CTRL	RW	General Purpose Hardware Encryption Module Control Registers	0X0000_0000
0X0010	Secret key 0 Low register	KEY00	RW	Key0 low 32-bit first input key (RC4/AES/DES/3DES), multiplexed CRC Ci	0X0000_0000
0X0014	Secret key 0 Higher register	KEY01	RW	Key0 High 32-bit first input key (RC4/AES/DES/3DES)	0X0000_0000
0X0018	Secret key 1 Low register	KEY10	RW	Key1 Low 32-bit second input key (RC4/AES//3DES)	0X0000_0000
0X001C	Secret Key 1 High Level Register	KEY11	RW	Key1 High 32-bit second input key (RC4/AES//3DES)	0X0000_0000

0X0020	Secret Key 2 Low Register	KEY20	RW	Key2 Low 32-bit third input key (3DES), multiplexed iv1 low 32-bit input initial vector (AES)	0X0000_0000
--------	---------------------------	-------	----	---	-------------

0X0024	Secret Key 2 High Level Register	KEY21	RW	Key2 High 32-bit third input key (3DES), multiplexed iv1 high 32-bit input initial vector (AES)	0X0000_0000
0X0028	Initial vector 0 Low register tool	IV00	RW	IV0 Low 32-bit input initial vector (AES/DES/3DES)	0X0000_0000
0X002C	Initial vector 0 high register tool	IV01	RW	IV0 High 32-bit input initial vector (AES/DES/3DES)	0X0000_0000
0X0030	status register	GPSEC_STS	RW	General Purpose Hardware Encryption Module Status Register	0X0000_0000
0X0034	Summary 0 Register	SHA1-DIGEST0	RW	sha1-digest0/md5-digest0	0X6745_2301
0X0038	Summary 1 Registers	SHA1-DIGEST1	RW	sha1-digest1/md5-digest1	0XEFC0_AB89
0X003C	Summary 2 Registers	SHA1-DIGEST2	RW	sha1-digest2/md5-digest2	0X98BA_DCFE
0X0040	Summary 3 Registers	SHA1-DIGEST3	RW	sha1-digest3/md5-digest3	0X1032_5476
0X0044	Summary 4 Registers	SHA1-DIGEST4	RW	sha1-digest4/CRC	0XC3D2_E1F0
0X0048	RNG_result	RNG_RESULT	RW	RNG Output	0X0000_0000
0X004C	Secret Key 3 Low Register	Key30	RW	Key3 Low 32-bit third input key (RC4) (256bit mode)	0X0000_0000
0X0050	Secret Key 3 Low Register	Key31	RW	Key3 High 32-bit third input key (RC4) (256bit mode)	0X0000_0000
0X0054	TRNG Configuration	TRNG_CR	RW	True Random Number Generator Configuration Options	0X40

7.4.2 configuration register

Table 32 Encryption Module Configuration Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31]	RW	<p>RC4_128_256</p> <p>0: Flag that RC4 encryption/decryption is performed according to 128bit block length;</p> <p>1: Flag RC4 encryption/decryption is performed according to 256bit block length.</p>	1'b0
[30]	RW	<p>RNG start 1'b0: do not start RNG</p> <p>1'b1: Initiate the RNG</p>	1'b0
[29]	RW	<p>RNG Load_seed</p> <p>hardware auto-clear 0</p> <p>1'b0: the random number generator will default to zero as the seed, generating a random number with the corresponding number of bits</p> <p>1'b1: start generating random numbers after the seed is loaded</p>	1'b0
[28]	RW	<p>RNG switch</p> <p>Controls the number of bits in the generated random number, 1'b0:</p> <p>16 bits</p> <p>1'b1: 32-bit</p>	1'b0

[27]	RW	<p>des_soft_reset</p> <p>des Hardware automatically clears</p> <p>0 after soft reset is completed 1'b0:</p> <p>no soft reset is generated and the current state is not changed</p> <p>1'b1: Encryption algorithm reset to initial state by software</p>	1'b0
[26]	RW	<p>aes_soft_reset</p> <p>aes Hardware automatically clears</p> <p>0 after soft reset is completed 1'b0:</p> <p>does not generate soft reset and does not change current state</p>	1'b0

		1'b1: Encryption algorithm reset to initial state by software	
[25]	RW	<p>rc4_soft_reset</p> <p>rc4 Hardware automatically clears 0</p> <p>after soft reset is completed 1'b0: no</p> <p>soft reset is generated and the current</p> <p>state is not changed</p> <p>1'b1: Encryption algorithm reset to initial state by software</p>	1'b0
[24]	RW	<p>crc_datarev</p> <p>1'b0: CRC input data not inverted</p> <p>1'b1: CRC input data reversal</p>	1'b0
[23]	RW	<p>crc_chksrev</p> <p>1'b0: CRC output result not inverted</p> <p>1'b1: CRC output result reversed</p>	1'b0
[22:21]	RW	<p>sub_mode</p> <p>Algorithm type submode selection:</p> <p>0: ECB mode for DES/AES cipher algorithm, CRC8 mode for reusable CRC algorithm</p> <p>1: CBC for DES/AES cipher algorithm, CRC16_0 mode for reusable CRC algorithm</p> <p>2: CTR mode for AES cipher algorithm, CRC16_1 mode for reusable CRC algorithm</p> <p>3: MAC mode of the AES cipher algorithm, reusable CRC algorithm CRC32</p>	2'b0

[20]	RW	encrypt_decrypt Encryption or decryption mode selection for RC4/AES/DES/3DES algorithms: 1'b0: encryption 1'b1: Decryption	1'b0
[19]	RW	gpsec_int_mask	1'b0

		1'b0: do not mask encryption/decryption completion interrupt 1'b1: Masked encryption/decryption completion interrupt	
[18:16]	RW	cypher_mode cipher algorithm type 3'b000: RSV 3'b001: RC4 3'b010: SHA1 3'b011: AES 3'b100: DES 3'b101: 3DES 3'b110: CRC 3'b111: MD5	3'b0
[15:0]	RW	total_byte The total number of bytes to be encrypted and decrypted.	16'h0

7.4.3 TRNG Control Register

Table 33 TRNG Module Control Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value

[31: 7]		reservations	
[6]	RW	TRNG_INT_MASK TRNG interrupt mask	1'b1

		0: TRNG module reports an interrupt; 1: The TRNG module does not report interrupts;	
[5:3]	RW	TRNG_CP TRNG Module Control Signal	3'd0
[2]	RW	TRNG_DIG_BYPASS TRNG digital post-processing module bypass: 0: TRNG module for post-processing; 1: The TRNG module does not perform post-processing;	1'b0
[1]	RW	TRNG_SEL. RNG Output selection signal. 0: Register 0x48 Displays the pseudo-random module output; 1: Register 0x48 Displays the TRNG output result;	1'b0
[0]	RW	TRNG_EN. TRNG Module enable signal. Valid high. 0: TRNG module stops; 1: The TRNG module starts working;	1'b0

7.4.4 control register

Table 34 Encryption Module Control Registers

classifier for	intervi ews	Operating Instructions	reset value

honorifi			
[31: 2]		reservations	

[1]	RW	<p>sec_stop</p> <p>Stop the encryption/decryption operation currently in progress</p> <p>1'b0: invalid</p> <p>1'b1: encryption/decryption stopped</p>	1'b0
[0]	RW	<p>sec strt</p> <p>Start encryption/decryption, after completing encryption/decryption operation byte number, hardware automatically clear 0 1'b0: do not start encryption/decryption</p> <p>1'b1: Initiate encryption/decryption</p>	1'b0

7.4.5 status register

Table 35 Encryption Module Status Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31: 17]		reservations	
[16]	RW	<p>int_flag</p> <p>Software write 1</p> <p>clear 1'b0: no</p> <p>encryption/decryption</p> <p>completion interrupt</p>	1'b0

		<p>generated</p> <p>1'b1: Generate encryption/decryption completion interrupt</p>	
[15:0]	RO	<p>transfer_cnt</p> <p>The number of bytes currently encrypted to completion.</p> <p>Clears 0 each time encryption and decryption is turned back on and starts counting again.</p>	16'h0

8 RSA encryption module

8.1 Functional overview

RSA operations hardware coprocessor providing Montgomery (FIOS algorithm) modulo operations. Works with the RSA software library to implement the RSA algorithm. Supports 128-bit to 2048-bit multiplication.

8.2 key feature

- Supports 128-bit to 2048-bit modulo (modulo length is an integer multiple of 32 bits)
- Supports 4 modulo modes including D^*D , X^*Y , D^*Y , X^*X , etc.

8.3 Functional Description

8.3.1 mode multiplication function (math.)

RSA operations hardware coprocessor providing Montgomery (FIOS algorithm) modulo multiplication. Works with the RSA software library to implement the RSA algorithm. Supports 128-bit to 2048-bit multiplication.

8.4 register description

8.4.1 register list

Table 36 RSA Register List

offset address	name (of a thing)	abridg e	intervi ews	descriptive	reset value
0X0000~0X00FC	Data X Register	XBUF	RW	Data X Register	
0X0100~0X01FC	Data Y Register	YBUF	RW	Data Y Register	
0X0200~0X02FC	Data M Register	MBUF	RW	Data M Register	

0X0300~0X03FC	Data D Register	DBUF	RW	Data D Register	
---------------	-----------------	------	----	-----------------	--

0X0400	RSA Control Register	RSACON	RW	RSA Control Register	0X0000_0000
0X0404	Parameter MC Register	RSAMC	WO	Parameter MC Register	0X0000_0000
0X0408	Parameter N Register	RSAN	RW	Parameter N Register	0X0000_0000

8.4.2 Data X Register

XBUF corresponds to the buffer of data X (2048bit), and the corresponding haddr values are 0000h~00fch. The corresponding rules are shown in the following table:

Table 37 RSA Data X Register

000h	004h	008h	00f8h	00fch
X[31:0]	X[63:32]	X[95:64]	X [2015:1984]	X [2047:2016]

8.4.3 Data Y Register

YBUF corresponds to the buffer of data Y (2048bit), and the corresponding haddr values are 0100h~01fch. The corresponding rules are listed in the following table:

Table 38 RSA Data Y Register

0100h	0104h	0108h	01f8h	01fch
Y[31:0]	Y [63:32]	Y [95:64]	Y [2015:1984]	Y [2047:2016]

8.4.4 Data M Register

MBUF corresponds to the buffer of data M (2048 bits), and the corresponding haddr values are 0200h~02fch. The corresponding rules are shown in the following table:

Table 39 RSA Data M Register

0200h	0204h	0208h	02f8h	02fch
M[31:0]	M[63:32]	M[95:64]	M[2015:1984]	M [2047:2016]

8.4.5 Data D Register

DBUF corresponds to the buffer of data D (2048bit), and the corresponding haddr values are 0300h~03fch. The corresponding rules are shown in the following table:

Table 40 RSA Data D Register

0300h	0304h	0308h	03f8h	03fc
D[31:0]	D[63:32]	D[95:64]	D[2015:198 4]	D[2047:2016]

8.4.6 RSA Control Register

RSACON, the RSA control register, is a 32bit register in actual physical space.

Table 41 RSA Control Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31: 6]		reservations	
[5]	RW	Modulo start control bit. The software writes "1" to start the modulo operation, and the hardware automatically clears "0" when the operation is finished.	1'b0
[4]	RW	Provide soft reset function with high validity. Software writes "1" for soft reset, and hardware automatically clears "0" after reset is completed. 1. Set the parameters MC and N to 0. 2. After starting the modulo multiplication (bit5 set to 1), this position "1" will terminate the current operation (when bit0 becomes high, it means that the execution of the soft reset command is completed and the operation is terminated), but the part of the operation result that has already been completed in the internal data buffer (X, Y, M, D) will be retained.	1'b0

[2 : 3]	RW	Die multiplication mode selection. 2'b00: $X = D*D \bmod M$ 2'b01: $D = X*Y \bmod M$ 2'b10: $X = D*Y \bmod M$ 2'b11: $D = X*X \bmod M$	2'b0
[1]	RW	reservations	1'b0

[0]	RW	Modulo operation completion mark, high validity. Hardware sets "1", software clears "0". Software write "1" is invalid.	1'b0
-----	----	---	------

8.4.7 Parameter MC Register

Table 42 RSA Parameters MC Register

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:0]	WO	RSAMC Corresponding parameter MC (32bit). The reset value is all 0. The readout value is all 0.	32'h0

8.4.8 Parameter N Register

RSAN corresponds to parameter N (7bit) the value of which is the length of the modulo divided by 32. That is, if you call a 1024bit modulo operation, you need to set N

=The lower 7 bits are valid for writing to this register, and the upper 25 bits are 0 for reading. the reset value is all 0.

Table 43 RSA Parameters N Register

classi fier for hono rific peop le	intervi ews	Operating Instructions	reset value
[31: 7]		reservations	
[6 : 0]	RW	RSAN corresponds to parameter N (7bit) the value of N is the value of the modulo length divided by 32.	7'h0

9 GPIO Module

9.1 Functional overview

The GPIO controller implements software configuration of GPIO properties, allowing the user to easily operate the GPIOs.

Each GPIO can be individually configured by software to set it as an input port and output port, set its suspend, pull-up and pull-down status, and set its rising edge, falling edge, double edge, high level and low level interrupt triggering mode.

9.2 Main characteristics

- Supports GPIO software configuration
- Supports GPIO interrupt configuration
- Up to 48 GPIOs available

9.3 Functional Description

The GPIOs provided in W800 are divided into two groups, one group is GPIOA and the other group is GPIOB. The GPIOA and GPIOB registers have different start addresses but the functions are the same.

If you want to use a specific IO as a software-controlled GPIO, set the corresponding position in the GPIO Multiplexing Selection Register to 0 to do so.

The GPIO Direction Control Register is used to control the direction of the GPIOs. 1 means the corresponding GPIO is used as an output pin and 0 means the corresponding GPIO is used as an input pin.

The GPIO pull-up and pull-down control registers are used to control the pull-up and pull-down functions of the corresponding IOs.

The GPIO pull-up control register is active low, set to 0 to turn on the pull-up function of the corresponding IO, set to 1 to turn off the pull-up and pull-down function.

See the IO Multiplexing table for the attributes that IO has.

The GPIO pull-down control register is active high, set to 1 to enable the pull-down function of the corresponding IO, set to 0 to disable the pull-down function. Please refer to the IO multiplexing table for the attributes of the IOs.

The GPIO data register indicates the input IO level when it is set to the input state, and you can write 1 or 0 to specify the output level of the IO when it is set to the output state. This register is controlled by the GPIO data enable register, and the GPIO data register can be read or written only when the GPIO data enable register is set to 1.

The GPIO module provides input signal detection function. By configuring the registers related to GPIO interrupt, high and low level detection as well as upper and lower edge jump detection can be realized. When the input signal of the corresponding IO meets the pre-set conditions, such as high level trigger or rising edge trigger, the GPIO interrupt will be triggered and reported to the MCU for processing, and the MCU needs to clear the corresponding interrupt status in order to avoid interrupt triggering by mistake.

9.4 register description

9.4.1 register list

Table 44 GPIOA Register List

offset address	name (of a thing)	abridge	inter view s	descriptive	reset value
0X0000	GPIO Data Register	GPIO_DATA	RW	Read/write GPIO current data	0X180B

0X0004	GPIO Data Enable Register	GPIO_DATA_E N	RW	Configure the enable bit of GPIO_DATA.	0xFFFF
0X0008	GPIO Direction Control Register	GPIO_DIR	RW	Configure GPIO direction	0X0000
0X000C	GPIO Pull-up Control Register	GPIO_PULL_E N	RW	Configure GPIO pull-ups	0xFFFF
0X0010	GPIO Multiplexing Select Register	GPIO_AF_SEL	RW	Configure the GPIO multiplexing enable bit.	0xFFFF

0X0014	GPIO multiplexing selection register 1	GPIO_SF_S1	RW	GPIO multiplexing function select bit high address bit	0X0000
0X0018	GPIO multiplexing selection register 0	GPIO_AF_S0	RW	GPIO multiplexing function select bit low address bit	0X0000
0X001C	GPIO Pull Down Control Register	GPIO_DN_ENA	RW	Configuring GPIO Pull Downs	0X0000
0X0020	GPIO Interrupt Trigger Mode Configuration Register	GPIO_IS	RW	Configure the interrupt triggering method for GPIOs	0X0000
0X0024	GPIO Interrupt Edge Trigger Mode Configuration depositor (computing)	GPIO_IBE	RW	Configuring GPIO Interrupt Edge Trigger Mode	0X0000
0X0028	GPIO interrupt upper and lower edge trigger configuration depositor (computing)	GPIO_IEV	RW	Configure GPIO interrupt upper and lower edge triggering or high and low level trigger	0X0000
0X002C	GPIO Interrupt Enable Configuration Register	GPIO_IE	RW	Configure GPIO interrupt enable	0X0000
0X0030	GPIO Bare Interrupt Status Register	GPIO_RIS	RO	Query GPIO bare interrupt status (before MASK)	0X0000
0X0034	GPIO Masked Interrupt Status Register	GPIO_MIS	RO	Queries the interrupt status after GPIO masking (MASK). (after)	0X0000
0X0038	GPIO Interrupt Clear Control Register	GPIO_IC	WO	Control GPIO interrupt clear	0X0000

Table 45 GPIOB Register List

offset address	name (of a thing)	abridge	inter views	descriptive	reset value
0X0000	GPIO Data Register	GPIO_DATA	RW	Read/write GPIO current data	0X0000_7304

0X0004	GPIO Data Enable Register	GPIO_DATA_E N	RW	Configure the enable bit of GPIO_DATA.	0X7FFF_FFFF
0X0008	GPIO Direction Control Register	GPIO_DIR	RW	Configure GPIO direction	0X0000_0000
0X000C	GPIO Pull-up Control Register	GPIO_PULL_E N	RW	Configure GPIO pull-ups	0XFFFF_FFFF

0X0010	GPIO Multiplexing Select Register	GPIO_AF_SEL	RW	Configure the GPIO multiplexing enable bit.	0xFFFF_FFFF
0X0014	GPIO multiplexing selection register 1	GPIO_SF_S1	RW	GPIO multiplexing function select bit high address bit	0X0000_0000
0X0018	GPIO multiplexing selection register 0	GPIO_AF_S0	RW	GPIO multiplexing function select bit low address bit	0X0000_0000
0X001C	GPIO Pull Down Control Register	GPIO_DN_ENA	RW	Configuring GPIO Pull Downs	0X0000_0000
0X0020	GPIO Interrupt Trigger Mode Configuration Register	GPIO_IS	RW	Configure the interrupt triggering method for GPIOs	0X0000_0000
0X0024	GPIO Interrupt Edge Trigger Mode Configuration depositor (computing)	GPIO_IBE	RW	Configuring GPIO Interrupt Edge Trigger Mode	0X0000_0000
0X0028	GPIO interrupt upper and lower edge trigger configuration depositor (computing)	GPIO_IEV	RW	Configure GPIO interrupt upper and lower edge triggering or High-Low Level Trigger	0X0000_0000
0X002C	GPIO Interrupt Enable Configuration Register	GPIO_IE	RW	Configure GPIO interrupt enable	0X0000_0000
0X0030	GPIO Bare Interrupt Status Register	GPIO_RIS	RO	Queries the GPIO bare interrupt status (MASK). (front)	0X0000_0000
0X0034	GPIO Masked Interrupt Status Register	GPIO_MIS	RO	Query the interrupt status after GPIO masking (after MASK)	0X0000_0000
0X0038	GPIO Interrupt Clear Control Register	GPIO_IC	WO	Control GPIO interrupt clear	0X0000_0000

9.4.2 GPIO Data Register

Table 46 GPIOA Data Registers

classif	intervi	Operating Instructions	reset value
---------	---------	------------------------	-------------

ierfor honor ific peopl e	ews		
[15:0]	RW	GPIO current data, each BIT corresponds to the corresponding GPIO line	16'h180b

Table 47 GPIOB Data Registers

classifier for honorific people	interviews	Operating Instructions	reset value
[31:0]	RW	GPIO current data, each BIT corresponds to the corresponding GPIO line	32'h7304

9.4.3 GPIO Data Enable Register

Table 48 GPIOA Data Enable Registers

classifier for honorific people	interviews	Operating Instructions	reset value
[15:0]	RW	<p>BIT enable bit corresponding to GPIO_DATA, operation on the corresponding bit of GPIO_DATA is valid only when the corresponding BIT is 1.</p> <p>1. Each BIT corresponds to the corresponding GPIO line, 1'bx:</p> <ul style="list-style-type: none"> [x] = 0, GPIO_DATA[x] is not readable or writable [x] = 1, GPIO_DATA[x] can be read and written 	16'hffff

Table 49 GPIOB Data Enable Registers

classifier for honorific people	interviews	Operating Instructions	reset value

e			
[31:0]	RW	<p>BIT enable bit corresponding to GPIO_DATA, operation on the corresponding bit of GPIO_DATA is valid only when the corresponding BIT is</p> <p>1. Each BIT corresponds to the corresponding GPIO line, 1'bx:</p> <p>[x] = 0, GPIO_DATA[x] is not readable or writable</p> <p>[x] = 1, GPIO_DATA[x] can be read and written</p>	32'h7fff_ffff

9.4.4 GPIO Direction Control Register

Table 50 GPIOA Direction Control Registers

classifier for honorific people	interviews	Operating Instructions	reset value
[15:0]	RW	GPIO direction control, each BIT corresponds to the corresponding GPIO line, 1'bx:	16'h0

		[x]=0, GPIO[x] is the input [x]=1, GPIO[x] is an output	
--	--	--	--

Table 51 GPIOB Direction Control Registers

classifier for honorific people	interviews	Operating Instructions	reset value
[31:0]	RW	GPIO direction control, each BIT corresponds to the corresponding GPIO line, 1'bx: [x]=0, GPIO[x] is the input [x]=1, GPIO[x] is an output	32'h0

9.4.5 GPIO Pull-Up and Pull-Down Control Registers

Table 52 GPIOA Pull-Up Control Registers

classifier for honorific people	interviews	Operating Instructions	reset value
[15:0]	RW	GPIO pull-up control, each BIT corresponds to the corresponding GPIO line, 1'bx: note: this register is active low [x]=0, GPIO[x] has a pull-up	16'hffff

		[x] = 1, GPIO[x] no pull-ups	
--	--	------------------------------	--

classification for honorable people	interviews	Operating Instructions	reset value
[15:0]	RW	<p>GPIO pull-down control, each BIT corresponds to the corresponding GPIO line, 1'b1: note: this register is active high</p> <p>[x] = 1, GPIO[x] has a pull-down</p> <p>[x] = 0, no pull-down for GPIO[x]</p>	16'h0000

Table 53 GPIOB Pull-Up and Pull-Down Control Registers

classification for honor ific people	interviews	Operating Instructions	reset value
[31:0]	RW	<p>GPIO pull-up control, each BIT corresponds to the corresponding GPIO line, 1'bx: note: this register is active low</p> <p>[x] = 0, GPIO[x] has a pull-up</p> <p>[x] = 1, GPIO[x] no pull-ups</p>	32'hffff_ffff

classification for honor ific people	interviews	Operating Instructions	reset value
[31:0]	RW	<p>GPIO pull-down control, each BIT corresponds to the corresponding GPIO line, 1'bx: note: this register is active high</p> <p>[x] = 1, GPIO[x] has a pull-down</p> <p>[x] = 0, no pull-down for GPIO[x]</p>	32'h0000_0000

9.4.6 GPIO Multiplexing Select Register

Table 54 GPIOA Multiplexing Select Registers

classif ier for honor ific peopl e	intervi ews	Operating Instructions	reset value
[15:0]	RW	GPIO multiplexing enable bit, each BIT corresponds to whether the corresponding GPIO multiplexing function is turned on or not, 1'bx:	16'hfff

	<p>[x] = 0, GPIO[x] multiplexing off</p> <p>[x] = 1, GPIO[x] multiplexing turned on</p> <p>When [x] = 1, the multiplexing function depends on the status of the corresponding BITS of the GPIO_AF_S1 and GPIO_AF_S0 registers.</p> <p>S1.[x] = 0, S0.[x] = 0, multiplex function</p> <p>1(opt1) S1.[x] = 0, S0.[x] = 1, multiplex function</p> <p>2(opt2) S1.[x] = 1, S0.[x] = 0, multiplex function</p> <p>3(opt3) S1.[x] = 1, S0.[x] = 1, multiplex function</p> <p>4(opt4)</p> <p>[x] = 0, if GPIO_DIR[x] = 0 and GPIO_PULL_EN[x] = 1, then GPIO is multiplexed to opt6 a nalog IO function</p> <p>For IO multiplexing function, see Chip Pin Multiplexing Relationships.</p>	
--	--	--

Table 55 GPIOB Multiplexing Selection Registers

classification	interviews	Operating Instructions	reset value
for honorific people			

[31:0]	RW	<p>GPIO multiplexing enable bit, each BIT corresponds to whether the corresponding GPIO multiplexing function is turned on or not, 1'bx:</p> <p>[x] = 0, GPIO[x] multiplexing off</p> <p>[x] = 1, GPIO[x] multiplexing turned on</p> <p>When [x] = 1, the multiplexing function depends on the status of the corresponding BITS of the GPIO_AF_S1 and GPIO_AF_S0 registers.</p> <p>S1.[x] = 0, S0.[x] = 0, reuse function 1 (opt1)</p>	32'hffff_ffff
--------	----	--	---------------

		<p>S1.[x] = 0, S0.[x] = 1, multiplex function</p> <p>2(opt2) S1.[x] = 1, S0.[x] = 0, multiplex</p> <p>function 3(opt3) S1.[x] = 1, S0.[x] = 1,</p> <p>multiplex function 4(opt4)</p> <p>[x]=0, if GPIO_DIR[x]= 0 and GPIO_PULL_EN[x]=1, then GPIO is</p> <p>multiplexed to opt6 a n a l o g IO function</p> <p>For IO multiplexing functions, see Chip Pin Multiplexing Relationships.</p>	
--	--	--	--

9.4.7 GPIO multiplexing selection register 1

Table 56 GPIOA Multiplexing Select Register 1

classification	interviews	Operating Instructions	reset value
[15:0]	RW	<p>High address bit of GPIO multiplexing function select bit, together with GPIO_AF_S0, determines the multiplexing function.</p> <p>For IO multiplexing functions, see Chip Pin Multiplexing Relationships.</p>	16'h0

Table 57 GPIOB Multiplexing Selection Register 1

classification	interviews	Operating Instructions	reset value

[31:0]	RW	High address bit of GPIO multiplexing function select bit, together with GPIO_AF_S0, determines the multiplexing function. For IO multiplexing functions, see Chip Pin Multiplexing Relationships.	32'h0
--------	----	---	-------

9.4.8 GPIO multiplexing selection register 0

Table 58 GPIOA Multiplexing Select Register 0

classifier for honorific people	interviews	Operating Instructions	reset value
[15:0]	RW	<p>GPIO multiplexing function select bit low address bit, together with GPIO_AF_S1 determines the multiplexing function</p> <p>See GPIO_AF_SEL register description for how to configure it</p>	16'h0

Table 59 GPIOB Multiplexing Select Register 0

classifier for honorific people	interviews	Operating Instructions	reset value
[31:0]	RW	<p>GPIO multiplexing function select bit low address bit, together with GPIO_AF_S1 determines the multiplexing function</p> <p>See GPIO_AF_SEL register description for how to configure it</p>	32'h0

9.4.9 GPIO Interrupt Trigger Mode Configuration Register

Table 60 GPIOA Interrupt Trigger Mode Configuration Registers

classifier for honorific people	interviews	Operating Instructions	reset value

people			
[15:0]	RW	<p>Interrupt triggering method for GPIOs, each BIT corresponds to the corresponding GPIO line, 1'bx:</p> <p>[x] = 0, GPIO[x] interrupt is edge triggered</p> <p>[x] = 1, GPIO[x] interrupt is level triggered</p>	16'h0

Table 61 GPIOB Interrupt Trigger Mode Configuration Registers

classifier for honorific people	intervews	Operating Instructions	reset value
[31:0]	RW	<p>Interrupt triggering method for GPIOs, each BIT corresponds to the corresponding GPIO line, 1'bx:</p> <p>[x] = 0, GPIO[x] interrupt is edge triggered</p>	32'h0

		[x] = 1, GPIO[x] interrupt is level triggered	
--	--	---	--

9.4.10 GPIO Interrupt Edge Trigger Mode Configuration Registers

Table 62 GPIOA Interrupt Edge Trigger Mode Configuration Registers

classif ier for honor ific peopl e	intervi ews	Operating Instructions	reset value
[15:0]	RW	GPIO interrupt edge-triggered mode, each BIT corresponds to the corresponding GPIO line, 1'bx: [x] = 0, GPIO[x] edge-triggered interrupt mode determined by GPIO_IEV [x] = 1, interrupt triggered on both GPIO[x] edges	16'h0

Table 63 GPIOB Interrupt Edge Trigger Mode Configuration Registers

classif ier for honor ific peopl e	intervi ews	Operating Instructions	reset value
[31:0]	RW	GPIO interrupt edge-triggered mode, each BIT corresponds to the corresponding GPIO line, 1'bx: [x] = 0, GPIO[x] edge-triggered interrupt mode determined by GPIO_IEV [x] = 1, interrupt triggered on both GPIO[x] edges	32'h0

9.4.11 GPIO interrupt upper and lower edge trigger configuration registers

Table 64 GPIOA Interrupt Top and Bottom Edge Trigger Configuration Registers

classification for honorific people	interviews	Operating Instructions	reset value
[15:0]	RW	<p>GPIO interrupt upper and lower edge-triggered or high and low level-triggered selections, each BIT corresponds to the corresponding GPIO line,</p> <p>1'bx:</p> <p>[x] = 0, GPIO[x] interrupt is low or falling edge triggered</p> <p>[x] = 1, GPIO[x] interrupt is triggered high or on rising edge</p>	16'h0

Table 65 GPIOB Interrupt Top and Bottom Edge Trigger Configuration Registers

classification for honorific people	interviews	Operating Instructions	reset value
[31:0]	RW	<p>GPIO interrupt upper and lower edge-triggered or high and low level-triggered selections, each BIT corresponds to the corresponding GPIO line, 1'bx:</p> <p>[x] = 0, GPIO[x] interrupt is low or falling edge triggered</p> <p>[x] = 1, GPIO[x] interrupt is triggered high or on rising edge</p>	32'h0

9.4.12 GPIO Interrupt Enable Configuration Register

Table 66 GPIOA Interrupt Enable Configuration Registers

classification for honorific people	interviews	Operating Instructions	reset value
[15:0]	RW	<p>GPIO interrupt enable control, each BIT corresponds to the corresponding GPIO line, 1'bx:</p> <p>[x] = 0, GPIO[x] interrupt disabled</p> <p>[x] = 1, GPIO[x] interrupt enable</p>	16'h0

Table 67 GPIOB Interrupt Enable Configuration Registers

classification for honorific people	interviews	Operating Instructions	reset value
[31:0]	RW	<p>GPIO interrupt enable control, each BIT corresponds to the corresponding GPIO line, 1'bx:</p> <p>[x] = 0, GPIO[x] interrupt disabled</p> <p>[x] = 1, GPIO[x] interrupt enable</p>	32'h0

9.4.13 GPIO Bare Interrupt Status Register

Table 68 GPIOA Bare Interrupt Status Registers

classification for honor ific people	interviews	Operating Instructions	reset value
[15:0]	RW	<p>GPIO bare interrupt state (before MASK), 1'bx per BIT corresponding to the corresponding GPIO line:</p> <p>[x] = 0, no interrupt generation for GPIO[x]</p> <p>[x] = 1, GPIO[x] has interrupt generation</p>	16'h0

Table 69 GPIOB Bare Interrupt Status Registers

classification for honor ific people	interviews	Operating Instructions	reset value
[31:0]	RW	<p>GPIO bare interrupt state (before MASK), 1'bx per BIT corresponding to the corresponding GPIO line:</p> <p>[x] = 0, no interrupt generation for GPIO[x]</p> <p>[x] = 1, GPIO[x] has interrupt generation</p>	32'h0

9.4.14 GPIO Masked Interrupt Status Register

Table 70 GPIOA Interrupt Status Register after Masking

classifier for honorific people	interviews	Operating Instructions	reset value
[15:0]	RW	<p>GPIO interrupt state after mask (after MASK), 1'bx per BIT corresponding to the corresponding GPIO line:</p> <p>[x] = 0, no interrupt generation for GPIO[x] (after MASK)</p> <p>[x] = 1, GPIO[x] interrupt generation (after MASK)</p>	16'h0

Table 71 Interrupt status registers after GPIOB masking

classification	interviews	Operating Instructions	reset value
[31:0]	RW	<p>GPIO interrupt state after mask (after MASK), 1'bx per BIT corresponding to the corresponding GPIO line:</p> <p>[x] = 0, no interrupt generation for GPIO[x] (after MASK)</p> <p>[x] = 1, GPIO[x] interrupt generation (after MASK)</p>	32'h0

9.4.15 GPIO Interrupt Clear Control Register

Table 72 GPIOA Interrupt Clear Control Registers

classification	interviews	Operating Instructions	reset value
[15:0]	RW	<p>GPIO interrupt clear control, 1'bx per BIT corresponding to the corresponding GPIO line:</p> <p>[x] = 0, no action</p> <p>[x] = 1, clear GPIO[x] interrupt status</p>	16'h0

Table 73 GPIOB Interrupt Clear Control Registers

classification	interviews	Operating Instructions	reset value

people			
[31:0]	RW	GPIO interrupt clear control, 1'bx per BIT corresponding to the corresponding GPIO line: [x] = 0, no action [x]=1, clear GPIO[x] interrupt status	32'h0

10 High Speed SPI Device Controller

10.1 Functional overview

Compatible with the universal SPI physical layer protocol, through the agreed data format for interaction with the host, the host can provide high-speed access to the device, the highest supported operating frequency is 50MHZ.

10.2 Main characteristics

- Compatible with universal SPI protocol
- Selectable level interrupt signal
- Supports up to 50Mbps
- Simple frame format with full hardware parsing and DMA

10.3 Functional Description

10.3.1 Introduction to SPI Protocol

SPI works in a master-slave mode, usually with one master device and one or more slave devices, and requires at least 4 wires, in fact 3 are possible (for unidirectional transfers) They are SDI (data in) SDO (data out) SCLK (clock) CS (chip select)

- (1) SDI - Serial Data In, Serial Data Input
- (2) SDO - Serial Data Out
- (3) SCLK - Serial Clock, clock signal, generated by master device
- (4) CS - Chip Select, slave device enable signal, controlled by the master device.

Where CS is the control signal for whether the slave chip is selected by the master chip, i.e., only if the chip select signal is a pre-specified enable signal (high).

This makes it possible to connect several SPI devices on the same bus. This makes it possible to connect several SPI devices on the same bus.

In addition to the above four signal lines, HSPI also adds an extra INT line to generate a falling edge interrupt when the slave device has data to be uploaded, so as to realize the active uploading of data.

SPI communication is accomplished by exchanging data, which is transferred one bit at a time, with a clock pulse provided by SCLK, and SDI, SDO accomplishing the data transfer based on this pulse. The data output is through the SDO line, and the data is changed on the rising or falling edge of the clock, and is read on the immediately following falling or rising edge. To complete a one-bit data transfer, the same principle is used for the inputs. Therefore, at least 8 changes of the clock signal are required (upper edge).

and the lower edge once in order to complete the transmission of 8-bit data.

The SCLK signal line is controlled by the master device; slave devices cannot control the signal line. In an SPI-based device, there is at least one master device.

10.3.2 SPI Work Process

The HSPI inside the chip works together with the wrapper controller, which integrates DMA and realizes the data exchange between the FIFO inside the HSPI and the internal cache of the chip through DMA. The operation is realized in hardware, the software does not need to care about the process of sending and receiving data, only need to configure the send and receive data chain table, as well as the operation of the corresponding registers of the wrapper controller.

For more information about the wrapper controller, please refer to the relevant section.

10.4 register description

10.4.1 List of registers operated inside the HSPI chip

Table 74 HSPI Internal Access Registers

offset address	name (of a thing)	abridge	inter views	descriptive	reset value

0X0000	HSPI FIFO Flush Registers	CLEAR_FIFO	RW	Clears the contents of the Tx and Rx FIFOs while the Circuits that will synchronize the reset system clock domain	0X0000_0000
0X0004	HSPI Configuration Register	SPI_CFG	RW	Configure the SPI transfer mode and size termination. install	0X0000_0000
0X0008	HSPI Mode Configuration Register	MODE_CFG	RW	Configure the ahb master to access the bus when the burst length	0X0000_0000
0X000C	HSPI Interrupt Configuration Register	SPI_INT_CPU_ MASK	RW	Configure whether interrupts are enabled or not	0X0000_0003
0X0010	HSPI Interrupt Status Register	SPI_INT_CPU_ STTS	RW	Getting and clearing interrupt status	0X0000_0000
0X0018	HSPI Data Upload Length Register	RX_DAT_LEN	RW	Configure the length of data that can be uploaded	0X0000_0000

10.4.1.1 HSPI FIFO Flush Registers

Table 75 HSPI Clear Registers FIFO

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31: 1]	RO	reservations	

[0]	RW	<p>Clear FIFOs, clears the contents of the Tx and Rx FIFOs, and also synchronously resets the circuitry in the system clock domain (except the registers in this list)</p> <p>0: FIFO not cleared</p> <p>1: Clearance effective</p> <p>Software set, hardware clear</p> <p>Note: To reset the entire circuit, the asynchronous reset leg of this module is required: <code>rst_n</code></p>	1'b0
-------	----	---	------

10.4.1.2 HSPI Configuration Register

Table 76 HSPI Configuration Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31: 4]	RO	reservations	
[3]	RW	The Bigendian, spi interface supports the selection of the small and large end of the data. 0: Small segment data transfer is supported 1: Support for large-end data transfer	1'b0
[2]	RW	spi_tx_always_drive 0: spi output is valid only when chip select is active, otherwise it is high resistance. 1: spi output is always active	1'b0
[1]	RW	SPI CPHA 0: Transmission mode A 1: Transmission mode B	1'b0
[0]	RW	SPI CPOL, SCK polarity at IDLE 0: 0 at SCK IDLE 1: 1 when SCK IDLE	1'b0

10.4.1.3 HSPI Mode Configuration Register

Table 77 HSPI Mode Configuration Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31: 1]	RO	reservations	

[0]	RW	Burst len, the length of the burst when the ahb master accesses the bus 0: burst len is 1 word 1: burst len is 4 characters It is recommended to set it to a 4-word burst transfer, so that when the frequency of the spi interface is high, it can be guaranteed not to flow continuously	1'b0
------	----	--	------

10.4.1.4 HSPI Interrupt Configuration Register

Table 78 HSPI Interrupt Configuration Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31: 2]	RO	reservations	
[1]	RW	IntEnRxOverrun, RxOverrun interrupt enable 0: Rx FIFO overflow interrupt enable 1: Rx FIFO overflow interrupts do not make the	1'b1
[0]	RW	IntEnTxUnderrun, TxUnderrun Interrupt Enable 0: Tx FIFO underflow Interrupt Enable 1: Tx FIFO underflow interrupt not enabled	1'b1

10.4.1.5 HSPI Interrupt

Status Registers

Table 79 HSPI Interrupt Status Registers

classifier	intervi	Operating Instructions	reset value
-------------------	----------------	-------------------------------	--------------------

for honorifi c people	ews		
[31: 2]	RO	reservations	
[1]	RW	RxOverrun 0: Rx FIFO overflow	1'b0

		1: Rx FIFO overflow Write 1 Clear	
[0]	RW	TxUnderrun 0: Tx FIFO underflow 1: Tx FIFO underflow write 1 cleared	1'b0

10.4.1.6 HSPI Data Upload Length Register

Table 80 HSPI Data Upload Length Registers

classifier for honori- c people	intervi- ews	Operating Instructions	reset value
[31:16]	RO	reservations	
[15:0]	RW	Rx_dat_len Indicates the length of data that can be uploaded, in bytes The upload lengths are all integer multiples of words. If the upload length is less than a whole word, round up.	16'h0

10.4.2 Host-side Access to HSPI Controller Register List

The host side accesses the SPI interface registers through a fixed SPI command format. The command length is fixed to one byte and the data length is fixed to two bytes.

Table 81 HSPI Interface Configuration Registers (Master Device Access)

misal ignm ent	name (of a thing)	abridge	inter views	descriptive	reset value
addr ess					

0X02	Get Data Length Register	RX_DAT_LEN	RO	When uploading data, the spi host is used to get the data from the Length of data read on the device side	0X0000
0X03	Downlink Data Flag Register	TX_BUFF_AVAIL	RO	When sending data from master to slave, it is used to determine whether it is possible to Upload data or commands	0X0000
0X04	reservations	RSV	RO		
0X05	Interrupt Configuration Register	SPI_INT_HOST_MASK	RW	Whether or not to mask interrupts	0X0000
0X06	Interrupt Status Register	SPI_INT_HOST_STTS	RO	Interrupt status register, spi host queries this bit Confirmation of data upload	0X0000
0X07	reservations	RSV	RO		
0X00	Data Port 0	DAT_PORT0	RW	The Spi host sends down data to the slave device through this register port, and the data frame preceding the downlink uses this ports	
0X10	Data Port 1	DAT_PORT1	RW	The Spi host sends a message to the slave device through this register port. Downlink data, downlink the last data frame using this port	
0X01	Command port 0	DN_CMD_PORT0	WO	The Spi host sends command data to the slave device through this register, sending the previous	

				command data using the this port	
0X11	Command port 1	DN_CMD_PORT1	WO	The Spi host sends the slave device down through this register Command data, send the last frame of command data using this port	

10.4.2.1 HSPI Get Data Length Register

Table 82 HSPI Get Data Length Registers

classifier for honorific people	intervie ws	Operating Instructions	reset value
[15:0]	RO	<p>spi host read-only register, mainly used to know how much data to read from the device side when uploading data.</p> <p>However, in this module, the upload length is an integer multiple of the word. If this upload length value is not an integer word, the host reads the number rounded up, i.e., it reads some more redundant bytes.</p>	16'h0

10.4.2.2 HSPI Downstream Data Flag Register

Table 83 HSPI Downstream Data Flag Registers

classifier for honorific people	intervie ws	Operating Instructions	reset value
[15: 2]	RO	reservations	
[1]	RO	<p>tx_cmdbuff_avail</p> <p>Flag whether the buff for sending cmd is available, if it is available, the host can send cmd. 0: send buff not available</p> <p>1: Send buffs available</p>	1'b0

[0]	RO	tx_buff_avail Flags whether the send buff is available, and if so, the host can send down data. 0: Send buff not available 1: Send buffs available	1'b0
-----	----	---	------

10.4.2.3 HSPI Interrupt Configuration Register

Table 84 HSPI Interrupt Configuration Registers

classifier for honorific people	intervie ws	Operating Instructions	reset value
[15: 1]	RO	reservations	
[0]	RO	<p>IntMaskup_dat_cmd_rdy</p> <p>interrupt masking</p> <p>0: Interrupts are not masked and can be generated</p> <p>1: Interruptions are blocked</p> <p>Note: It is recommended to use the host's own internal interrupt masking to improve efficiency.</p>	1'b0

10.4.2.4 HSPI Interrupt Status Register

Table 85 HSPI Interrupt Status Registers

classifier for honorific people	intervie ws	Operating Instructions	reset value
[15: 1]	RO	reservations	
[0]	RO	<p>up_dat_cmd_rdy</p> <p>Status register for generating an interrupt to the SPI host 0: data or command not ready</p> <p>1: Data or command is ready legible</p>	1'b0

10.4.2.5 HSPI Data Port 0

Table 86 HSPI Data Ports 0

classifier for honorific people	intervie ws	Operating Instructions	reset value
	RW	<p>The SPI host transmits data to and from the device through this register port.</p> <p>Write to this register to send data down, read from this register to upload data.</p> <p>If the frame being transmitted needs to go through multiple transmissions to complete the</p> <p>If the last transmission is successful, the register port DAT_PORT1 is used for the last transmission, and DAT_PORT0 is used for the others.</p>	

10.4.2.6 HSPI Data Port 1

Table 87 HSPI Data Ports 1

classifier for honorific people	intervie ws	Operating Instructions	reset value
	RW	<p>The SPI host transmits data to and from the device through this register port.</p> <p>Write to this register to send data down, read from this register to upload data.</p> <p>If the frame being transmitted needs to go through multiple transmissions to finish the</p> <p>If the last transmission is successful, the register port DAT_PORT1 is used for the last transmission, and DAT_PORT0 is used for the others.</p>	

10.4.2.7 HSPI Command Port 0

Table 88 HSPI Command Ports 0

classifier for honorific	intervie ws	Operating Instructions	reset value

people			
	RW	<p>The SPI host interacts with the device through this register port by writing numbers to this register to issue commands. If the command being transmitted requires multiple transmissions to complete, the last transmission uses register port DN_CMD_PORT1, and the others use DN_CMD_PORT0.</p> <p>Note: This window is only used to issue commands for driver and firmware negotiation.</p>	

10.4.2.8 HSPI Command Port 1

Table 89 HSPI Command Ports 1

classifier for honorific people	intervie ws	Operating Instructions	reset value
	RW	<p>The SPI host interacts with the device through this register port by writing numbers to this register to issue commands. If the command being transmitted requires multiple transmissions to complete, the last transmission uses register port DN_CMD_PORT1, and the others use DN_CMD_PORT0.</p> <p>Note: This window is only used to issue commands for driver and firmware negotiation.</p>	

10.4.3 High-Speed SPI Device Controller Interface Timing

It mainly describes the SPI read/write timing and how the master SPI interacts with the HSPI.

10.4.3.1 data format

The data format is divided into two parts: command field and data field, as shown below. The fixed length of the command field is 8 bits, and the length of the data field is different depending on the access object, see below.

The highest bit of the command field is the read/write flag bit and the remaining 7 bits are the address.

- 0 means read data from the next 7bit address.

-
- 1 Indicates writing data to the next 7-bit address.

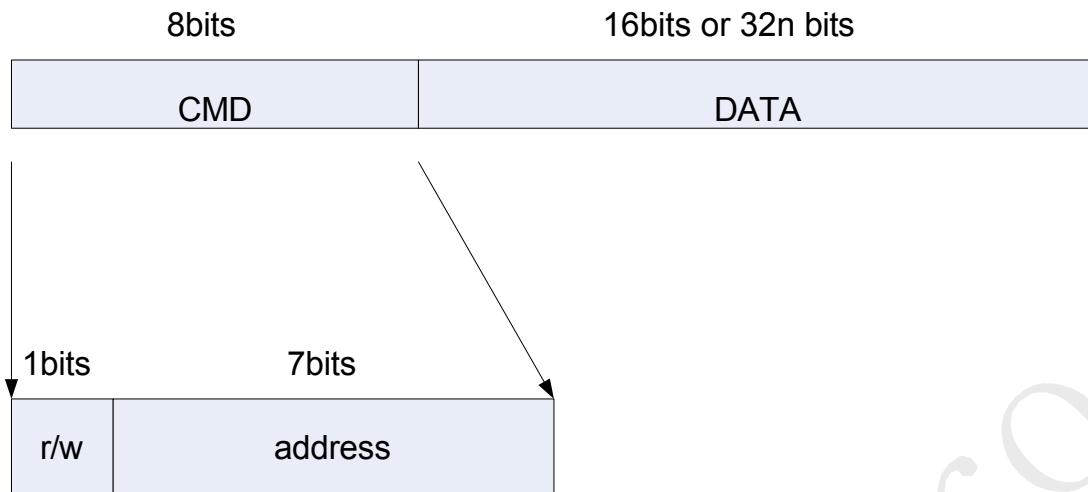


Fig. 5 Upper computer SPI send/receive data format

The data field of this module supports only two lengths, the upper SPI access interface configuration registers (Table 2) and the data field length is 16bit;

Transmits data through the ports (Data Port 0, Data Port 1, Command Port 0 and Command Port 1) with a 32-bit integer data field length

Tim
es;

The following figure shows the timing diagram for reading and writing the interface configuration registers. The default configuration of the slave device is small end mode.

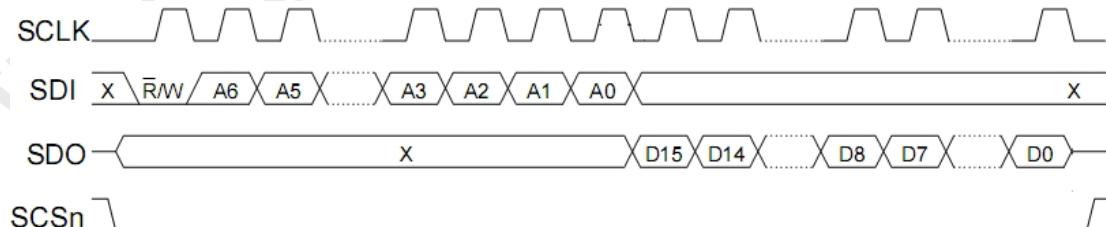


Fig. 6 HSPI register read operation (big-end mode)

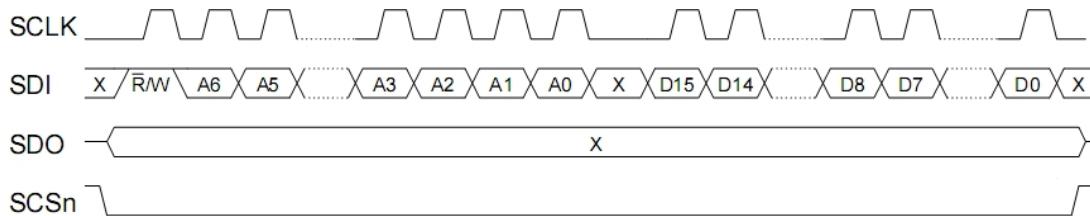


Fig. 7 HSPI Register Write Operation (Big End Mode)

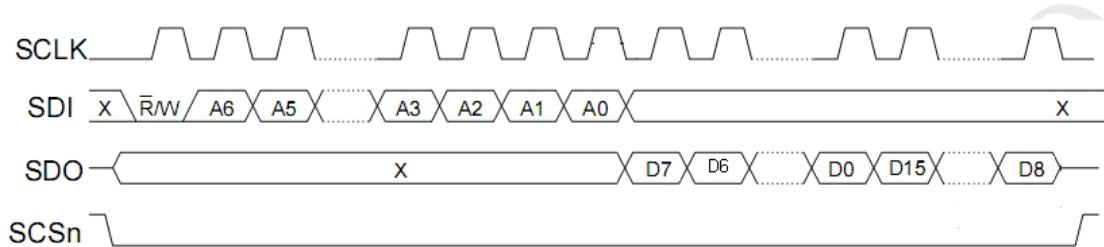


Fig. 8 Register read
operation (small end mode)

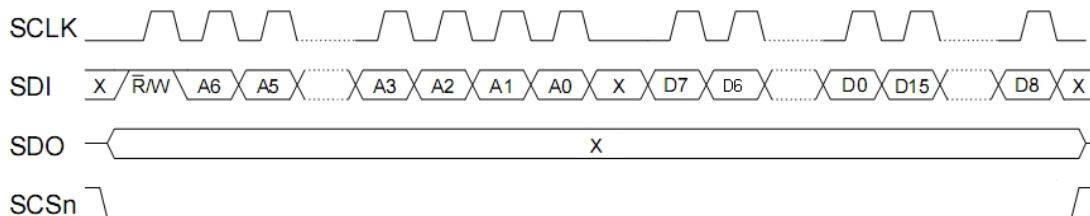


Fig. 9 Register write
operation (small end mode)

The following figure shows the timing diagram for reading and writing data. The length of the data field is an integer multiple of 32bit, and the figure shows that only one word is transmitted.

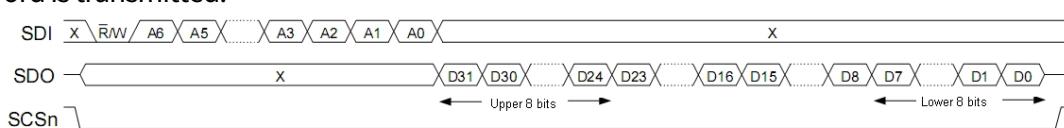


Figure 10 Port Read Operation (Big End Mode)

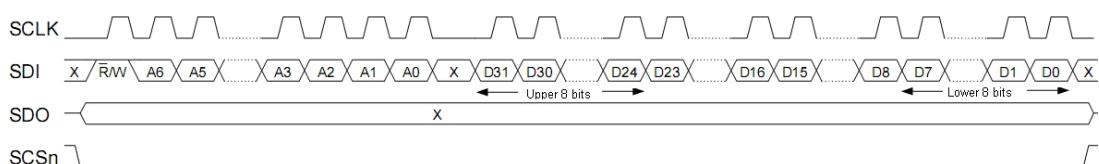


Figure 11 Port Write Operation (Big End Mode)

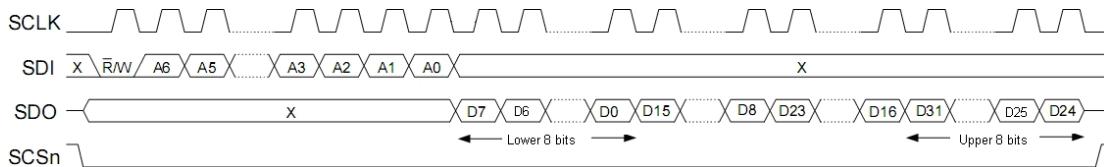


Figure 12 Port Read
Operation (Small End
Mode)

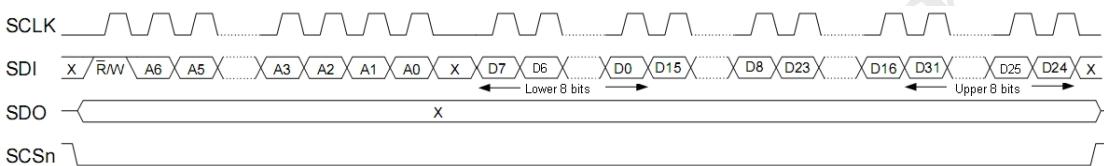


Figure 13 Port Write
Operation (Small End
Mode)

Note: There can be no wait time between commands and data, i.e., the transfer of a command field can be followed immediately by a data transfer without excess idle clocks or idle time. It is also possible to have a time delay, but no idle clocks can occur.

10.4.3.2 chronology

This module supports half-duplex, and the timings that can be supported are categorized into 4 types depending on the clock phase and sampling point. The following timings just give the phase and sampling relationship of the clock. Note that the chip supports (CPOL=0,CPHA=0) by default.

Note: There can be no wait time between commands and data, i.e., the transfer of a command field can be followed immediately by a data transfer without excess idle clocks or idle time. It is also possible to have a time delay, but no idle clocks can occur.

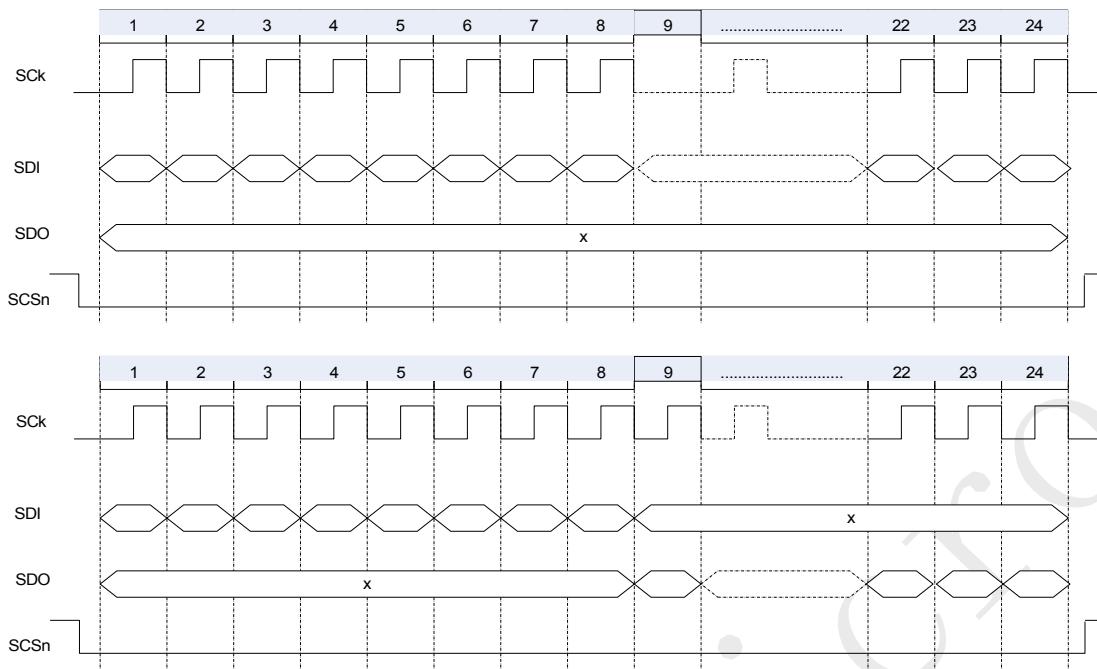


Figure 14
 $\text{CPOL}=0, \text{CPHA}=0$

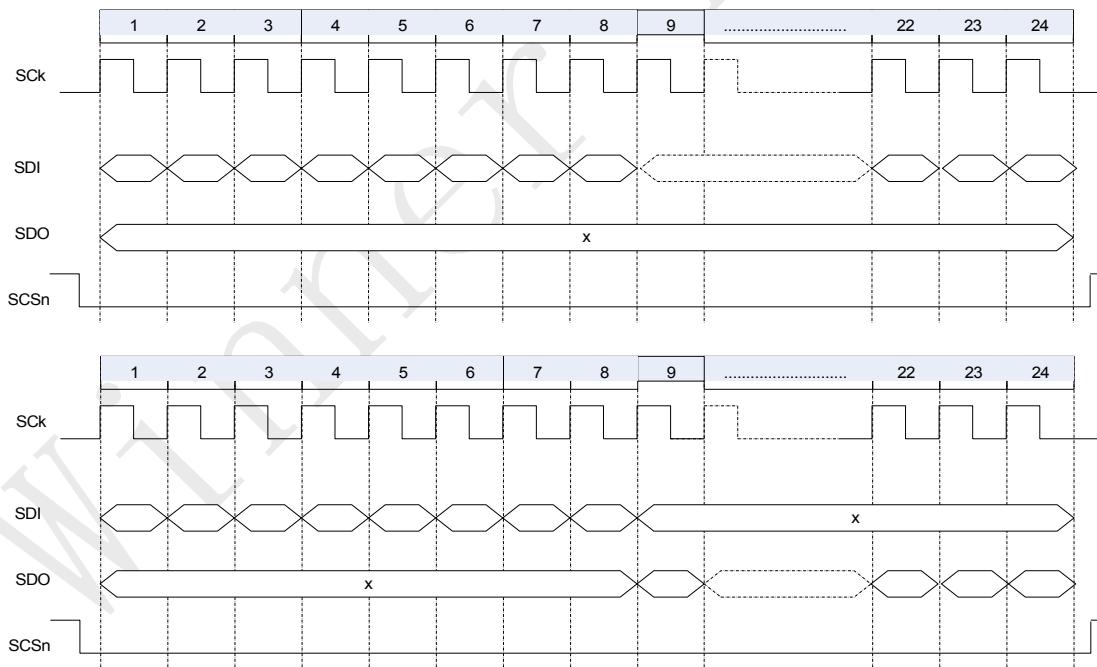


Figure 15 $\text{CPOL}=0, \text{CPHA}=1$

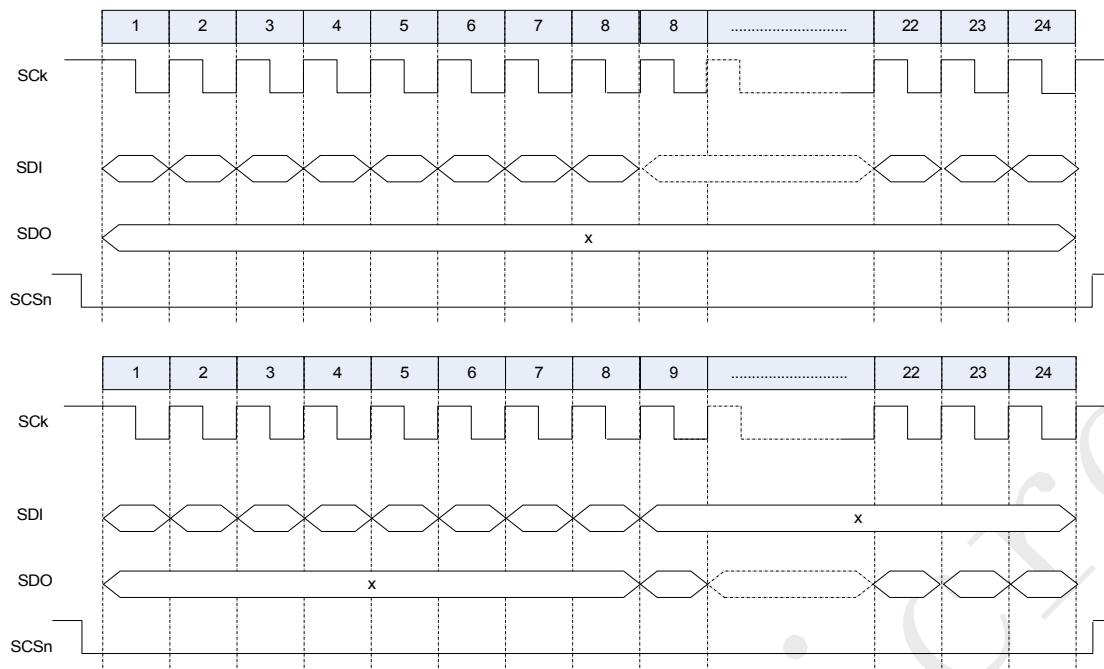


Figure 16
CPOL=1,CPHA=0

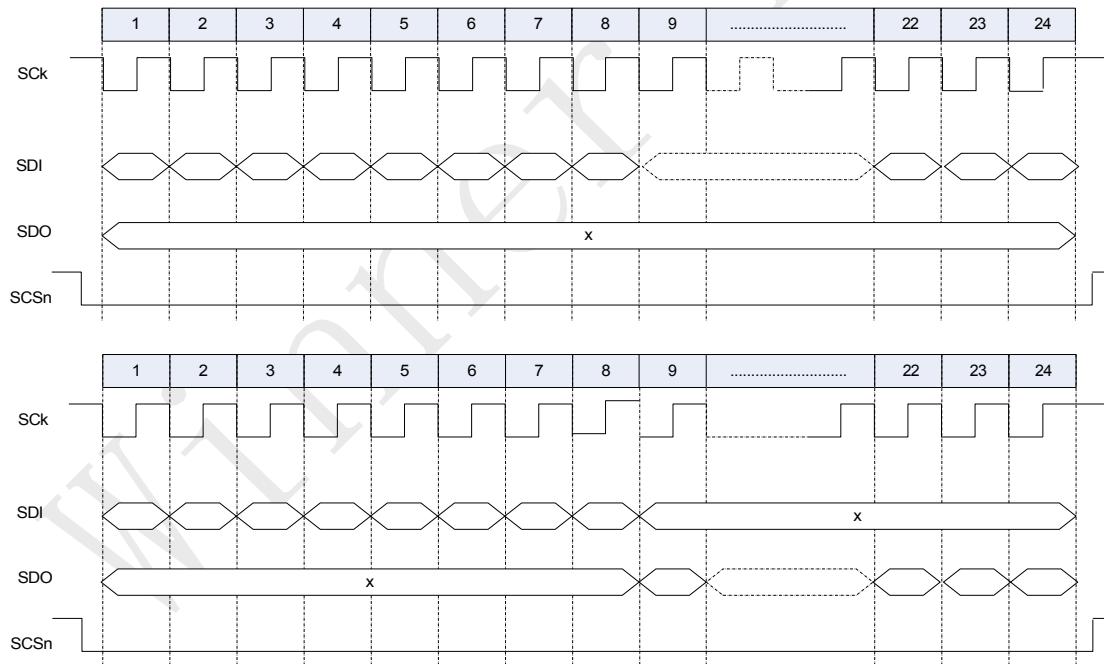


Figure 17 CPOL=1,CPHA=1

10.4.3.3 disruptions

The interrupt signal is sent from the slave device to the master device and is triggered by the SPI_INT pin,

active low.

spi_int Mainly notifies the spi host that there is data or commands to be uploaded. the interface registers that the spi host cares about when handling interrupts are:

- SPI_INT_HOST_MASK
- SPI_INT_HOST_STTS
- RX_DAT_LEN

Note: Each uploaded frame corresponds to an interrupt. Only after the transmission of the current frame to be uploaded is completed, if there are still frames to be uploaded, at that time, a new interrupt will be generated. The following figure shows one way to handle interrupts.

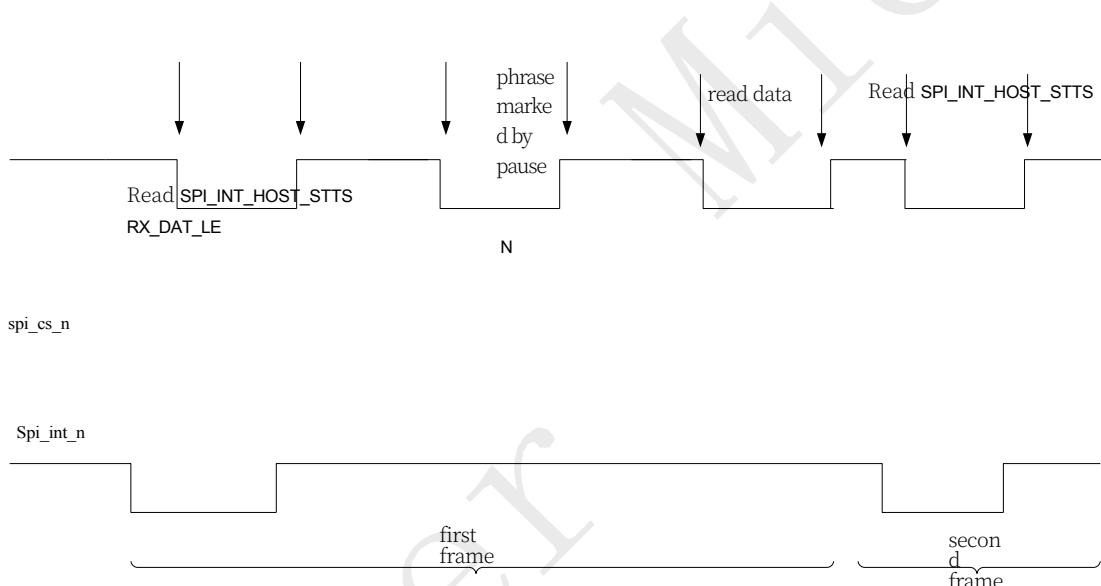


Figure 18 Main SPI
interrupt handling flow

10.4.3.4 Main SPI Send and Receive Data Workflow

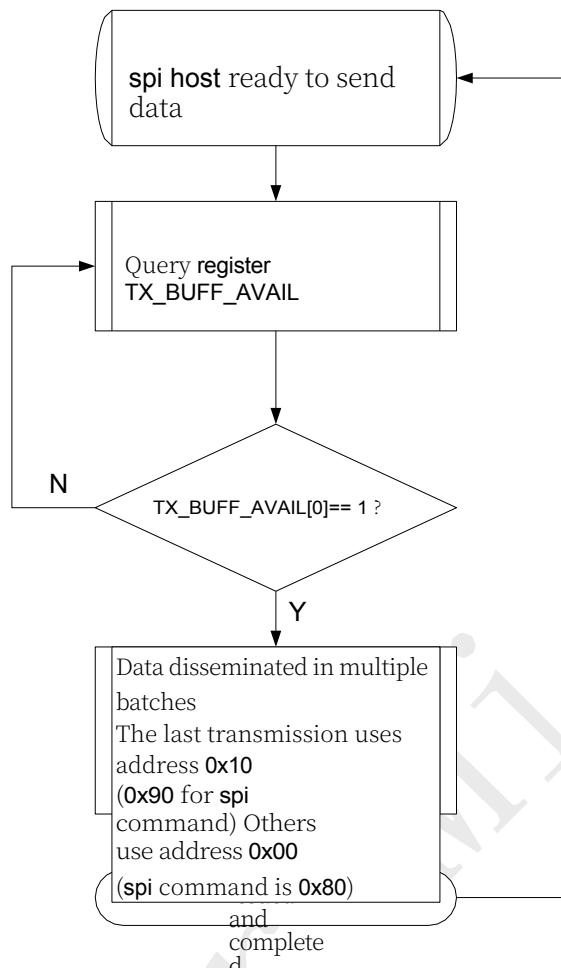


Figure 19 Downstream data flow chart

Note: The length of the data being sent down must be in words, if it is not a whole word, fill in 0 to make up the difference.

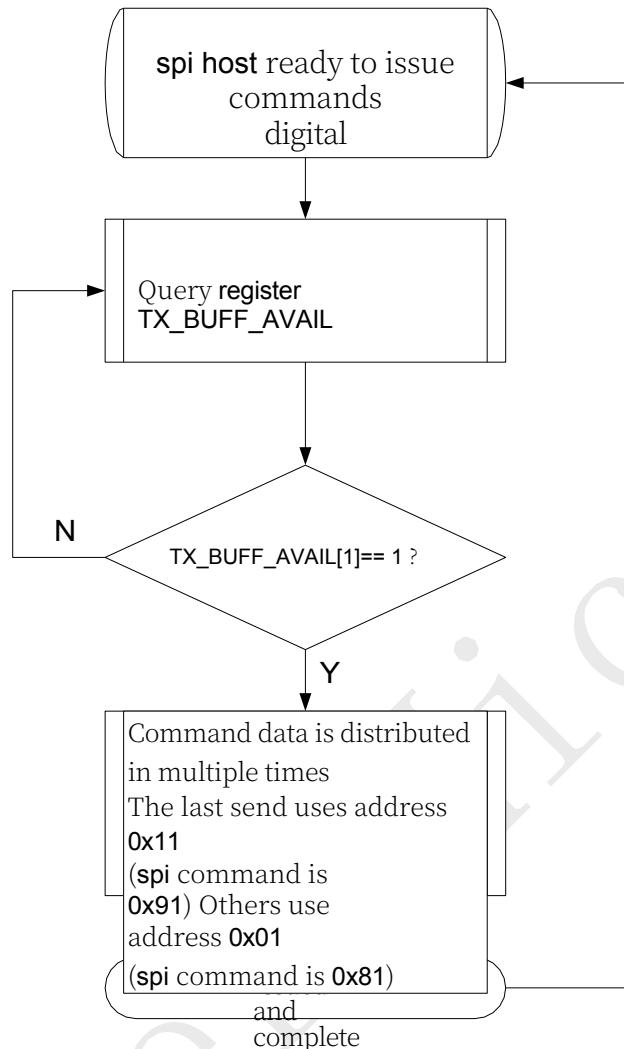


Figure 20
Downstream
Command Flowchart

Note: The length of the command issued must be in words; if it is not a whole word, fill in 0 to compensate.

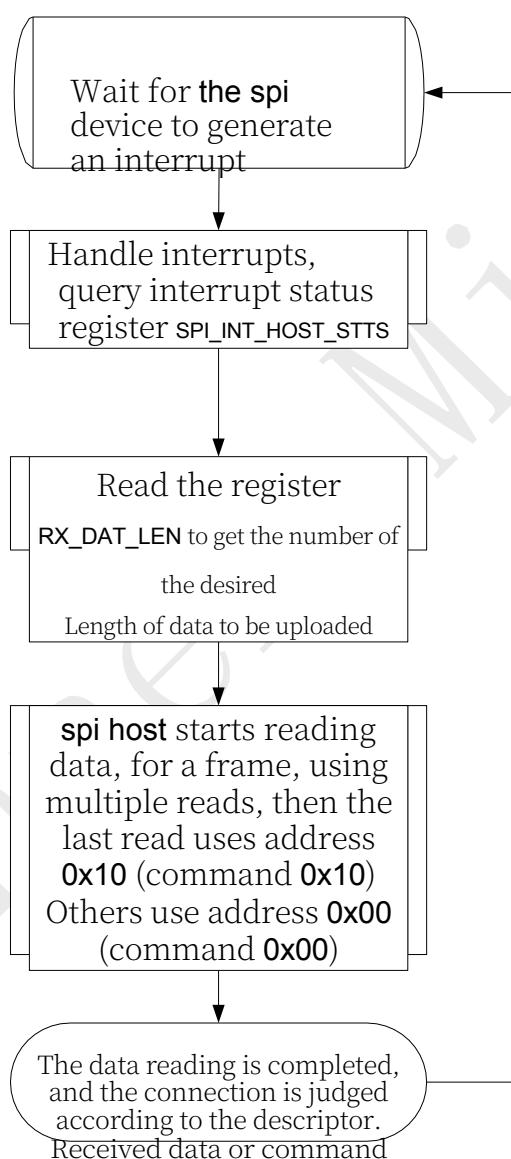


Figure 21 Uplink Data (Command) Flowchart

The process is the same for uplinked data and uplinked commands.

It should be noted here that the length of the data on the upstream line must be in words, if the effective

length is not a whole word, the excess data at the end can be thrown

Drop.

It should be noted that there are two channels, data channel and command channel, for data interaction between master and slave, and users can choose to use one channel or both according to their needs. The maximum data length for one interaction on the command channel is 256 bytes, and that for one interaction on the data channel is 1500 bytes. The data length limit is controlled by the slave device, and if the length is exceeded, it will destroy

the data structure of the slave device.

11 SDIO Device

Controller

11.1 Functional overview

W800 integrates SDIO device side interface as a slave device to complete the interaction with the host data. A 1024byte asynchronous FIFO is integrated internally to complete the data interaction between the host and the chip.

11.2 Main characteristics

- Compatible with SDIO Card Specification 2.0
- Support host rate 0~50MHz
- Supports blocks up to 1024 bytes
- Supports 1-bit SD and 4-bit SD modes

11.3 Functional Description

11.3.1 SDIO Bus

SDIO bus is similar to USB bus, SDIO bus also has two ends, one of which is the host end and the other end is the device end, the design of HOST-DEVICE is to simplify the design of DEVICE, and all the communication starts from the commands issued by the HOST end. As long as the DEVICE side can parse the commands from the HOST, it can communicate with the HOST, and the HOST of SDIO can connect to multiple DEVICES.

In the SDIO bus definition, the DAT1 signal line is multiplexed as an interrupt line. In the 1BIT mode of SDIO, DAT0 is used to transfer data, and DAT1 is used as the interrupt line. In 4BIT mode of SDIO, DAT0-DAT3 are used to transfer data, where DAT1 is multiplexed as the interrupt

11 SDIO Device
Controller

11.3.2 SDIO

Commands

On the SDIO bus, the HOST side initiates the request and then the DEVICE side responds to the request, where the request and response will contain data information:

- Command: The command used to start the transmission is sent from the HOST side to the DEVICE side, where the command is transmitted via the CMD signaling line;
- Response: The response is returned by DEVICE as a response to Command. It is also sent through the CMD line;
- Data: Data is transmitted in both directions. It can be set to 1-wire mode or 4-wire mode.
Data is transmitted via the DAT0- DAT3 signal lines.

Each operation of SDIO is initiated by the HOST with a CMD on the CMD line, and for some CMDs the DEVICE needs to return a Response, and for others, it does not.

For the read command, first HOST will send a command to DEVICE, and then DEVICE will return a handshake signal, at this time, when HOST receives the handshake signal in response, it will put the data on the 4-bit data line, and will follow the CRC checksum while transmitting the data. When the whole read transmission is finished, HOST will send a command again to inform DEVICE that the operation is finished, and DEVICE will return a response at the same time.

For the write command, first HOST will send a command to DEVICE, and then DEVICE will return a handshake signal, at this time, when HOST receives the handshake signal in response, it will put the data on the 4-bit data line, and will follow the CRC checksum while transmitting the data. When the whole write transfer is finished, HOST will send a command again to inform DEVICE that the operation is finished, and DEVICE will return a response at the same time.

11.3.3 SDIO Internal Storage

11.3.2 SDIO

SDIO devices have a fixed internal storage mapping, including a general information area (CIA) and a function unique area.

The registers in the CIA include I/O port functions, interrupt generation, and port operation information, and the registers defined in the CIA can be operated by read/write function 0. The CIA contains three types of information, CCCR, FBR, and CIS. The CIA contains three types of information: CCCR, FBR, and CIS.

The host side can check the SDIO card and operate the ports by operating the CCCR, address of CCCR is 0X00-.
 0xFFFFBR defines the operation of supported port functions 1 to 7, including the requirements and functions of each port, power control, etc., and the address of FBR is 0Xn00-0Xnff (where n is the function port number) CIS defines some information structure of the card, and the address is 0X1000-0X17FFF, and there are public CIS and the respective CIS for each function port. The CIS has a public CIS and the respective CIS of each function port, where the initial address of the public CIS is shown below. The storage mapping for CIA is shown below.

CIS is in the CIS Pointer field of the CCCR, and the CIS of each port function is in the CIS Pointer field of each function port FBR.

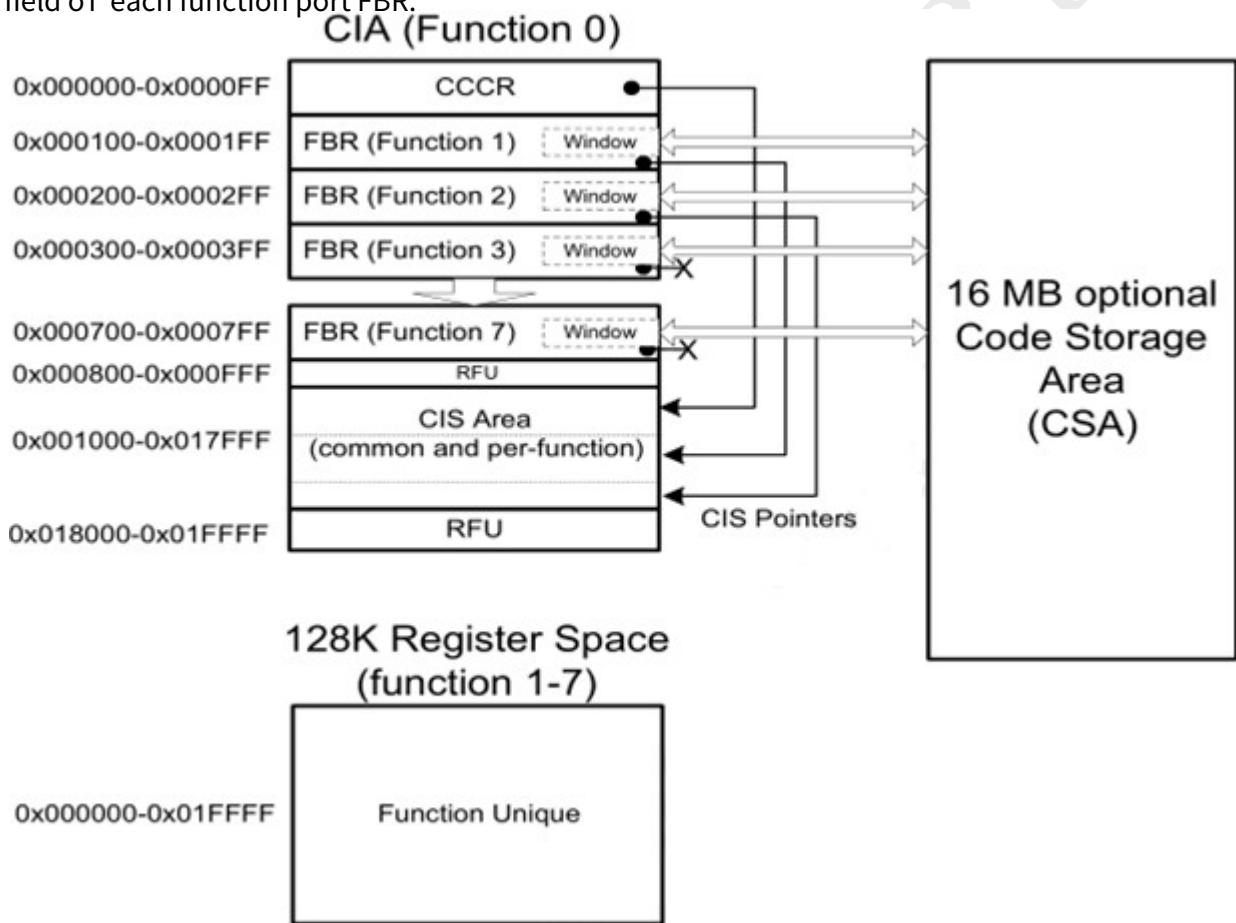


Figure 22 SDIO Internal Storage Mapping

The host side can check the SDIO card and operate the ports by operating the CCCR. The address of CCCR is 0X00. Refer to the following for a description of each register in the CIA. For an in-depth understanding of the CIA, refer to the SDIO protocol specification.

11.4 register description

11.4.1 register list

11.4.2 SDIO Fn0 register

Fn0 register is the register specified by SDIO protocol, its address range is 0x00000~0x1FFF, total 128K. the start address is 0x00000.

The Fn0 register is accessed by the SDIO host via the CMD52 command, the offset address is the access address and the function number is 0.

Address	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x00	CCCR/SDIO Revision	SDIO bit 3	SDIO bit 2	SDIO bit 1	SDIO bit 0	CCCR bit 3	CCCR bit 2	CCCR bit 1	CCCR bit 0
0x01	SD Specification Revision	RFU	RFU	RFU	RFU	SD bit 3	SD bit 2	SD bit 1	SD bit 0
0x02	I/O Enable	IOE7	IOE6	IOE5	IOE4	IOE3	IOE2	IOE1	RFU
0x03	I/O Ready	IOR7	IOR6	IOR5	IOR4	IOR3	IOR2	IOR1	RFU
0x04	Int Enable	IEN7	IEN6	IEN5	IEN4	IEN3	IEN2	IEN1	IENM
0x05	Int Pending	INT7	INT6	INT5	INT4	INT3	INT2	INT1	RFU
0x06	I/O Abort	RFU	RFU	RFU	RFU	RES	AS2	AS1	AS0
0x07	Bus Interface Control	CD Disable	SCSI	ECSI	RFU	RFU	RFU	Bus Width 1	Bus Width 0
0x08	Card Capability	4BLS	LSC	E4MI	S4MI	SBS	SRW	SMB	SDC
0x09-0x0B	Common CIS Pointer	Pointer to card's common Card Information Structure (CIS)							
0x0C	Bus Suspend	RFU	RFU	RFU	RFU	RFU	RFU	BR	BS
0x0D	Function Select	DF	RFU	RFU	RFU	FS3	FS2	FS1	FS0
0x0E	Exec Flags	EX7	EX6	EX5	EX4	EX3	EX2	EX1	EXM
0x0F	Ready Flags	RF7	RF6	RF5	RF4	RF3	RF2	RF1	RFM
0x10-0x11	Fn0 Block Size	I/O block size for Function 0							
0x12	Power Control	Reserved for Future Use (RFU)						EMPC	SMPC
0x13	High-Speed	RFU	RFU	RFU	RFU	RFU	RFU	EHS	SHS
0x14-0xEF	RFU	Reserved for Future Use (RFU)							
0xF0-0xFF	Reserved for Vendors	Area Reserved for Vendor Unique Registers							

Figure 23 CCCR Register Storage Structure

Address	7	6	5	4	3	2	1	0
0x100	Function 1 CSA enable	Function 1 supports CSA	RFU	RFU	Function 1 Standard SDIO Function interface code			
0x101	Function 1 Extended standard SDIO Function interface code							
0x102	RFU	RFU	RFU	RFU	RFU	RFU	EPS	SPS
0x103-0x108	Reserved for Future Use (RFU)							
0x109-0x10B	Pointer to Function 1 Card Information Structure (CIS)							
0x10C-0x10E	Pointer to Function 1 Code Storage Area (CSA)							
0x10F	Data access window to Function 1 Code Storage Area (CSA)							
0x110-0x111	I/O block size for Function 1							
0x112-0x1FF	Reserved for Future Use							
0x200-0x7FF	Function 2 to 7 Function Basic Information Registers (FBR)							
0x800-0xFFFF	Reserved for Future Use							

Figure 24 FBR1
Register Structure

Address	7	6	5	4	3	2	1	0
0x0001000 - 0x017FFF	Card Common Card Information Structure (CIS) area for card common and all functions							
0x018000-0x01FFFF	Reserved for Future Use							

Figure 25 CIS Storage Space Structure

11.4.2.1 SDIO CCCR Register and FBR1 Register Listings

Table 90 List of SDIO CCCR Registers and FBR1 Registers

misali gnment address	name (of a thing)	abridg e	interv iews	descriptive	reset value

0X00	CCCR/SDI O Revision	SDIOx	RO	[3:0], indicating supported CCCR/FBR formats 4'h0: CCCR/FBR Version 1.00 4'h1: CCCR/FBR Version 1.10 4'h2: CCCR/FBR Version 1.20 4'h3-4'hF: Rsv b y CIA Register[3:0].	4'h2
------	------------------------	-------	----	--	------

		CCCRx	RO	[7:4], indicates the supported SDIO protocol version 4'h0: SDIO Version 1.00 4'h1: SDIO Version 1.10 4'h2: SDIO Version 1.20 (unreleased) 4'h3: SDIO Version 2.00 4'h4-4'hF: Rsv by the CIA Register [7:4].	4'h3
0X01	SD specification Revision	SDx	RO	[3:0], indicating the supported SD protocol version 4'h0: SD Physical Version 1.01 (March 2000) 4'h1: SD Physical Version 1.10 (October 2004) 4'h2: SD Physical Version 2.00 (May 2006) 4'h3-4'hF: Rsv by the CIA Register [11:8].	4'h2
				RFU	4'h0
0X02	I/O Enable	IOEx	RO	RFU	1'b0
			RW	[7:1], Function enable, bit1-bit7 corresponds to 7 functions, the corresponding SD host enables the corresponding function if the corresponding bit is 1, otherwise the corresponding function cannot work. Note: CIS0, CIS1 and CSA are placed in Fn1, in this case, even if Fn1 is not enabled, the SD host can read and write to these three areas (CIS0 and CIS1 cannot). (Write)	7'b0

0X03	I/O Ready	IORx	RO	RFU		1'b0
------	-----------	------	----	-----	--	------

			RO	<p>[7:1],IOR has 7 bits, corresponding to the state of 7 functions, if the corresponding bit is 1, it means the function can work.</p> <p>In this design, the HC8051 configures the function ready bit of the program register to be 1, so that the bit1 of this register will be 1, thus marking that Fn1 can work normally.</p> <p>Note: Read and write operations to CIS0, CIS1, and CSA are independent of IOR1, i.e., even if the IOR1=0, the contents of these three stores can also be accessed.</p>	7'b0
0X04	Int Enable	IENM	RW	<p>[0],Interrupt enable signal</p> <p>0: Interrupts from the card cannot be sent to the SD host.</p> <p>1: Interrupts from any function can be sent to the host.</p>	1'b0
		IENx	RW	<p>[7:1],functionx interrupt enable.</p> <p>IEN1=0, the interrupt from Fn1 will not be sent to the host. IEN1=1, the interrupt from Fn1 is allowed to be sent to the host.</p>	7'b0
0X05	Int Pending		RO	[0],RFU	1'b0
		INTx	RO	<p>[7:1], interrupt pending for functionx. INT1=0, no interrupt for Fn1 is pending</p> <p>INT1=1, Fn1 has interrupt</p>	7'b0

				<p>pending.</p> <p>Note: If IEN1 and IENM are not 1, the host does not receive a pending interrupt</p>	
0X06	I/O Abort	ASx	WO	<p>[2:0], cancel the IO read or write, thus releasing the bus.</p> <p>To cancel the Fn1 operation, CMD52 Write Write 3'b1</p> <p>should be used. this command is not supported under SPI.</p>	3'b0
		RES	WO	[3],Soft reset signal	1'b0

				<p>1: Circuit to reset the SD clock domain, this bit is automatically cleared after being set and does not require a special bit.</p> <p>This reset signal does not affect the current card protocol selection (SD or SPI mode) and does not affect CD Disable.</p> <p>This reset signal does not affect the current card protocol selection (SD or SPI mode) and does not affect CD Disable.</p> <p>only CMD52 operation can be used.</p>	
		RO	RFU		4'b0
0X07	Bus Interface control	Bus Width	RW	<p>[1:0], data line width</p> <p>2'b00: 1bit Data line mode</p> <p>2'b10: 4bit Data Line Mode</p> <p>Reset or power up, it will change to 2'b00</p>	2'b00
		RO	[4:2],RFU		3'b000
	ECSI	RW		<p>[5], allowing continuous SPI interrupts.</p> <p>If SCSI is 1, this register is used to allow the SDIO card to give an interrupt at any time in SPI mode when the state of the CS line is not a concern.</p>	1'b0
	SCSI	RO		<p>[6], which supports continuous SPI interrupts.</p> <p>If it is 1, it means that the SDIO card supports SPI mode to give an interrupt at any time without caring about the state of CS.</p> <p>This register is set when program reg[2] is 1.</p>	1'b1

	CD Disable	RW	[7], connect or disconnect the 10-90K pull-up resistor on CD/DAT[3] (pin1). 0: Connect the pull-up resistor 1: Disconnect the pull-up resistor This register is cleared after power up, i.e. connecting the pull-up resistor. The state of this register is not affected by the reset command in the SD protocol.	1'b0
0X08	Card	SDC	RO [0], supports execution of CMD52 command during data transfer. Not supported in SPI mode	1'b1

Capability			This register. When program_reg[3] is 1, this register is 1	
	SMB	RO	[1], indicates that the SDIO card supports the Block transfer mode required by CMD53. When program_reg[4] is 1, this register is 1	1'b1
	SRW	RO	[2], indicates that the SDIO supports Read Wait-Read Wait Control (RWC) operation. When program_reg[5] is 1, this register is 1	1'b1
	SEBS	RO	[3] ,indicates that the SDIO card supports suspend/resume. If 0, the (0x0C-0x0F) registers are not supported If 1, all functions except Fn0 will hang or resume as required by the SD host. When program_reg[6] is 1, this register is 1	1'b1
	S4MI	RO	[4], indicates that the SDIO card supports generating interrupts to the host in 4bit Multi-Block data transfer mode. 0: interrupt generation between Block transfers is not supported, in this case, SDIO can still initiate interrupts to the host during other interrupt cycles as long as IENx=1 1 1: interrupt generation between Block transfers is supported When program_reg[7] is 1, this register is 1	1'b1

	E4MI	RW	[5], interrupt enable. Allows interrupts to be generated to the host in the middle of a two-block data transfer in 4bit multi-block mode. 0: Not allowed 1: Permission	1'b0
--	------	----	--	------

				A power-on reset or reset command clears this register.	
	LSC	RO	[6]. 0: Indicates that the SDIO card is a full speed device 1: Indicates that the SDIO card is a low-speed device When program_reg[8] is 1, this register is 1	1'b0	
	4BLS	RO	[7]. 0: Indicates that the SDIO is a low-speed mode device or does not support 4bit mode. 1: Indicates that SDIO is a low-speed mode device and supports 4bit mode When program_reg[9] is 1, this register is 1	1'b1	
0X09- 0X0B	Common CIS pointer	RO	[23:0], Pointer to the start address of the SDIO card common CIS (CIS0). CIS0 Package Contains information about the entire card. Its access space is: 0x001000-0x017FFF. Pointers are stored in small end format (LSB).	24'h00 1000	

0X0C	Bus Suspend	BS	RO	<p>[0] ,Bus status.</p> <p>0: The currently selected function does not use the data bus.</p> <p>1: The currently selected function (using FSx or the function number in the IO command) is executing the command that will transfer data on the data line.</p> <p>This register is used by the host to determine which function is currently using the data bus.</p> <p>If the SDIO card does not support the hang recovery function, this register is 0.</p> <p>Any operation that accesses the CIA cannot hang; this register is always 1, even if the BR register is 1.</p>	1'b0
------	-------------	----	----	--	------

				SPI mode, read-only and 0.	
	BR	RW		<p>Bus Release Request/Status. This register is used to request the selected function (selected with FSx or CMD53 total function number) to release the data bus and suspend the associated operation.</p> <p>If the host sets this register to 1, the selected function temporarily stops data transfer on the data line and hangs the command for the current data operation. The BR register remains at 1 until the release process is complete.</p> <p>Once a function is hung, the device notifies the host by clearing BS, BR. The host can monitor the execution status of the pending request by reading BR. If BR is 1, the pending request is still executing. The host can cancel a pending request by actively writing 0 to BR.</p> <p>SPI mode, read-only and 0.</p>	4'h0
		RO		[7:2],RFU	6'b0

0X0D	Function Select	FSx	RW	<p>[3:0], which are used to pick function[0-7] during suspend/resume operations. Two ways to write FSx:</p> <p>Performs an IO write operation to the CCCR</p> <p>Newly initiated IO commands will cause the FSx to be set to the function number in the command.</p> <p>If function is currently hung, writing the number of that function to FSx will resume the data transfer operation for that function when reading FSx. The returned value will be the number of the currently selected function.</p> <p>Note: If BS=0 when reading FSx, the value of FSx is undefined.</p>	4'b0
------	-----------------	-----	----	---	------

			4'b0000: Transaction of function 0 (CIA) 4'b0001-4'b0111: Transaction to functions 1-7 4'b1000: Transaction of memory in combo card 4'b1001-4'b1111: Not defined, reserved for future use	
	RO	[3:1],RFU		3'b000
DF	RO	<p>[7] to restore the data flag. Writing function number to FSx will resume data transfer for the selected function. Once the data transfer is resumed, the DF register will flag if there is more data to transfer.</p> <p>0: There is no more data to be transferred after the function has been restored. 1: There is more data to be transferred after the function has been restored.</p> <p>DF is used to control the interrupt cycle in 4bit mode. If 1, function resumes after there is more data to transfer, in which case the interrupt cycle is canceled. If 0, function resumes after the end of the data transfer (in the BUSY case), in which case, after resumption, there is no data to be transferred so the host can function. The start of the interrupt cycle is monitored after recovery.</p>	1'b0	

0X0E	Exec Flags	EXx	RO	[7:0], execution flag bits. The host uses these bits to determine the status of all function [7- 1] execution commands. These registers inform the host that a function is executing a command and therefore no new commands can be issued to that function. SPI mode, read-only and 0.	8'h00
0X0F	Ready Flags	RFx	RO	[7:0], read flag bits. The host can use these registers to learn the value of the function [7- 1] Read-write busy state. If a function is executing a write transaction, the corresponding	8'h00

				The RFx bit is cleared to indicate that the function is busy and not ready to receive more numbers. Data. If a function is performing a read operation, the corresponding RFx bit is cleared to zero to indicate that the read data is invalid, or 1 to indicate that the read data can be transmitted. SPI is invalid, read-only, and 0.	
0X10- 0X11	FN0 Block Size	RW	[15:0],Fn0 Block size size for corresponding Block transfer. Maximum 2048 Byte, minimum 1 Byte. stored in small segment format (LSB)	16'h00	
0X12	Power Control	SMPC	RO	<p>[0], support host power control.</p> <p>0: Total SDIO current less than 200mA, even if all functions are active</p> <p>(IOEx=1).EMPC, SPS, EPS are all 0.</p> <p>1: Total SDIO current can exceed 200mA. valid for EMPC, SPS, EPS.</p>	1'b1
		EMPC	RO	<p>[1],Host power control enable.</p> <p>0: The total current of the SDIO card is less than 200mA. the SDIO card automatically switches function(s) to low current mode or disallows some function enables while it ignores the value of EPS so that the current of the card is less than or equal to 200mA.</p> <p>1: The total current of the SDIO card can exceed 200mA with SPS and EPS active. The host uses the SPS, EPS, and IOEx in the FBR to enable higher current functions based on its ability to provide current.</p>	1'b0

			RO	[7:2],RFU	
0X13	High-Speed	SHS	RO	[0], indicates that the SDIO card supports high speed 0: does not support high speed 1: Support high speed	1'b1

		EHS	RW	[1], high speed enable 0: SDIO card working at default speed, maximum frequency 25MHZ 1: The SDIO card can work in high speed mode up to 50MHZ.	1'b0
			RO	[7-2], RFUs	
0X14- 0XEF	RFU	RO		Reserved for Future Use (RFU)	8'b0
0XF0- 0xFF	Reserved for Vendors	RO		Area Reserved for Vendor Unique Registers	8'b0
0X100	I/O Device Interface Code	RO		<p>[3:0], flags what kind of device Fn1 is. Programmable through registers CIA[15:12].</p> <p>4'h0 No SDIO standard interface supported by this function</p> <p>4'h1 This function supports the SDIO Standard UART.</p> <p>4'h2 This function supports the SDIO Type-A for Bluetooth standard interface</p> <p>4'h3 This function supports the SDIO Type-B for Bluetooth standard interface</p> <p>4'h4 This function supports the SDIO GPS standard interface</p> <p>4'h5 This function supports the SDIO Camera standard interface</p> <p>4'h6 This function supports the SDIO PHS standard interface</p>	4'h7

			4'h7 This function supports the SDIO WLAN interface 4'h8 This function supports the Embedded SDIO-ATA standard interface	
	RFU	RO	[5:4],RFU	2'b00
	Function supports CSA	RO	[6]. 0: Fn1 does not support CSA 1: Fn1 Supported with CSA It can be programmed via register CIA [16]	1'b0
	Function CSA enable	RW	[7]. 0: Access not allowed CSA 1: Allow access to CSA	1'b0
0X101	Extended standard I/O device type code	RO	[7:0], the extension of the I/O Device Interface Code is programmable via registers CIA [24:17].	8'b0
0X102	SPS	RO	[0], marking whether Fn1 has a power selection 0: no power selection 1: There are two power consumption options, which can be selected by EPS Programmable via register CIA[25].	1'b0

EPS	RW	[1], Power consumption selection 0: Fn1 operating in high current mode 1: Fn1 operating in low current mode	1'b0
-----	----	---	------

		RO	[7:2], RFU	6'b0
0X103- 0X108		RO	RFU	0
0X109- 0X10B	Address pointer to function CIS1	RO	[16:0], Pointer to Fn1's CIS address, CIS1, instructs the host to access Fn1's CIS Starting address. Stored in LSB segment format.	17'h02 000
		RO	[23:17], RFU	7'b0
0X10C - 0X10E	Address pointer to function CSA	RW	[23:0], a 24bit address pointer to the CSA, which is automatically incremented by 1 when the host accesses the CSA through the CSA Access Window. Addresses are stored in small segment format (LSB)	24'h00 0000
0X10F	Data access window to CSA	RW	[7:0], the window for reading and writing to the CSA. For write operation to this address, the corresponding data is written to the address indicated by the CSA 24bit address pointer through this window, and for read operation to this address, the data is read from the address indicated by the 24bit CSA address pointer through the This window is sent to the host.	8'b00
0X110- 0X111	Function1 IO Block Size	RW	[15:0], 16bit registers, used to set the IO block size, the maximum block size is 2048Byte, the minimum is 1. This data is stored in small end mode (LSB).	16'b0

0X100 0- 0X101	CIS0	RO	The host accesses the CIS address space of Fn0, i.e., the host accesses CIS0 through this address space segment. This SDIO card supports up to 17 bytes of CIS0.	
0X200 0-	CIS1	RO	The host accesses the CIS address space of Fn1, i.e., the host accesses this address space segment through the This SDIO card supports CIS1 byte data from 55 to 308.	

0X213				
3				
RFU			RFU	

11.4.3 SDIO Fn1 register

Fn1 register is the address space allocated to function1 by SDIO protocol, and its address range is 0x00000~0x1FFF, totaling 128K. Since the address bit width of AHB bus inside the chip is 32 bits, SDIO can't use the 17-bit address to access the inside of the chip directly, so it is necessary to complete the address mapping once in the design. The specific mapping relationship is shown in the following table(FN1 access space)

Table 91 SDIO Fn1 Address Mapping Relationships

SDIO Host Access Window	Corresponds to the actual physical address space	Actual physical address space contents
0X0000~ 0X00FF	0X0000~ 0X00FF	SDIO module internal register address space.
0X1000~ 0X1FFF	configurable	CIS0 Physical space, the exact physical space is configured by the firmware.
0X2000~ 0X2FFF	configurable	CIS1 Physical space, the exact physical space is configured by the firmware.
0X4000~ 0X4FFF	configurable	Downstream and upstream cmd physical space, specific physical space The address is configured by the firmware.
0X5000~ 0X5FFF 0X15000~ 0X15FFF	changeable	Sends the buffer address space, depending on the sdio_txbd The instructions in the

0X6000~ 0X7FFF	changeable	Receives the buffer address space as per sdio_rxbd
0X16000~ 0X17FFF		The instructions in the
0X8000~ 0X9FFF	0X0E000000~ 0X0E002000	AHB bus config address space.
0XA000~ 0XBFFF	0X0F000000~ 0X0F002000	AHB Bus APB Address Space.

The driver should avoid accessing spaces beyond the above, which may have unintended consequences.

The first of these address space registers is internal to the SDIO and can only be accessed by the SDIO HOST; access to the other address spaces will be mapped to other spaces within the chip according to the description.

This section only describes the registers in the SDIO 0x0000~ 0x00FF address space, which are directly accessed by the SDIO host via the CMD52 command, and the offset address is the access address, with function number 1.

Table 92 SDIO Fn1 Partial Registers (for HOST access)

offset address	name (of a thing)	bit width	intervi ews	descriptive	reset value
0X00~0X03			RO	RSV	
0X04	int_read_data	[7:1]	RO	RSV	7'b0
		[0]	RW	Uplink data interrupt. Active high, write 1 to clear. It is also automatically cleared when 0x1C is read.	1'b0
0X05	int_mask	[7:1]	RO	RSV	7'b0
		[0]	RW	Mask enable signal corresponding to int_src. 1 masks the corresponding interrupt.	1'd0
0X06	wlan_awake_stts	[7:2]	RO	RSV	6'b0
		[0]	RO	Current WLAN status: 1 is ACTIVE; 0 is SLEEP.	1'b1

0X1C	dat_len0	[6:0]	RO	Uplink data length high 7bit. total 12bit, low 5bit in 0x1D Center.	7'b0
	dat_vld	[7]	RO	1'b1	1'b1

0X1D	dat_len1	[7:3]	RO	Uplink data length low 5bit. total 12bit, high 7bit in 0x1C Center.	5'b0
		[2:0]	RO	RSV	3'b0
0X1E			RO	RSV	
0X1F		[7:2]	RO	RSV	6'b0
	down_cmdbuf_vld	[1]	RO	The downstream command buffer is available, 1 is valid.	1'b1
	txbuf_vld	[0]	RO	Downlink data buffer available. 1 Valid, indicates that there are available transmitting buffer.	1'b0
0X20	wlan_wake_en	[0]	RW	Chip wake-up enable issued by SDIO in SLEEP state, active high. This bit is automatically cleared by hardware when the chip is woken up.	1'b0
0X21		[7:1]	RO	RSV	7'b0
	fn1_RST	[0]	RW	Soft reset, 1 active. After the software writes 1, (i.e., the chip wlan part of the circuit) is reset, and after writing 0, the function1 reset is released.	1'b0
0X22		[7:1]	RO	RSV	7'b0

	fn1_recov	[0]	RW	<p>Error recovery enable, 1 is valid, this bit is automatically cleared to 0 after the end of the command RESPONSE.</p> <p>This function is used to fulfill the same function of fn0/fn1 io abort. When there is a cmd exception or the command is terminated prematurely, this register can be set to 1 to fulfill the io abort operation.</p> <p>The io abort command is limited in some versions of the bus driver.</p>	1'b0
--	-----------	-----	----	--	------

				(i.e., this register has restricted access to the address space) and does not allow the user driver to make the In this case, writing a 1 to this register replaces the io abort operation and produces the same effect.	
--	--	--	--	--	--

11.4.3.1 SDIO AHB Interface Slave Device Registers

The following registers are used when the SDIO slave device is initialized.

When transferring data, it is necessary to work with a wrapper controller, refer to the HSPI section of the documentation for the Wrapper Controller section.

Table 93 SDIO AHB Bus Registers

offset address	name (of a thing)	bit width	interviews	descriptive	reset value
0X0000			RO	Rsv	
0X0004					
0X0008	CIS function0 address	[31:0]	RW	CIS0 Offset address stored in the system's internal memory. CIS0 actual storage start address = 0x01000 (read command start address) + this offset address	32'b0
0X000C	CIS function1 address	[31:0]	RW	The offset address where CIS1 is stored in the system's internal memory. CIS1 actual storage start address = 0x02000 (start of read	32'b0

				command) address) + this offset address	
0X0010	CSA address	[31:0]	RW	Firmware initialization sets the cheap address for accessing the CSA in the same way as the CIS settings are the same. CSA is not supported in this design.	32'b0
0X0014	Read address	[31:0]	RW	Used to set the starting address for the DMA to read data from memory. Match	32'b0

				The combination of the Data Port registers enables the use of the internal memory of the system. Read operation (i.e., access address 0x00+ Read address). In this design, this method is not used, so the default is 0	
0X0018	Write address	[31:0]	RW	It is used to set the start address for DMA to write data to memory. Together with the Data Port register, it can realize the write operation to the internal memory (i.e., the access address is 0x00+ write address). In this setup, the This method is not used in the calculations, so the default is 0.	32'b0
0X001C	AHB Transfer count	[20:0]	RW	The SDIO device informs the host of how many bytes of data need to be read in the read data operation that will be initiated. [23:21]RFU	32'b0
0X001F		RO	--	rsv	
0X0020	SDIO Transfer count	[20:0]	RO	The number of bytes sent down from the host to the SDIO device during a data transfer. The number of bytes sent from the host to the SDIO device in a single data transfer. [31:21]RFU	32'b0

0X0024	CIA register	--	RW	[3:0]:CCCR Revision. default 4'h2 4'h0 CCCR/FBR Version 1.00 4'h1 CCCR/FBR Version 1.10 4'h2 CCCR/FBR Version 1.20 [7:4] SDIO Revision. default is 4'h3 4'h0 SDIO Specification 1.00	32'h06017 232
--------	--------------	----	----	--	----------------------

			<p>4'h1 SDIO Version 1.10</p> <p>4'h2 SDIO Version 1.20</p> <p>4'h3 SDIO Version 2.0</p> <p>[11:8] SD Revision. default is 4'h2</p> <p>4'h0 SD Physical Specification 1.01</p> <p>4'h1 SD Physical -Spec-1.10</p> <p>4'h2 SD Physical Spec 2.0</p> <p>[15:12]IO-Device Code. the default is 4'h7. that is. this product is a WLAN device.</p> <p>[16] csa_support, default is 1. 0: CSA not supported 1: Support for CSA</p> <p>For initialization, the firmware needs to configure this register to 0.</p> <p>[24:17],Extended IO-device code The default is 8'b0.</p> <p>Extension of standard IO-device code for Fn1. [25],SPS, default is 1, i.e. Fn1 supports high power consumption 0: not supported</p>	
--	--	--	--	--

				<p>1: Support</p> <p>[26],SHS, default is 1, high speed supported 0: not supported</p> <p>1: Support</p> <p>[31:27]:RFU</p>	
0X0028	Program Register	--	RW	<p>[0],function ready. hc8051 sets the Fn1 register to indicate to the SD host that Fn1 is ready for operation after completing the Fn1 register assignment required for SD initialization. The default is 0.</p> <p>0:Fn1 not ready</p> <p>1:Fn1 ready</p> <p>[1]This register is set when Fn1 is ready to send data to the SD host. When the host responds to a read interrupt and reads the Interrupt Identification register, this bit is automatically cleared to zero. The default is 0.</p> <p>0: No data sent to host</p> <p>1: There is data sent to the host.</p> <p>[2]SCSI. continuous SPI interrupts are supported. Default is 1. 0: Not supported</p> <p>1: Support</p> <p>[3]SDC. indicates that the SDIO card supports</p>	16'h 02fc

				the execution of CMD52 commands while data is being transferred. Default is 1 0: Not supported	
--	--	--	--	--	--

				<p>1: Support</p> <p>[4] The SDIO card supports Block transfer of CMD53. The default is 1.</p> <p>0: Not supported</p> <p>1: Support</p> <p>[5] SRW: Indicates that the SDIO card supports read wait. Default is 1. 0: Not supported</p> <p>1: Support</p> <p>[6] SBS: Indicates that the SDIO card supports suspend/resume. Default is 1 0: Not supported</p> <p>1: Support</p> <p>[7] S4MI. indicates that the SDIO card supports generating interrupts during 4bit multi-Block data transfers. Default is 1.</p> <p>0: Not supported</p> <p>1: Support</p> <p>[8] LSC. indicates that the SDIO card is a low-speed device. Default is 0. 0: Full speed device</p> <p>1: Low-speed equipment</p> <p>[9] The SDIO card is a low-speed device, but supports 4bit data transfer. The default is 1.</p>	
--	--	--	--	---	--

				0: Not supported 1: Support	
--	--	--	--	--------------------------------	--

			[10],card ready. signal belonging to the SD clock domain, after the power-on reset is released, this register is automatically changed to 1 to indicate that the SDIO (interface section) is ready. The firmware monitors this signal, so that it can configure the Fn1 register needed for initialization. 0 at power-on reset. [15:11],RFU. default is 0	
--	--	--	--	--

0X0034	OCR register	--	RW	[23:0], operating condition register, internally programmable, mainly used to match the host operating voltage range. Default is: 24'hff8000.	32'h00ff8000
				Register Bit	
				Supported Voltage	
				Range 0-3 Reserved	
			4	Reserved	
			5	Reserved	
			6	Reserved	
			7	Reserved	
			8	2.0-2.1	
			9	2.1-2.2	
			10	2.2-2.3	
			11	2.3-2.4	
			12	2.4-2.5	
			13	2.5-2.6	
			14	2.6-2.7	
			15	2.7-2.8	

				16 2.8-2.9 17 2.9-3.0 18 3.0-3.1 19 3.1-3.2 20 3.2-3.3 21 3.3-3.4 22 3.4-3.5 23 3.5-3.6 [31:24], 8'b0	
0X0038		RW	--	rsv	32'h0
0X003C	CD_State Register	--	RW	[0], marking the status of the SD dat[3] data line pull-up resistor. 0: Pull-up active 1: Invalid pull-up This register can be accessed by any of the on-chip AHB buses. Note: The result of an AHB write operation to this register indirectly modifies the value of CCCR7[7] CD. [31:1],RFU	32'b0

0X0040	Fn1_Ena Register	--	RW	[0], mark whether Fn1 is enabled or not. 0: Disable 1: Enabling	32'b0
--------	---------------------	----	----	---	-------

				This register can be accessed by any of the on-chip AHB buses. Note: The result of the AHB write operation to this register indirectly modifies the value of CCCR1[1] IOE1. [31:1],RFU	
--	--	--	--	--	--

12 HSPI/SDIO Wrapper Controller

12.1 Functional overview

With the interface controllers (SDIO and HSPI), it accomplishes the DMA operations between the host and the internal caches of the chip. Including upstream and downstream data cache hardware and software interaction control, send and receive cache fill and release, upstream data interrupt generation and so on.

Both SDIO and HSPI interact with the host computer through the wrapper controller, and the control commands and processes are the same. For convenience, some registers are prefixed with SDIO, and the corresponding register operations are also applicable to HSPI.

Note that for registers prefixed with SDIO_TX, the control is for the device to receive data, and for SDIO_RX, the control is for the device to send data.

The presence of the cmd field in the register is only a descriptive distinction from the data frame, and does not mean that the command channel can only be used to transmit commands, but can also be used to transmit data. The difference between command and data here is that the command channel has only one cache and the cache length is generally less than 256 bytes, while the data channel has multiple caches, each of which is more than 1K in length, and the multiple caches form a chained-list structure, with the specific length configured by the software. Due to this chain-list cache structure of data frames, the transmission rate will be faster than that of the command channel.

12.2 Main characteristics

-
- Support for word-aligned data moves
 - Supports DMA function
 - Support for chained table structure management
 - Support for interrupt generation
 - Receives up to 4096 bytes of data

12.3 Functional Description

12.3.1 uplink data reception function

The uplink direction is the direction in which the master device (SDIO or HSPI) sends data to the slave device (W800).

When the master device sends data to the slave device, the SDIO or HSPI module receives the data, links the data to the receiving BD via WRAPPER, and generates an interrupt to notify the application software to process the data.

Receive BD descriptor:

sdio_rxbd			0	
31	Vld[31]	RSV		word0
	Sdio_rx_info_size[31:26]	Frm_len[25:12]	Rxbufo_offset[11:0]	word1
		Sdio_rxbuf0_addr[31:0]		Note: The longest frame receive d is 4096 and occupi es up to three slices. rxbu fo_of fset is only valid for the first slice in which the frame is stored (i.e., the slice in sdio_rx buf0), and data is stored sequen tially from the base
		Sdio_rxbuf1_addr[31:0]		
		Sdio_rxbuf2_addr[31:0]		
		Next_sdio_rxbd_addr[31:0]		

Description:

1. vld, 1 is valid, indicating that the current descriptor points to a valid receive frame.
2. rxbufo_offset: byte address, word aligned, indicates the offset address of the word where the first valid byte of the 802.3 frame is located relative to sdio_rxbuf0.
3. sdio_rxbuf0_addr: byte address, word-aligned, buf of first slice of uplink data frame Base address.
4. sdio_rxbuf1_addr: byte address, word-aligned, buf of the second slice of the uplink data frame Base address.
5. sdio_rxbuf2_addr: byte address, word-aligned, buf of the third slice of the uplink data frame Base address.
6. next_sdio_rxbd_addr, byte address, word-aligned, base address of the next sdio_rxbd.
7. frm_len, byte length, indicates the length of the uplink data, excluding sdio_rx_info.
8. sdio_rx_info_size, byte length, indicates the number of sdio_rx_info bytes, must be an integer multiple of 4 bytes.

word2

word3

word4

word5

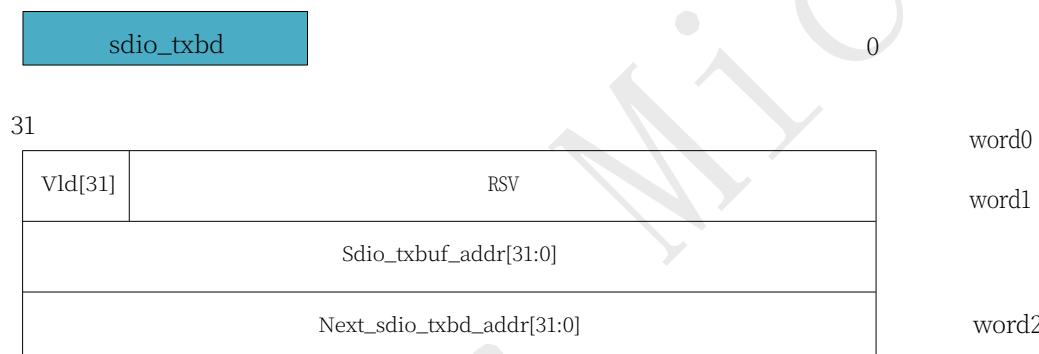
Figure 26 SDIO Receive BD Descriptors

When the W800's SDIO module or HSPI module detects that the receive enable is active, it reads RXBD and determines the flag of Vld.

12.3.2 Downlink Data Migration Function

When W800 has data to send to the master device, the software first prepares the send description, then, notifies WRAPPER to move the data. WRAPPER notifies the master device to read the device to be sent through the interrupt signals of SDIO or HSPI, and WRAPPER generates the send completion interrupt to notify the program when the data sending is completed.

Send BD Descriptor:



Description:

1. vld, 1 is valid, indicating that the current descriptor points to an available transmit cache.
2. Sdio_txbuf_addr, byte address, must be word-aligned. Indicates that this descriptor points to the base address where the transmit data is stored. This address is offset relative to the base address of each sdio_txbuf to ensure that the base address of the sdio_txbuf is not exceeded when the firmware adds LLC upwards at the beginning of the frame.
3. next_sdio_txbd_addr, byte address, word-aligned. Base address of the next sdio_txbd.

Figure 27 SDIO Transmit BD Descriptor

12.4 register description

12.4.1 register list

Table 94 WRAPPER Controller Registers

misali gnme nt	name (of a thing)	abridge	inter views	descriptive	reset value
----------------------	-------------------	---------	----------------	-------------	----------------

address					
0X0000	WRAPPER Interrupt Status Register	INT_STTS	RW	Command or data frame interrupt status	0X0000

0X0004	WRAPPER Interrupt Configuration Register	INT_MASK	RW	Whether command or data frame interrupts are masked	0X0000
0X0008	WRAPPER Uplink Command Ready Mail depositor (computing)	UP_CMD_AVAIL	RW	Uplink Command Ready	0X0000
0X000c	WRAPPER The next command, buf, is a command for register	DOWN_CMD_BUF_A VAIL	RW	The downstream command buf is ready.	0X0000
0X0010	SDIO_TX Link Enable Register	SDIO_TX_BD_LINK_E N	RW	Indicates whether the sdio_txbd linked table descriptor is efficiently	0X0001
0X0014	SDIO_TX Link Address Register	SDIO_TX_BD_ADDR	RW	The address pointed to by the current sdio_txbd, which needs to be To word align, initialization requires configuring the	0X0000
0X0018	SDIO_TX Enable Register	SDIO_TX_EN	RW	SDIO Send Frame Enable	0X0000
0X001c	SDIO_TX Status Register	SDIO_TX_STTS	RO	SDIO Transmit Status	0X0000
0X0020	SDIO_RX Link Enable Register	SDIO_RX_BD_LINK_E N	RW	Indicates whether the sdio_rxbd linked table descriptor is efficiently	0X0001
0X0024	SDIO_RX Link Address Register	SDIO_RX_BD_ADDR	RW	The current address pointed to by sdio_rxbd needs to be To word align, initialization requires configuring the	0X0000
0X0028	SDIO_RX Enable Register	SDIO_RX_EN	RW	SDIO receive frame enable	0X0000
0X002c	SDIO_RX Status Register	SDIO_RX_STTS	RO	SDIO Receive Status	0X0000

0X0030	WRAPPER CMD BUF Base address processor register	CMD_BUF_BASE_AD DR	RW	Downstream cmd buf base address	0X0000
0X0034	WRAPPER CMD BUF SIZE processor register	CMD_BUF_SIZE	RW	Cmd buf byte size	0X0064

12.4.2 WRAPPER Interrupt Status Register

Table 95 WRAPPER Interrupt Status Registers

classifier for honorific people	intervie ws	Operating Instructions	reset value
[31: 4]	RO	reservations	
[3]	RW	int_down_cmd down_cmd Frame completion interrupt. Write 1 to clear 0.	1'b0
[2]	RW	int_up_cmd Upstream cmd frame word completion interrupt. Write 1 Clear 0.	1'b0
[1]	RW	int_sdio_txfrm Downstream data frame completion interrupt. Write 1 Clear 0.	1'b0
[0]	RW	int_sdio_rxfrm Uplink data frame completion interrupt. Write 1 Clear 0.	1'b0

12.4.3 WRAPPER Interrupt Configuration Register

Table 96 WRAPPER Interrupt Configuration Registers

classifier for honorific people	intervie ws	Operating Instructions	reset value
[31: 4]	RO	reservations	
[3]	RW	int_mask_down_cmd down_cmd Frame Completion Interrupt Mask Register. 1 is masked, same below.	1'b0
[2]	RW	int_mask_up_cmd Upstream cmd frame completion interrupt mask register.	1'b0
[1]	RW	int_mask_sdio_txfrm Downstream data frame completion interrupt mask register.	1'b0
[0]	RW	int_mask_sdio_rxfrm Uplink data frame completion interrupt mask register.	1'b0

12.4.4 WRAPPER Uplink Command Ready Register

Table 97 WRAPPER Uplink Command Ready Registers

classifier for honorific people		intervie ws	Operating Instructions	reset value

[31: 1]		RO	reservations	
[0]		RW	<p>When an uplink cmd frame exists, the firmware sets this bit to 1.</p> <p>When the uplink cmd transmission is complete, the firmware sets this bit to 1.</p> <p>The device automatically clears it to 0 and generates the int_up_cmd interrupt.</p>	1'b0

12.4.5 WRAPPER down command buf ready register

Table 98 WRAPPER Down Command buf Ready Registers

classifier for honorific people	intervie ws	Operating Instructions	reset value
[31: 1]	RO	reservations	
[0]	RW	<p>After sending the downstream cmd, the hardware clears this bit to 0 and generates an interrupt. When the firmware finishes processing the downlink command, the hardware clears this bit and generates an interrupt.</p> <p>After the order, place this position 1.</p>	1'b0

12.4.6 SDIO TX Link Enable Register

Table 99 SDIO TX Link Enable Registers

classifier for honorific people	intervie ws	Operating Instructions	reset value
[31: 1]	RO	reservations	

[0]	RW	<p>sdio_txbd Link enable, active high.</p> <p>If this bit is valid, the hardware finishes processing an sdio_txbd descriptor and proceeds directly to the next descriptor pointed to by next_sdio_txbd_addr. If this bit is not valid, the hardware stops immediately after processing an sdio_txbd.</p> <p>The same applies to HSPI.</p>	1'b1
-----	----	---	------

12.4.7 SDIO TX Link Address Register

Table 100 SDIO TX Link Address Registers

classifier for honorific people	intervie ws	Operating Instructions	reset value
[31:0]	RW	<p>Current sdio_txbd byte address, software needs to strictly ensure word alignment, below.</p> <p>Initially the firmware needs to configure this register, and the hardware updates next_sdio_txbd_addr in the sdio_txbd descriptor to this register each time it finishes processing a send buf.</p> <p>The same applies to HSPI.</p>	32'h0

12.4.8 SDIO TX Enable Register

Table 101 SDIO TX Enable Register

classifier for honorific people	intervie ws	Operating Instructions	reset value
[31: 1]	RO	reservations	
[0]	RW	<p>SDIO Send Frame Enable, active high.</p> <p>The firmware notifies the SDIO module of the existence of a new transmit descriptor by setting this bit to 1 after it finishes forming each descriptor sdio_txbd descriptor. the SDIO module detects that this bit is valid and initiates a read of the current sdio_txbd and completes the transmit frame process.</p> <p>The hardware automatically clears this register.</p>	1'b0

		The same applies to HSPI.	
--	--	---------------------------	--

12.4.9 SDIO TX Status Register

Table 102 SDIO TX Status Registers

classifier for honorific people	intervie ws	Operating Instructions	reset value

[31: 1]	RO	reservations	
[0]	RW	<p>SDIO Transmit Status.</p> <p>0: SDIO has stopped the transmit process because there are no transmit descriptors available</p> <p>1: SDIO is in the process of transmitting.</p>	1'b0

12.4.10 SDIO RX Link Enable Register

Table 103 SDIO RX Link Enable Registers

classifier for honorific people	intervie ws	Operating Instructions	reset value
[31: 1]	RO	reservations	
[0]	RW	<p>sdio_rxbd Link enable, active high.</p> <p>If this bit is valid, the hardware finishes processing an sdio_rxbd descriptor and proceeds directly to the next descriptor pointed to by next_sdio_rxbd_addr. If this bit is not valid, the hardware stops immediately after processing an sdio_rxbd.</p> <p>The same applies to HSPI.</p>	1'b1

12.4.11 SDIO RX Link Address Registers

Table 104 SDIO RX Link Address Registers

classifier for honorific people	intervie ws	Operating Instructions	reset value

[31:0]	RW	<p>Current sdio_rxbd byte address.</p> <p>Initially the firmware needs to configure this register, and the hardware updates next_sdio_rxbd_addr in the sdio_rxbd descriptor to this register each time it finishes processing a send buf.</p>	32'h0
--------	----	---	-------

		The same applies to HSPI.	
--	--	---------------------------	--

12.4.12 SDIO RX Enable Register

Table 105 SDIO RX Enable Registers

classifier for honorific people	intervie ws	Operating Instructions	reset value
[31: 1]	RO	reservations	
[0]	RW	<p>SDIO Receive Frame Enable, active high.</p> <p>The firmware notifies the SDIO module of the existence of a new receive descriptor by setting this bit to 1 after it finishes forming each descriptor sdio_rxbd descriptor. the SDIO module detects that this bit is valid and initiates a read of the current sdio_txbd and completes the receive frame process.</p> <p>The hardware automatically clears this register.</p> <p>The same applies to HSPI.</p>	1'b0

12.4.13 SDIO RX Status Register

Table 106 status register
SDIO RX

classifier for honorific people	intervie ws	Operating Instructions	reset value
[31: 1]	RO	reservations	

[0]	RW	<p>SDIO receive status.</p> <p>0: SDIO has stopped the receive process due to the absence of a valid uplink descriptor and uplink command</p> <p>1: SDIO is in the process of receiving also applies to HSPI.</p>	1'b0
-----	----	---	------

12.4.14 WRAPPER CMD BUF Base Address Register

Table 107 WRAPPER CMD BUF Base Address Registers

classifier for honorific people	intervie ws	Operating Instructions	reset value
[31:0]	RW	<p>Downstream cmd buf base address.</p> <p>The uplink cmd buf base address is this base address plus cmd_buf_size.</p>	32'h0

12.4.15 WRAPPER CMD BUF SIZE Register

Table 108 WRAPPER CMD BUF SIZE Registers

classifier for honorific people	intervie ws	Operating Instructions	reset value
[31:12]	RO	reservations	
[11:0]	RW	cmd buf byte size, must be an integer multiple of 4 bytes.	12'd64

13 SDIO HOST Device

Controller

13.1 Functional overview

The SDIO HOST Device Controller provides a digital interface capable of accessing Secure Digital Input/Output (SDIO) cards as well as MMC cards. Access to SDIO devices and SD card devices compatible with the SDIO 2.0 protocol is possible. The main interfaces are CK, CMD and 4 data lines.

13.2 Main characteristics

- Compatible with SD card specification 1.0/1.1/2.0 (SDHC)
- Compatible with SDIO Memory Card Specification 1.1.0
- Compatible with MMC specification 2.0~4.2
- Configurable interface clock rate, supports host rate 0~50MHz.
- Supports standard MMC interface
- Supports blocks up to 1024 bytes
- Supports soft reset function
- Automatic Command/Response CRC generation/verification;
- Automatic data CRC generation/checksum;
- Configurable timeout detection;
- Supports SPI, 1-bit SD and 4-bit SD modes
- Supports DMA data transfer

13.3 Functional Description

13.4 register description

13.4.1 register list

sites	Register Name	name (of a thing)	bit width	causality	clarification	reset value
0x00	mmc_ctrl	RSV	[15:11]	RO		
			[10]	RW	SDIO read wait enable '1' : enable SDIO read wait '0' : disable SDIO read wait	1'b0
			[9]	RW	SDIO Interrupt Enable '1' : SDIO interrupt enable '0' : SDIO interrupt disable	1'b0
			[8]	RW	SDIO mode enable '1' : SDIO '0' : SD/MMC	1'b0
			[7]	RW	SD/MMC/SDIO interface data width '1' : 4 bits '0' : 1 bit	1'b0

	[6]	RW	SD/MMC/SDIO transfer mode '1' : High-Speed Mode '0' : Low-Speed Mode	1'b1
	[5:3]	RW	SDIO/MMC/SDIO port clock rate selection Refer to Table 2	3'b000

		[2]	RW	SDIO/MMC/SDIO interface drive mode selection '1': open_DrainMode '0' :Push-Pull Mode	1'b1
		[1]	RW	Signal Mode Selection '1' : Automatic selection of transmission mode '0' :use mmc_port register to select	1'b0
		[0]	RW	Port Mode Selection '1' : MMC mode '0' :SPI mode	1'b1
0x04	mmc_io	RSV	RO	[15:10]	6'd0
			RW	[9]	SDIO cmd12/IO Abort Flag '1' : Flag the current command as cmd12/IO Abort '0' : Flag the current command as not cmd12/IO Abort
			RW	[8]	SDIO Command Attributes '1' : Marks the current command followed by a data block; '0' : Marks the current command followed by no data block and no command response;

		RW	[7]	Enable auto generate 8 null clock after response/command or single block data	1'b0
--	--	----	-----	---	------

				Automatic generation of 8 null clock functions after a response/command or a single block of data '1': enable '0': Close	
	RW	[6]		Enable auto receive response after command Automatically receive response after command function '1': Enable '0': Close	1'b0
	RW	[5]		8 null clock generation on SD/MMC/SDIO port clock lines '1': generate 8 null clocks '0': Set according to bit 3 Receive response/transmit command	1'b0
	RW	[4]		Designed for CID and CSD reading. When the CID or CSD command is sent, the SD/MMC/SDIO card device will reply with 136bit CID or 11 CSD data on the CMD line. When this bit is set to 1, the CID or CSD data will be stored in the [135:8] command buffer area.	1'b0

		RW	[3]	Response/command selection when bit[5] is '0' '1': Receive response	1'b0
--	--	----	-----	--	------

					'0': Send command. Setting Auto 8 null Clock/Command/Response Transfer Function '1': enable automatic 8 null clock/command/response transfer '0': Disables automatic 8 null clock/command/response transfer. Depending on the bit 5 and bit 3 settings, 8 null clocks are generated to receive a response or transmit a command. This bit is cleared automatically when the transmission is completed;	
		RW	[1]		Set data transfer direction '1' : read data; . '0' : Write data;	1'b0
		RW	[0]		Setting up automatic data transfer '1' : Enable automatic data transfer '0' : Disable automatic data transfer. This bit is automatically cleared when the data transfer is complete;	1'b0
0x08	mmc_bytectl		RW	[15:0]	Data Transfer Byte Count Register	16'h0200

0x0C	mmc_tr_blockcnt		RO	[15:0]	Completed block counter for multiple block transfers	16'h0000
0x10	mmc_crcctl		RW	[7]	SD/MMC/SDIO port CMD Line CRC circuit enable. SD/MMC/SDIO port CMD line CRC function	1'b0

				(usually used in the negative) have the possibility of '1': enable. '0': Close.	
	RW	[6]	SD/MMC/SDIO Port Data Line CRC Function '1'. Enable. '0': Close.	1'b0	
	RW	[5]	Enable automatic CRC checking crc_status '1': enable, when crc_status != 3'b010, a crc status interrupt is generated, the write data transfer will be interrupted by the stop command, and mmc_io[0] or mmc_io_mbctl[2:0] will be cleared; 0: Close;	1'b0	
	RW	[4]	Read multiple blocks before response function '1': enable. '0' : Close	1'b0	
	RW	[3:2]	DAT CRC selection. DAT CRC Selection Refer to Table 4	1'b0	
	RO	[1]	CMD CRC error.	1'b0	
	RO	[0]	DAT CRC error	1'b0	

0x14	cmd_crc	RSV	RO	[7]	RSV	1'b0
			RO	[6:0]	CMD CRC register value	7'd0

0x18	dat_crcl		RO	[7:0]	The DAT CRC Low Register Value	NA
0x1C	dat_crch		RO	[7:0]	The DAT CRC high register value	NA
0x20	mm_port		RW	[7]	SD/MMC/SDIO port Clock line signal.	1'b0
			RW	[6]	SD/MMC/SDIO port CMD Line Signal	1'b1
			RW	[5]	SD/MMC/SDIO port Data Line Signal	1'b1
			RW	[4]	Automatic Check Ncr Timeout Function 1': Enable auto check Ncr timeout. '0' :Disable auto check Ncr timeout.	1'b1
			RW	[3:0]	Ncr Time-out count value (number of SD/MMC/SDIO clocks).	4'hF
0x24	mmc_int_mask	RSV	RO	[15:9]		7'd0
				[8]	SDIO data line 1 interrupt Mask '1': not masked '0': Mask	1'b0
				[7]	CRC status token Error Interrupt Mask '1': no masking '0': Mask	1'b0
				[6]	Command and Response Ncr Timeout Interrupt Masking '1': No Masking	1'b0

				'0': Mask	
		[5]	Multi-block timeout interrupt Masking. '1': not blocked	1'b0	

				'0': Mask	
		[4]		Multi-block transfer completion interrupt Mask. '1': no masking '0': Mask	1'b0
		[3]		Command CRC Error Interrupt Masking '1': No Masking '0': Mask	1'b0
		[2]		Data CRC Error Interrupt Masking '1': No Masking '0': Mask	1'b0
		[1]		Data transfer completion interrupt mask '1': no masking '0': Mask	1'b0
		[0]		Command transfer completion interrupt mask '1': no masking '0': Mask	1'b0
0x28	clr_mmc_int	RSV	RO	[15:9]	7'd0

	RW	[8]	W: Clear SDIO data line 1 interrupt R: SDIO data line 1 interrupt status	1'b0
	RW	[7]	W: clear CRC status token error interrupt R: CRC status token error Interrupt status;	1'b0

				When this bit is 1, judge mmc_sig[6:4]	
	RW	[6]		W: clear command and response Ncr timeout interrupt; R: command and response Ncr timeout interrupt status;	1'b0
	RW	[5]		W: clear multiblock timeout interrupt; . R: Multi-block timeout interrupt status;	1'b0
	RW	[4]		W: clear multi-block transfer completion interrupt; . R: Multi- block transfer completion interrupt status;	1'b0
	RW	[3]		W: clear command CRC error interrupt; . . R: Command CRC error interrupt status;	1'b0
	RW	[2]		W: Clear data CRC error interrupt; . R: Data CRC error interrupt status;	1'b0
	RW	[1]		W: Clear data transfer completion interrupt; . R: Data transfer completion interrupt status;	1'b0

		RW	[0]	W: clear command transfer completion interrupt; . R: Command transfer completion interrupt status;	1'b0
0x2C	mmc_cardsel	RW	[7]	SD/MMC/SDIO controller enable '1': enable	1'b0

					'0': Close	
		RW	[6]		Enable SD/MMC/SDIO card device clock line '1': enable '0': Close	1'b1
		RW	[5:0]		SD/MMC/SDIO Time Base Factor Use this register to establish a 1MHz clock; $1\text{MHz} = \text{Fhclk} / ((\text{mmc_cardsel}[5:0] + 1) * 2)$	6'd0
0x30	mmc_sig		RW	[7]	SD/MMC/SDIO port CMD Line Signal When this register is read, the SDIO controller will generate a clock pulse on the clock line. The state of the CMD line at the rising edge of the clock will be stored in this register.	1'b1
		RW	[6:4]		CRC status[2:0] When writing data CRC status token;	3'b111
		RW	[3]		SD/MMC/SDIO port DAT3 data signal.	1'b1
		RW	[2]		SD/MMC/SDIO port DAT2 data signal.	1'b1
		RW	[1]		SD/MMC/SDIO port DAT1 Data signal.	1'b1
		RW	[0]		SD/MMC/SDIO port DAT0 Data signal.	1'b1
0x34	mmc_io_mbctl		RW	[7:6]	SD/MMC/SDIO NAC timeout range selection	2'b0

				Refer to Appendix 2 - Table 5.	
	RW	[5:4]		<p>SD/MMC/SDIO Busy timeout scale selection.</p> <p>SD/MMC/SDIO device busy timeout range selection.</p> <p>Refer to Appendix 2 - Table 6</p>	2'd1
	RW	[3]		<p>SD/MMC/SDIO port Clock line polarity 1: Clock falling edge transmit, rising edge capture;</p> <p>0: Clock rising edge transmit, falling edge capture;</p>	1'b0
	RW	[2]		<p>Setting up SD/MMC/SDIO ports for fully automatic command and multi-block transfers</p> <p>'1': enable</p> <p>'0': Close</p> <p>Setting this bit to 1 (mmc_io[7:6]==11) triggers an SD/MMC/SDIO command, response, 8 null clock, multiple data block transfer. When the data transfer is finished, this bit will be cleared automatically.</p>	1'b0

		RW	[1]	Select multiple block data transfer direction. Set the direction of multi-block transfer '1' : read data. '0' : Write data.	1'b0
--	--	----	-----	--	------

		RW	[0]	<p>Set SD/MMC/SDIO port auto multiple block data transfer.</p> <p>Setting the SD/MMC/SDIO Port for Automatic Multi-Block Transfer</p> <p>'1' : enable.</p> <p>'0' : Close.</p> <p>Setting this bit to 1 (mmc_io[7:6]==11) triggers an SD/MMC/SDIO multi-block transfer. The number of data blocks is specified in mmc_blocknt</p> <p>This bit is set in the register. When the data transfer is complete, this bit will Automatic zeroing.</p>	1'b0
0x38	mmc_blockcnt	RW	[15:0]	<p>Data block number register.</p> <p>Configure this register definition to be used in multiple block transfers for a total of</p> <p>Number of transmission data blocks.</p>	16'h0001
0x3C	mmc_timeoutcnt	RW	[7:0]	<p>Data transfer timeout count register.</p> <p>Time= Scale* bit[7:0].</p> <p>Scale Scale Through the following</p> <p> Hosting memory</p> <p>Defined by mmc_io_mbctl[7:6]/[5:4];</p>	8'h40
0x40	cmd_buf0	RW	[7:0]	The cmd_buf byte 0. Mapped to	8'h00

					command line bit [15:8]	
--	--	--	--	--	-------------------------	--

					Command buf byte 0, mapped to command line bit[15:8]	
0x44	cmd_buf1		RW	[7:0]	The cmd_buf byte 1. Mapped to command line bit [23:16] Command buf byte 1, mapped to command line bit[23:16]	8'h00
0x48	cmd_buf2		RW	[7:0]	The cmd_buf byte 2. Mapped to command line bit [31:24] Command buf byte 2, mapped to command line bit[31:24]	8'h00
0x4C	cmd_buf3		RW	[7:0]	The cmd_buf byte 3. Mapped to command line bit [39:32] Command buf byte 3, mapped to command line bit[39:32]	8'h00
0x50	cmd_buf4		RW	[7:0]	The cmd_buf byte 4. Mapped to command line bit [47:40] Command buf byte 4, mapped to command line bit[47:40]	8'h00

0x54	cmd_buf5		RW	[7:0]	The cmd_buf byte 5. Mapped to command line bit [55:48] Command buf byte 5, mapped to command line bit[55:48]	8'h00
------	----------	--	----	-------	--	-------

0x58	cmd_buf6		RW	[7:0]	The cmd_buf byte 6. Mapped to command line bit [63:56] Command buf byte 6, mapped to command line bit[63:56]	8'h00
0x5C	cmd_buf7		RW	[7:0]	The cmd_buf byte 7. Mapped to command line bit [71:64] The command buf byte 7 is mapped to the command line. bit[71:64]	8'h00
0x60	cmd_buf8		RW	[7:0]	The cmd_buf byte 8. Mapped to command line bit [79:72] Command buf byte 8, mapped to command line bit[79:72]	8'h00
0x64	cmd_buf9		RW	[7:0]	The cmd_buf byte 9. Mapped to command line bit [87:80] The command buf byte 9 is mapped to the command line. bit[87:80]	8'h00
0x68	cmd_buf10		RW	[7:0]	The cmd_buf byte 10. Mapped to command line bit [95:88] The command buf byte 10 is mapped to the command line. bit[95:88]	8'h00
0x6C	cmd_buf11		RW	[7:0]	The cmd_buf byte 11. Mapped to command line bit [103:96]	8'h00

					The command buf, byte 11, is mapped to the command line. bit[103:96]	
0x70	cmd_buf12		RW	[7:0]	The cmd_buf byte 12. Mapped to command line bit [111:104] The command buf, byte 12, is mapped to the command line. bit[111:104]	8'h00
0x74	cmd_buf13		RW	[7:0]	The cmd_buf byte 13. Mapped to command line bit [119:112] The command buf byte 13 is mapped to the command line. bit[119:112]	8'h00
0x78	cmd_buf14		RW	[7:0]	The cmd_buf byte 14. Mapped to command line bit [127:120] Command buf byte 14, mapped to command line bit[127:120]	8'h00
0x7C	cmd_buf15		RW	[7:0]	The cmd_buf byte 15. Mapped to command line bit [135:128] Command buf byte 15, mapped to command line bit[135:128]	8'h00

0x80	buf_ctrl		RW	[15]	Data buffer clear enable 1: Trigger data buffer clear; 0: Maintained When a 1 is written, this register is automatically cleared after one clock.	1'b0
------	----------	--	----	------	--	------

				Zero;	
	RW	[14]		DMA Request Mask 0: No mask; 1: Shielding; Note: Please configure this register before enabling dma; configuring this register after enabling dma has no effect;	1'b0
	RW	[13]	RSV		1'b0
	RW	[12]		Data FIFO status signal mask configuration 1: active 0: Default Data FIFO status signal, valid high; Mask the FIFO full signal when reading a card device; mask the FIFO empty signal when writing a card device;	1'b0
	RW	[11]		Setting the buffer access direction 1: Write 0: Read	1'b0

		RW	[10]	DMA Hardware Interface Enable: 1: Enables the DMA interface; 0: AHB interface accesses the data cache; When using the DMA interface, this bit is automatically reset when the data transfer is complete (single block transfer or multiple data transfers).	1'b0
--	--	----	------	--	------

					(block transfer)	
		RW	[9:2]		Data Buffer Data waterline setting; valid only when buf_ctl[10]=1; Note: The data cache depth is 128 words, do not configure this register to be greater than 127.	8'd0
		RO	[1]		Data cache empty signal	1'b1
		RO	[0]		Data cache full signal	1'b0
0x100~ 0x2FF	data_buf		RW		data cache	NA

Bit5	Bit4	Bit3	Speed
0	0	0	1/2 base clock
0	0	1	1/4 base clock
0	1	0	1/6 base clock
0	1	1	1/8 base clock
1	0	0	1/10 base clock
1	0	1	1/12 base clock
1	1	0	1/14 base clock
1	1	1	1/16 base clock

Table 2 Mmc_ctrl[5:3] Detailed Definitions

Note: When mmc_ctrl[6] = 1 and the controller works in high speed

mode, base clk = hclk; when mmc_ctrl[6]=0 and the controller

works in low speed mode, base clk = clk1m; Clk1m=

Fhclk/((mmc_cardsel[5:0]+1)*2);

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	operating mode	transfer byte
x	x	x	x	x	0	x	0	no operation	N/A
x	x	1	x	0	Trig	x	0	Generate 8 null clk	N/A
0	0	0	x	0	Trig	x	0	Send command	6
0	0	0	0	1	Trig	x	0	receive a response	6
0	0	0	1	1	Trig	x	0	receive a response	17

1	0	0	0	0	Trig	x	0	Transfer command + generate 8 null clk	N/A
---	---	---	---	---	------	---	---	--	-----

0	1	0	0	0	Trig	x	0	Transmit Command + Receive Ring agree (to do sth)	N/A
1	1	0	0	0	Trig	x	0	Transmit Command + Receive Ring Should + generate 8 null clk	N/A
x	x	x	x	x	0	1	Trig	Read single data +8 null clk	mmc_bytectn
x	x	x	x	x	0	0	Trig	Write single data +8 null clk	mmc_bytectn

Table 3 Mmc_io[7:0] Detailed Definitions

Note:

1. All but the last two rows of Table 3 generate a CMD DONE interrupt;
2. The last two lines of Table 3 generate a data transfer completion interrupt.

Bit3	Bit2	The data_crcl and data_crch registers show the contents of the
0	0	DAT0 On-Line Data CRC Value
0	1	DAT1 On-Line Data CRC Value
1	0	DAT2 On-Line Data CRC Value

1	1	DAT3 On-Line Data CRC Value
---	---	-----------------------------

Table 4 mmc_crctrl[3:2] Detailed Definitions

Bit7	Bit6	Bit2	Bit1	Bit0	Operation Description	Number of bytes transferred
mmc_io		mmc_io_mbctl				
1	1	Trig	0	0	Write multi-block command + response + 8 null clock + data	mmc_blockcnt
1	1	Trig	1	0	Read Multiblock Command + Response + 8 null clock + data	mmc_blockcnt
x	x	0	0	Trig	Write multiple blocks of data	mmc_blockcnt
x	x	0	1	Trig	Read multiple blocks of data	mmc_blockcnt

Table 5 mmc_io[7:6] and mmc_io_mbctl[2:0] Detailed Definitions

Note:

1. The first two columns of operations in Table 5 generate a multi-block data completion interrupt, a data completion interrupt for each block of data, and a CMD DONE interrupt;
2. When a timeout occurs, the multiblock data completion interrupt will not be generated, but a timeout interrupt will be generated instead.

Bit7	Bit6	time unit
------	------	-----------

0	0	1us
0	1	100us
1	0	10ms
1	1	1s

Table 6 mmc_io_mbctl[7:6] NAC Timeout Interrupt Unit Selection

Bit7	Bit6	time unit
0	0	1us
0	1	100us
1	0	10ms
1	1	1s

Table 7 mmc_io_mbctl[5:4] Port Timeout Interrupt Unit Selection

Note:

1. When using the DMA interface, DMA enable needs to be turned on first;
2. If interrupts are not used, mmc_io[2] can be queried when transferring a command/response/8 null clock; mmc_io[0] can be queried when data needs to be transferred; mmc_io_mbctl[2] can be queried when transferring multiple blocks of data
/mmc_io_mbctl0];
3. When the Ncr timeout occurs, the data transfer is interrupted and the controller needs to reconfigure a new transfer;

Winner Micro

14 SPI Controller

14.1 Functional overview

SPI stands for Serial Peripheral Interface, which is a high-speed, full-duplex, synchronous communication bus. the communication principle of SPI is very simple, it works in a master-slave mode, which usually has one master device and one or more slave devices, and requires at least 4 wires, in fact, 3 wires are also possible (for unidirectional transmission), including SDI (data input), SDO (data output), SCLK (clock), CS (chip select). These include SDI (data in) SDO (data out) SCLK (clock) CS (chip select)

14.2 Main characteristics

- Can be used as either an SPI master or SPI slave device
- 8-word deep FIFOs for each of the transmit and receive paths
- master supports 4 formats of motorola spi (CPOL, CPHA) TI timing, macrowire timing
- slave supports 4 formats of motorola spi (CPOL, CPHA)
- Supports full and half duplex
- The master device supports bit transfers up to 65535bit.
- Slave devices support various byte-length transfer modes
- The maximum clock frequency of spi_clk input from the device is 1/6 of the system APB clock.

14.3 Functional Description

14.3.1 master-slave mating

The SPI controller supports both the device as an SPI communication master and as an SPI

communication slave. By setting Bit2 of the SPI_CFG register, you can switch the device master and slave roles back and forth.

14.3.2 Multi-mode support

As a master device, you can set Bit1 (CPHA) and Bit0 (CPOL) of SPI_CFG register to transmit data in four formats of MOTOROLA SPI. CPOL is used to determine the level of the SCK clock signal when it is idle, CPOL=0, the idle level is low, and CPOL=1, the idle level is high. CPHA is used to determine the sampling moment, CPHA=0, sampling at the first clock edge of each cycle, CPHA=1, sampling at the second clock edge of each cycle. You can also set the master device to transmit data in TI timing or microwire timing by setting the TRANS_MODE register, and the length of transmitted data is adjustable in both timings.

As a slave device, only the four formats of MOTOROLA SPI are supported, and the format selection is realized by setting the same registers as in the case of a master device.

14.3.3 Efficient data transfer

A FIFO memory is a first-in, first-out dual-port buffer, meaning that the first data to go into it is the first to be shifted out, with one of the memory's input ports and the other being the memory's output port. The SPI controller integrates two (one for each transceiver) FIFO memories, each with a depth of 8 words, to increase data transfer rates, handle large data streams, and match systems with different transfer rates, thereby improving system performance. The trigger level of RXFIFO and TXFIFO can be set by setting Bit[8:6] and Bit[4:2] in the MODE_CFG register to meet the performance requirements at different transfer rates, and when the trigger level of FIFO is triggered, an interrupt or a DMA can be triggered to move the data from memory to TXFIFO or to move the data to TXFIFO. When the trigger level of the FIFO is triggered, an interrupt or DMA can be triggered to move data from memory to TXFIFO or from RXFIFO to memory.

14.4 register description

14.4.1 register list

Table 109 SPI Register List

offset address	name (of a thing)	abridge	intervie ws	descriptive	reset value
0X0000	Channel Configuration Register	CH_CFG	RW	This is used to perform a number of transceiver channel related entries configure	0X0000_0000
0X0004	SPI Configuration Register	SPI_CFG	RW	Configuring SPI communication items	0X0000_0004
0X0008	Clock Configuration Register	Clk_CFG	RW	Used to set the clock division factor	0X0000_0000
0X000C	Mode Configuration Register	MODE_CFG	RW	Configuring the Transfer Mode	0X0000_0000
0X0010	Interrupt Control Register	SPI_INT_MASK	RW	Mask or enable related interrupts	0X0000_00FF
0X0014	Interrupt Status Register	SPI_INT_SOURCE	RW	For querying the interrupt source	0X0000_0000
0X0018	SPI Status Register	SPI_STATUS	RO	List the relevant states in SPI communication	0X0000_0000
0X001C	SPI Timeout Register	SPI_TIME_OUT	RW	Setting the SPI communication timeout	0X0000_0000
0X0020	Data Transmission Register	SPI_TX_DATA	RW	TX FIFO for holding data to be sent	0X0000_0000
0X0024	Transmission Mode Register	TRANS_MODE	RW	Setting the transmission mode	0X0000_0000
0X0028	Data Length Register	SLV_XMIT_LEN	RO	When used as a slave device, it is used to store the data sent out or The length of the data received by the	0X0000_0000
0X002C		RSV		reservations	0X0000_0000
0X0030	Data Receive Register	SPI_RX_DATA	RW/RO	RX FIFO to hold received data	0X0000_0000

14.4.2 Channel

Configuration

Register

Table 110 SPI Channel Configuration Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31]		RSV	1'b0
[30:23]	RW	RX_INVALID_BIT	8'h0

		<p>Indicates how many bits are invalid data when the receive path starts to receive, these invalid data need to be thrown away directly without entering the Rx FIFO, only the subsequent data enter the Rx FIFO.</p> <p>This register is used in conjunction with Tx/Rx length. The final amount of data actually deposited into the Rx FIFO is Tx/Rx length - RX_INVALID_BIT</p> <p>Note: Master mode active Motorola/ TI mode active</p>	
[22]	RW	<p>Clear FIFOs, clear the contents of Tx and Rx FIFOs, and synchronously reset all circuits of master and slave.</p> <p>(except configuration registers) 1'b0: do not clear FIFOs 1'b1: Clearance effective Software set to 1, hardware clear to 0</p> <p>Note: master/slave are valid Motorola/TI/microwire mode is valid</p>	1'b0

[21]	RW	<p>continue mode, in which spi transmissions are not affected by empty Tx FIFOs.</p> <p>In this mode, the spi transmission is not affected by the empty Tx FIFO and continues until the whole transmission process is completed.</p> <p>1'b0: normal, Tx FIFO empty, need to wait for data in the FIFO, SCK stop flip-flop, similarly, Rx FIFO full, SCK stop flip-flop, wait for RX FIFO have space to receive data</p> <p>1'b1: continue mode, Tx fifo is empty, still can transmit until transmission is completed, but at this time, if rx fifo is full, then need to pause transmission until rx fifo can store number</p> <p>Note: Master effective</p> <p>Normally, this mode is not set.</p>	1'b0
------	----	--	------

		<p>When this mode is enabled, if there is no data in tx fifo, it may cause invalid data to be sent out first. Therefore, please fill in the data first, and then start spi master.</p> <p>Motorola/TI/microwire modes are valid.</p>	
[20]	RW	<p>RxChOn, receive channel on or off 1'b0: Rx channel off 1'b1: Rx channel on</p> <p>Note: master/slave are valid</p> <p>Motorola/TI/microwire modes are valid</p>	1'b0
[19]	RW	<p>TxChOn, transmit channel on or off 1'b0: Tx channel off 1'b1: Tx channel on</p> <p>Note: master/slave are valid</p> <p>Motorola/TI/microwire modes are valid</p>	1'b0

[18:3]	RW	<p>Tx/Rx length</p> <p>Spi Number of valid SCKs during transmission</p> <p>It also indirectly reflects the length of the sent or received data.</p> <p>When (TxChOn=1, RxChOn=1), it indicates the number of bits to send, and the maximum number of bits to receive (exactly how many to receive is related to RX_INVALID_BIT)</p> <p>When (TxChOn=1, RxChOn=0), it indicates the number of bits sent.</p> <p>When (TxChOn=0, RxChOn=1), the</p> <p>Indicates the maximum number of received bits (the exact number of received bits is related to RX_INVALID_BIT, and the actual number of received bits is Tx/Rx).</p>	16'h0
--------	----	---	-------

		<p>length - RX_INVALID_BIT)</p> <p>Meaningless when (TxChOn=0, RxChOn=0).</p> <p>Note: Master effective</p> <p>Motorola/TI/microwire modes are valid.</p>	
[2]	RW	<p>Chip selects</p> <p>1'b0: SPI_CS valid signal is 0</p> <p>1'b1: SPI_CS valid signal is</p> <p>1 Note: master valid</p> <p>Motorola/TI/microwire modes are valid.</p>	1'b0
[1]	RW	<p>Force CS out</p> <p>1'b0: spi_cs signal output controlled by hardware</p> <p>1'b1: spi_cs signal output is controlled by software, the specific output value is</p> <p>Chip selects</p> <p>This signal, together with Chip selects, enables the output csn signal to be programmable, i.e., when this signal is 1, spi_cs</p> <p>= Chip selects</p> <p>Note: master is</p> <p>valid.</p> <p>Motorola/TI/microwire modes are valid.</p>	1'b0

[0]	RW	SPI start. Command SPI to start receiving or transmitting, write 1 for spi to start working, after that, auto reset to zero 1'b0: stop spi working 1'b1: one transmission or reception of the spi is initiated and automatically zeroed out Note: Master effective	1'b0
-----	----	--	------

		Motorola/TI/microwire modes are valid.	
--	--	--	--

14.4.3 SPI Configuration Register

Table 111 SPI Configuration Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:19]		RSV	13'h0
[18:17]	RW	FRAM FORMAT 2'b00:motorola 2'b01:TI 2'b10:microwire 2'b11:RSV Select which manufacturer's protocols are supported by master Note: Master effective	2'b0
[16]	RW	SPI_TX pin always driven 1'b0: spi outputs are driven only when spi_cs is active, otherwise they are tri-stated 1'b1: spi output is always driven, even without data transfer Note: valid for both master/slave Valid for Motorola/TI/microwire modes	1'b0
[15]		RSV	1'b0

[14:12]	RW	<p>cs hold, the time that spi_cs remains valid after the data transfer is completed, i.e. the hold time of spi_cs 3'b000 >= Hold time of spi_cs</p> <p>3'b000 3'b000 >=1 APB bus CLK</p> <p>3'b001 >=2 APB Bus CLK</p> <p>3'b010 >=4 APB Bus CLK</p>	3'b0
---------	----	--	------

		<p>3'b011 >=8 APB Bus CLK</p> <p>3'b100 >=16 APB Bus CLKs</p> <p>3'b101 >=32 APB Bus CLKs</p> <p>3'b110 >=64 APB Bus CLKs</p> <p>3'b111 >=127 APB bus CLKs</p> <p>Note: master is valid Motorola mode is valid</p>	
[11:9]	RW	<p>cs setup, the amount of time that spi_cs is valid before the data transfer, i.e. the setup time o f spi_cs 3'b000 >=1 APB bus CLK</p> <p>3'b001 >=2 APB buses CLK</p> <p>3'b010 >=4 APB bus CLKs</p> <p>3'b011 >=8 APB Bus CLKs</p> <p>3'b100 >=16 APB Bus CLKs</p> <p>3'b101 >=32 APB Bus CLKs</p> <p>3'b110 >=64 APB Bus CLKs</p> <p>3'b111 >=127 APB bus CLKs</p> <p>Note: Master is valid Motorola mode is valid</p>	3'b0

[8:7]	RW	SPI-out delay, SPI data output delay relative to SCK, mainly for hold time consideration. [8:7] Number of system clock cycles (APB clock) 2'b00 0 2'b01 1	2'b0
-------	----	---	------

		<p>2'b10 2</p> <p>2'b11 3</p> <p>Note: Master/slave is valid Motorola mode is valid</p>	
[6:4]	RW	<p>Frame delay, by default, the interval between the end of transmission of one frame (while spi_cs is valid) and the beginning of the next frame is half of the SCK clock period, i.e., the time when SPI_CS is invalid. However, this is configurable for compatibility.</p> <p>Default at least 0.5SCK [6:4] SCK clock</p> <p>3'b000 0</p> <p>3'b001 2</p> <p>3'b010 4</p> <p>3'b011 8</p> <p>3'b100 16</p> <p>3'b101 32</p> <p>3'b110 64</p> <p>3'b111 127</p> <p>For example, if 128byte of data is transferred in block mode, the set delay time will be added after the data transfer is completed.</p> <p>Note: Master effective</p>	3'b0
[3]	RW	<p>Big endian</p> <p>1'b0: The data format is in small-end mode, i.e., the low byte is sent first during transmission</p> <p>1'b1: The data format is in big-endian mode, i.e., during transmission, the high byte is sent first</p>	1'b0

[2]	RW	MASTER/SLAVE	1'b1
-----	----	--------------	------

		1'b0: slave, this device is a slave 1'b1: master, this device is master Note: master/slave are both valid	
[1]	RW	SPI CPHA 1'b0: Transmission mode A 1'b1: Transmission mode B Note: Master/slave is valid Motorola mode is valid	1'b0
[0]	RW	SPI CPOL, SCK polarity at IDLE 1'b0: 0 at SCK IDLE 1'b1: 1 for SCK IDLE Note: Master/slave is valid Motorola mode is valid	1'b0

14.4.4 Clock Configuration Register

Table 112 SPI Clock Configuration Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:16]		RSV	16'h0

[15:0]	RW	<p>Divider</p> <p>$f_{SCK} = F_{APB_CLK} / (2 \times (\text{Divider} + 1))$</p> <p>Note: master valid</p> <p>Motorola/TI/microwire modes are valid.</p>	16'h0
[31:16]		RSV	16'h0

[15:0]	RW	<p>Divider</p> <p>$f_{SCK} = FAPB_CLK / (2 \times (\text{Divider} + 1))$</p> <p>Note: master valid</p> <p>Motorola/TI/microwire modes are valid.</p>	16'h0
--------	----	--	-------

14.4.5 Mode Configuration Register

Table 113 SPI Mode Configuration Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:9]		RSV	23'h0
[8:6]	RW	<p>RxTrigger level</p> <p>Threshold for data stored in RX FIFO to trigger an interrupt or DMA request: 0~7word</p> <p>Only if the data in the rxbuffer is greater than the RxTrigger level will an interrupt be triggered or a DMA move be requested Note: Valid for both master/slave.</p> <p>Motorola/TI/microwire modes are valid.</p>	3'b0
[5]		RSV	1'b0

[4:2]	RW	<p>TxTrigger level</p> <p>Threshold for interrupt or DMA request triggered by data stored in TX FIFO: 0~7word</p> <p>An interrupt is triggered or a DMA move is requested only if the data in the txbuffer is greater than or equal to the TxTrigger level.</p> <p>Note: Both master/slave are valid.</p> <p>Motorola/TI/microwire modes are valid.</p>	3'b0
[1]	RW	RxDMA On with DMA shift data enable	1'b0

		<p>1'b0: without DMA.</p> <p>1'b1: using DMA</p> <p>Note: master/slave are valid</p> <p>Motorola/TI/microwire mode is valid</p>	
[0]	RW	<p>TxDMA On, DMA shift data enable adopted 1'b0: DMA not adopted.</p> <p>1'b1: using DMA</p> <p>Note: master/slave are valid</p> <p>Motorola/TI/microwire mode is valid</p>	1'b0

14.4.6 Interrupt Control Register

Table 114 SPI Interrupt Control Registers

classifier for honori- fic people	intervi- ews	Operating Instructions	reset value
[31:8]		RSV	24'h0
[7]	RW	<p>IntEn_spi_timeout</p> <p>1'b0: spi_timeout interrupt is allowed 1'b1: spi_timeout interrupt is not allowed Note: master/slave are valid</p>	1'b1

		Motorola/TI/microwire mode is valid	
[6]	RW	IntEn_spi_done 1'b0: spi send or receive complete, allow interrupt to be generated 1'b1: spi send or receive complete, no interrupt allowed	1'b1

		Note: master/slave are valid in Motorola/TI/microwire mode.	
[5]	RW	<p>IntEnRxOverrun</p> <p>1'b0: Rx FIFO overflow interrupt</p> <p>enabled 1'b1: Rx FIFO overflow</p> <p>interrupt not enabled</p> <p>Note: master/slave are valid in Motorola/TI/microwire mode.</p>	1'b1
[4]	RW	<p>IntEnRxUnderrun</p> <p>1'b0: Rx FIFO underflow interrupt</p> <p>not enabled 1'b1: Rx FIFO</p> <p>underflow interrupt enabled</p> <p>Note: master/slave are valid in Motorola/TI/microwire mode.</p>	1'b1
[3]	RW	<p>IntEnTxOverrun</p> <p>1'b0: Tx FIFO overflow interrupt</p> <p>enable 1'b1: Tx FIFO overflow</p> <p>interrupt not enable</p> <p>Note: master/slave are valid in Motorola/TI/microwire mode.</p>	1'b1
[2]	RW	<p>IntEnTxUnderrun</p> <p>1'b0: Tx FIFO underflow interrupt</p> <p>enable 1'b1: Tx FIFO underflow</p> <p>interrupt not enable</p> <p>Note: master/slave are valid in Motorola/TI/microwire mode.</p>	1'b1

[1]	RW	IntEnRx_fifoRdy 1'b0: Rx FIFO with data upload interrupt enable 1'b1: Rx FIFO with data upload interrupt not enabled Note: master/slave are valid in Motorola/TI/microwire mode.	1'b1
[0]	RW	IntEnTx_fifoRdy	1'b1

		<p>1'b0: Tx FIFO can write data to TX FIFO.</p> <p>1'b1: Tx FIFO can write data to TX FIFO, interrupt is not enabled.</p> <p>Note: master/slave are valid in Motorola/TI/microwire mode.</p>	
--	--	--	--

14.4.7 Interrupt Status Register

Table 115 SPI Interrupt Status Register

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:8]		RSV	24'h0
[7]	RW	<p>spi_timeout</p> <p>1'b0: There is no ending data in rxfifo that needs to be fetched by the CPU.</p> <p>1'b1: rxfifo has ending data that needs to be picked up by the CPU to write 1 to clear it.</p> <p>Note: master/slave are valid in Motorola/TI/microwire mode.</p>	1'b0
[6]	RW	<p>spi_done</p> <p>1'b0: SPI send or receive not completed</p> <p>1'b1: SPI send or receive completion write 1 clear</p> <p>Note: master/slave are valid in Motorola/TI/microwire mode.</p>	1'b0

[5]	RW	RxOverrun 1'b0: Rx FIFO overflow 1'b1: Rx FIFO overflow write 1 clear	1'b0
-----	----	--	------

		Note: master/slave are valid in Motorola/TI/microwire mode.	
[4]	RW	<p>RxUnderrun</p> <p>1'b0: Rx FIFO underflow</p> <p>1'b1: Rx FIFO underflow</p> <p>write 1 clear</p> <p>Note: master/slave are valid in Motorola/TI/microwire mode.</p>	1'b0
[3]	RW	<p>TxOverrun</p> <p>1'b0: Tx FIFO overflow</p> <p>1'b1: Tx FIFO overflow</p> <p>write 1 cleared</p> <p>Note: master/slave are valid in Motorola/TI/microwire mode.</p>	1'b0
[2]	RW	<p>TxUnderrun</p> <p>1'b0: Tx FIFO underflow</p> <p>1'b1: Tx FIFO underflow</p> <p>write 1 cleared</p> <p>In the case of continue mode = 1, this interrupt is never generated.</p> <p>Note: master/slave are valid in Motorola/TI/microwire mode.</p>	1'b0
[1]	RW	<p>RxFifoRdy</p> <p>1'b0: Rx FIFO data volume <= RxTrigger level, no upload required 1'b1: Rx FIFO data volume > RxTrigger level, upload required</p> <p>Write 1 Clear</p> <p>Note: master/slave are valid in Motorola/TI/microwire mode.</p>	1'b0

[0]	RW	<p>TxFifoRdy</p> <p>1'b0: Tx FIFO data amount > TxTrigger level, can't write data to TX FIFO 1'b1: Tx FIFO data amount <= TxTrigger level, can write data to TX FIFO Write 1 clear</p> <p>Note: master/slave are valid in Motorola/TI/microwire mode.</p>	1'b0
-----	----	--	------

14.4.8 SPI Status Register

Table 116 SPI Status Register

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:13]		RSV	19'h0
[12]	RO	<p>SPI Busy</p> <p>1'b0: SPI has no send and receive tasks 1'b1: SPI in transmit or receive process</p> <p>Note: master/slave are valid in Motorola/TI/microwire mode.</p>	1'b0
[11:6]	RO	<p>RX FIFO fill level</p> <p>Amount of data in Rx FIFO in bytes</p> <p>Note: master/slave are valid in Motorola/TI/microwire mode.</p>	6'h0
[5:0]	RO	<p>Tx FIFO fill level</p> <p>Amount of data in Tx FIFO in bytes</p> <p>Note: master/slave are valid in Motorola/TI/microwire mode.</p>	6'h0

14.4.9 SPI Timeout Register

Table 117 SPI Timeout Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31]	RW	<p>spi_timer_en 1'b0: timer not allowed</p> <p>1'b1: timer allowed</p> <p>Note: master/slave are valid in Motorola/TI/microwire mode.</p>	1'b0
[30:0]	RW	<p>SPI_TIME_OUT</p> <p>When a transmission is completed, in the receive path rxfifo, the ending data needs to be timed to notify the CPU to move the ending data if it cannot trigger the receive interrupt RxFifoRdy or DMA request.</p> <p>Specific method: when rxfifo is in idle state (no read/write operation, no dma request, cs is invalid, there is data in rxfifo, and the amount of data is less than or equal to RxTrigger level), it will start counting, and when it reaches the value set in this register, it will trigger the timeout interrupt to request the CPU to move out the ending data.</p> <p>Any read or write operation to or from the rxfifo will clear the timeout timer. The time indicated is: $T = \text{SPI_TIME_OUT}/\text{FAPB_CLK}$</p> <p>Note: master/slave are valid in Motorola/TI/microwire mode.</p>	31'h0

14.4.10 Data Transmission Register

Table 118 SPI Data Transmission Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value

[31:0]	RW	Window address for writing data to the Tx FIFO Note: master/slave are both valid. Motorola/TI/microwire modes are valid.	32'h0
--------	----	--	-------

14.4.11 Transmission Mode Register

Table 119 SPI Transfer Mode Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:30]		RSV	16'd0
[29:24]	RW	TI_BLK_LEN In TI's timing mode, the length of each block transfer, i.e., the length of data transferred after each CS is valid. Support 4~32bit 6'h4: 4bit long data 6'h5: 5bit long data 6'h6: 6bit long data 6'h20: 32bit long data note: master valid TI mode active	6'd0

[16]	RW	MICRO_BURST 1b'1: In Microwire mode, burst transmission is used, i.e., the Tx sends the control word and the Rx receives the data, alternating in sequence, MICRO_CONTROL_LEN indicates the length of the control word, and MICRO_DAT_LEN indicates the length of the data word.	1'b0
------	----	---	------

	<p>is the length of the transmit or receive word, Tx/Rx length indicates the effective sck during the whole transmission process, and in burst mode, the number of times the transmit and receive alternately is $(Tx/Rx \text{ length})/(MICRO_CONTROL_LEN+ MICRO_DAT_LEN+1)$</p> <p>1'b0: Microwire mode without burst transmission In this mode, there are two cases</p> <p>1) tx_ch_on = 1, rx_ch_on = 0, at this time, only send, MICRO_CONTROL_LEN is indicated by the length of the control word, Tx/Rx length is indicated by the effective sck of the whole transmission process, at this time, the length of the data sent is $m * MICRO_DAT_LEN = Tx/Rx \text{ length} - MICRO_CONTROL_LEN$, where m indicates how many (MICRO_DAT_LEN)-length words are sent.</p> <p>2) tx_ch_on = 1, rx_ch_on = 1, at this time, Tx sends the control word, Rx receives the data, MICRO_CONTROL_LEN indicates the length of the control word, Tx/Rx length indicates the effective sck of the whole transmission process, and the length of the received data is $m * MICRO_DAT_LEN = Tx/Rx \text{ length}-1$, where m indicates how many words of (MICRO_DAT_LEN) length are received. MICRO_CONTROL_LEN-1, where m indicates how many (MICRO_DAT_LEN)-length words are received Note: master is valid.</p> <p>The microwire model works</p>	
--	---	--

[13:8]	RW	MICRO_DAT_LEN In Microwire mode, the length of each burst transmission data is from 1~32 when in burst mode mode: 6'h1: 1bit long data 6'h2: 2bit long data	6'd0
--------	----	---	------

		<p>6'h3: 3bit long data</p> <p>.....</p> <p>6'h20 32bit long</p> <p>data note: master</p> <p>valid microwire</p> <p>mode valid</p>	
[5:0]	RW	<p>MICRO_CONTROL_LEN</p> <p>In Microwire mode, the length of the command word is from 1 to 32:</p> <p>6'h1: 1bit long</p> <p>command 6'h2:</p> <p>2bit long</p> <p>command 6'h3:</p> <p>3bit long</p> <p>command</p> <p>.....</p> <p>6'h20 32bit Long</p> <p>Command Note:</p> <p>master valid</p> <p>microwire mode</p> <p>valid</p>	6'd0

 14.4.12 Data length

registers

Table 120 SPI Data Length Register

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:16]	RO	<p>The length of data sent out during the validity period of cs as a slave, in bitsNote: Valid for slaves.</p> <p>Motorola mode active</p>	16'h0

[15:0]	RO	Length of data received in bits during the validity period of cs as a slave Note: Slave validity Motorola mode active	16'h0
--------	----	---	-------

14.4.13 Data Receiving Registers

Table 121 SPI Data Receive Registers

classi fier for hono rific peop le	intervi ews	Operating Instructions	reset value
[31:0]	RO	Window address for reading data from Rx FIFO Note: valid for both master/slave Motorola/TI/microwire modes are valid.	32'h0

15 I2C Controller

15.1 Functional overview

The I2C bus is a simple, bi-directional two-wire synchronous serial bus. It requires only two wires to transfer information between devices connected to the bus.

The master device is used to start the bus transferring data and generates a clock to open the device for transmission, at which point any addressed device is considered to be a slave device. The relationship between master and slave, transmitter and receiver on the bus is not constant, but depends on the direction of data transmission at this time. If the host wants to send data to the slave device, the host first addresses the slave device, then actively sends data to the slave device, and finally the host terminates the data transmission; if the host wants to receive data from the slave device, the master device first addresses the slave device. Then the host receives the data sent from the slave device, and finally the host terminates the receiving process. In this case. The host is responsible for generating the timing clock and terminating the data transfer.

15.2 Main

character

istics

- APB bus protocol standard interface
- Can only be used as a master device controller
- I2C work rate can be configured, 100KHz~400KHz
- Multiple GPIOs can be multiplexed into I2C communication interfaces.
- Fast output and detection of timing signals

15.3 Functional Description

15.3.1 Transmission Rate Selection

Setting register PRERlo and register PRERhi configures the data rate on the I2C bus from 100KHz to

An integer divider value of any bus frequency between 400KHz.

15.3.2 Interrupt and start-stop controllable

The I2C controller can be allowed or disabled to generate interrupts by setting Bit6 of register CTR, and the I2C controller can be started or stopped at any time by setting Bit7.

15.3.3 Fast output and detection signals

Setting the corresponding bit of register CR_SR enables the controller to quickly output or detect the bus START signal, bus STOP signal, bus ACK signal, and bus NACK signal. In master mode, the I2C interface initiates the data transfer and generates the clock signal. A serial data transfer always starts with a start signal and ends with a stop signal. Once the start signal has been generated on the bus, the master device mode is selected.

15.4 register description

15.4.1 register list

Table 122 I2C Register List

offset address	name (of a thing)	abridge	intervie ws	descriptive	reset value
0X0000	Clock Divider Register_1	PRERlo	RW	Stores the lower 8 bits of the crossover frequency value to be used for the APB Bus Clock Dividing	0X0000_00FF
0X0004	Clock Divider Register_2	PRERhi	RW	Stores the high 8-bit crossover frequency value for the APB Bus Clock Dividing	0X0000_00FF

0X0008	control register	CTR	RW	Enable to control interrupts and I2S control. Enabling of the device	0X0000_0040
0X000C	data register	TXR_RXR	RW	Used to store data to be sent or received	0X0000_0000

				data	
0X0010	Transceiver Control Register	CR_SR	RW	Used to control some data read and write related operations	0X0000_0000
0X0014	TXR Readout register	TXR	RO	Read TXR register value on I2C transmit	0X0000_0000
0X0018	CR Read Register	CR	RO	Read the set value of the I2C control register CR	0X0000_0000

15.4.2 Clock Divider Register_1

Table 123 I2C Clock Divider Register_1

classif ier for hono rific peop le	intervi ews	Operating Instructions	reset value
[31: 8]		reservations	
[7 : 0]	RW	<p>The lower 8 bits of the clock divider configuration prescale.</p> <p>for example:</p> <p>apb_clk=40MHz, SCL=100KHz</p> $\text{prescale} = (40*1000)/(5*100) - 1 = 16'd79$ <p>apb_clk = 40M, SCL=400K</p> $\text{prescale} = (40*1000)/(5*400) - 1 = 16'd19$	8'hff

15.4.3 Clock Divider

Register_2

Table 124 I2C Clock Division Register_2

classi fier for hono rific peop le	intervi ews	Operating Instructions	reset value
[31: 8]		reservations	
[7 : 0]	RW	<p>Clock divider configuration prescale is 8bit higher.</p> <p>Example:</p>	8'hff

		<p>apb_clk=40MHz, SCL=100KHz</p> <p>prescale = $(40*1000)/(5*100) - 1 = 16'd79$</p> <p>apb_clk = 40M, SCL=400K</p> <p>prescale=(40*1000)/(5*400) - 1 = 16 'd19</p>	
--	--	---	--

15.4.4 control register

Table 125 I2C Control Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:8]		reservations	
[7]	RW	I2C enable control, 1'b0: not enabled 1'b1: enable	1'b0
[6]	RW	Interrupt MASK, 1'b0: allow interrupt 1'b1: Interrupt generation not allowed	1'b1
[5:0]		reservations	

15.4.5 Data registers

Table 126 I2C Data Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:8]		reservations	
[7:0]	WR	Write this register for the transmit register TXR, the	8'h0

		<p>Indicates the next byte to be sent when is the device address</p> <p>[0]: 1 for read, 0 for write.</p> <p>When this register is read, it is the receive register RXR, for the latest byte received from the I2C.</p>	
--	--	--	--

15.4.6 Transceiver Control Register

Table 127 I2C Transceiver Control Registers

classifier for honori- c people	intervi- ews	Operating Instructions	reset value
[31:8]		reservations	

[7:0]	WR	<p>When this register is written, it is CR and functions as follows: [7]:</p> <p>STA, controls the generation of START timing; 1'b0: invalid 1'b1: Generate START Timing</p> <p>[6]: STO, control to generate STOP timing; 1'b0: invalid 1'b1: Generate STOP timing</p> <p>[5]: RD, read from SLAVE; 1'b0: invalid</p>	8'h0
-------	----	--	------

	<p>1'b1: read from SLAVE</p> <p>[4]: WR, write to SLAVE;</p> <p>1'b0: invalid</p> <p>1'b1: Write to SLAVE</p> <p>[3]: control to send back ACK/NACK to SLAVE; 1'b0: return ACK</p> <p>1'b1: back to NAK</p> <p>[2:1]: Reserved;</p> <p>[0]: IACK, clear interrupt status, 1 valid; 1'b0: invalid</p> <p>1'b1: clear interrupt flag</p> <p>When this register is read, it is SR and functions as follows:</p> <p>[7]: RxACK, ACK/NACK status received from SLAVE; 1'b0: ACK received from SLAVE</p> <p>1'b1: NAK received from SLAVE</p> <p>[6]: BUSY;</p> <p>1'b0: STO postback 0</p>	
--	---	--

		<p>1'b1: STA post 1</p> <p>[5]: AL, Arbitration Lost, this bit is reserved; [4:2]: Reserved;</p> <p>[1]: TIP;</p> <p>1'b0: no transmission in progress</p> <p>1'b1: there is a transmission in progress</p> <p>[0]: IF, interrupt status bit; 1'b0: no interrupt generation</p> <p>1'b1: Set to 1 when transmission is completed or AL is active.</p>	
--	--	---	--

15.4.7 TXR Readout register

Table 128 I2C TXR Readout Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:8]		reservations	
[7:0]	RO	<p>Read-only, TXR register readout value.</p> <p>See TXR_RXR register for functional description;</p>	8'h0

15.4.8 CR Read Register

Table 129 I2C CR Readout Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value

[31:8]		reservations	
[7:0]	RO	Read-only, CR register readout value. See CR_SR register for functional description;	8'h0

16 I2S Controller

16.1 Functional overview

I2S (Inter-IC Sound) is a bus standard for audio data transmission between digital audio devices such as CD players, digital sound processors, and digital TV sound systems. It is designed to transmit clock and data signals on separate wires. By separating the data and clock signals, it avoids distortion induced by time differences and saves the user the cost of purchasing specialized equipment that resists audio jitter. The standard I2S bus cable consists of three serial wires: one for time-division multiplexing (TDM) data, one for word select, and one for clock.

16.2 Main characteristics

- Implemented I2S interface, supports I2S and PCM protocols
- Supports amba APB bus interface, 32bit single read/write operation
- Supports master-slave mode
- Supports 8, 16, 24, and 32 bit widths with a maximum sampling frequency of 192KHz.
- Supports mono and stereo modes
- Compatible with I2S and MSB justified data formats, compatible with PCM A/B format
- Supports DMA request read/write operations, word-by-word only

16.3 Functional Description

16.3.1 Multi-mode support

The data format can be set to I2S format, MSB justified format, PCM A format, or PCM B format by setting Bit[25:24] of the I2S Control register; mono or stereo mode can be selected by setting Bit[22]

of the I2S Control register. Mono or stereo mode can be selected by setting Bit[22] of the I2S Control register.

Setting Bit[5:4] of the I2S Control register sets the bit width of the data transfer word, which can be set to 8bit, 16bit, 24bit, 32bit.

16.3.2 Zero cross detection

By setting Bit[17:16] of I2S Control register, you can set whether to enable the zero-crossing detection function of left and right channels; by setting Bit[9:8] of I2S_IMASK register, you can set whether the zero-crossing detection function of left and right channels generates an interrupt. If the detection function is enabled and an interrupt is generated, the program will execute the interrupt subroutine when the zero-crossing phenomenon is detected, and at the same time, the corresponding bits of Bit[9:8] of the I2S_INT_FLAG register will be set to one.

16.3.3 Efficient data transfer

A FIFO memory is a first-in, first-out, dual-port buffer, meaning that the first data to go into it is the first to be shifted out, with one of the memory's input ports and the other being the memory's output port. I2S controllers incorporate two FIFO memories (one for each transmitter and receiver), each with an 8-word depth, to increase the data transfer rate, handle large streams of data, and match systems with varying transfer rates, thereby improving system performance. The trigger level of RXFIFO and TXFIFO can be set by setting Bit[14:12] and Bit[11:9] in the I2S Control Register to meet the performance requirements at different transfer rates. When the trigger level of FIFO is triggered, an interrupt or DMA can be triggered to move data from memory to TXFIFO or from RXFIFO to memory.

16.4 I2S/PCM Timing Diagram

This module provides support for 4 protocols, Standard I2S, MSB Justified, PCM-A, PCM-B. Select

which protocol to use by configuring register 0x00[25:24]. The interface timing of each protocol is shown in Fig1 to Fig4.

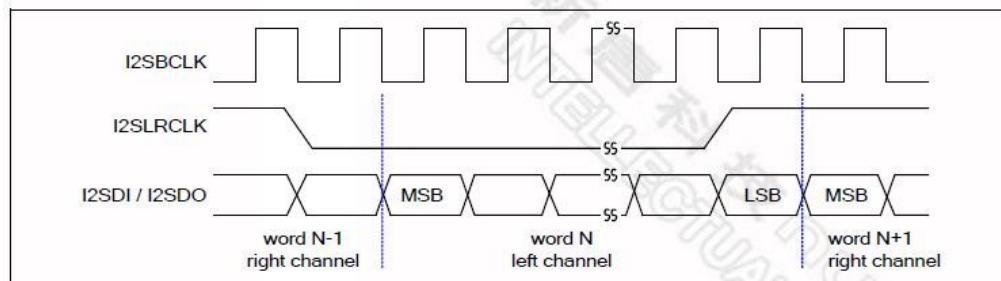


Fig1. I2S Bus Timing Diagram (PCM=0, Format=0)

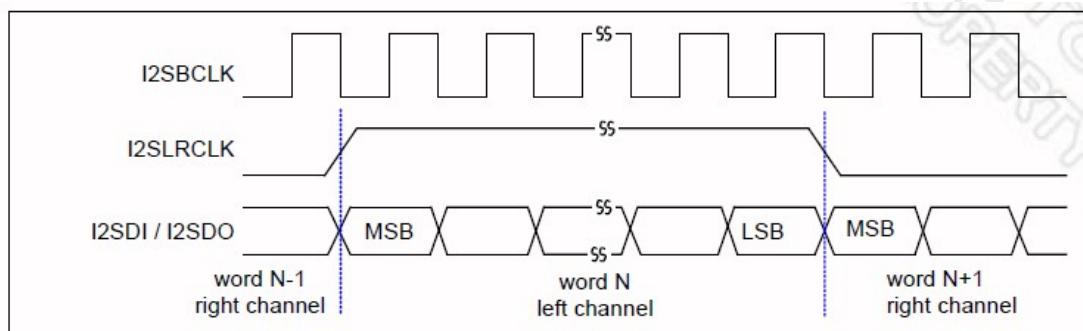


Fig2. MSB Justified Timing Diagram (PCM=0, Format=1)

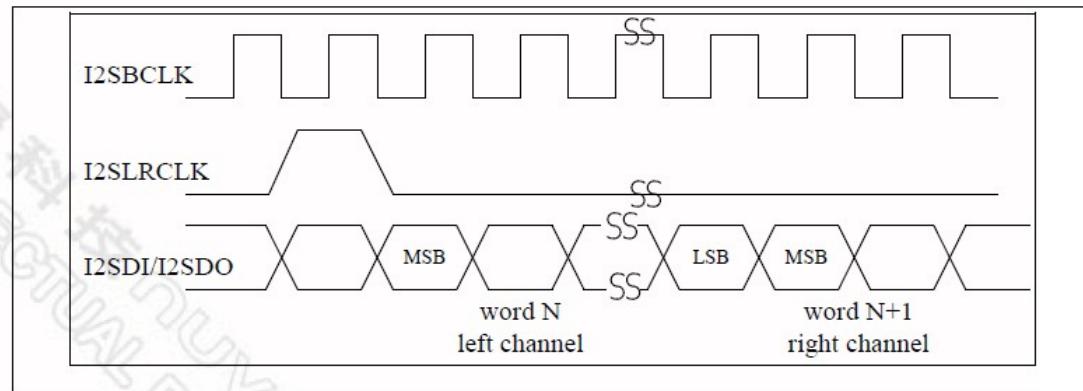


Fig3. PCM A Audio Diagram (PCM=1, Format=0)

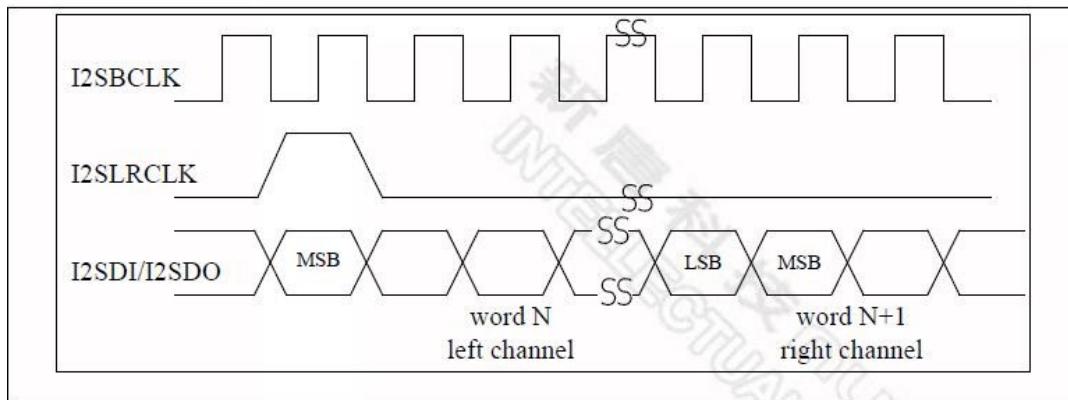


Fig4. PCM B Audio Diagram (PCM=1, Format=1)

16.5 FIFO Storage Structure Diagram

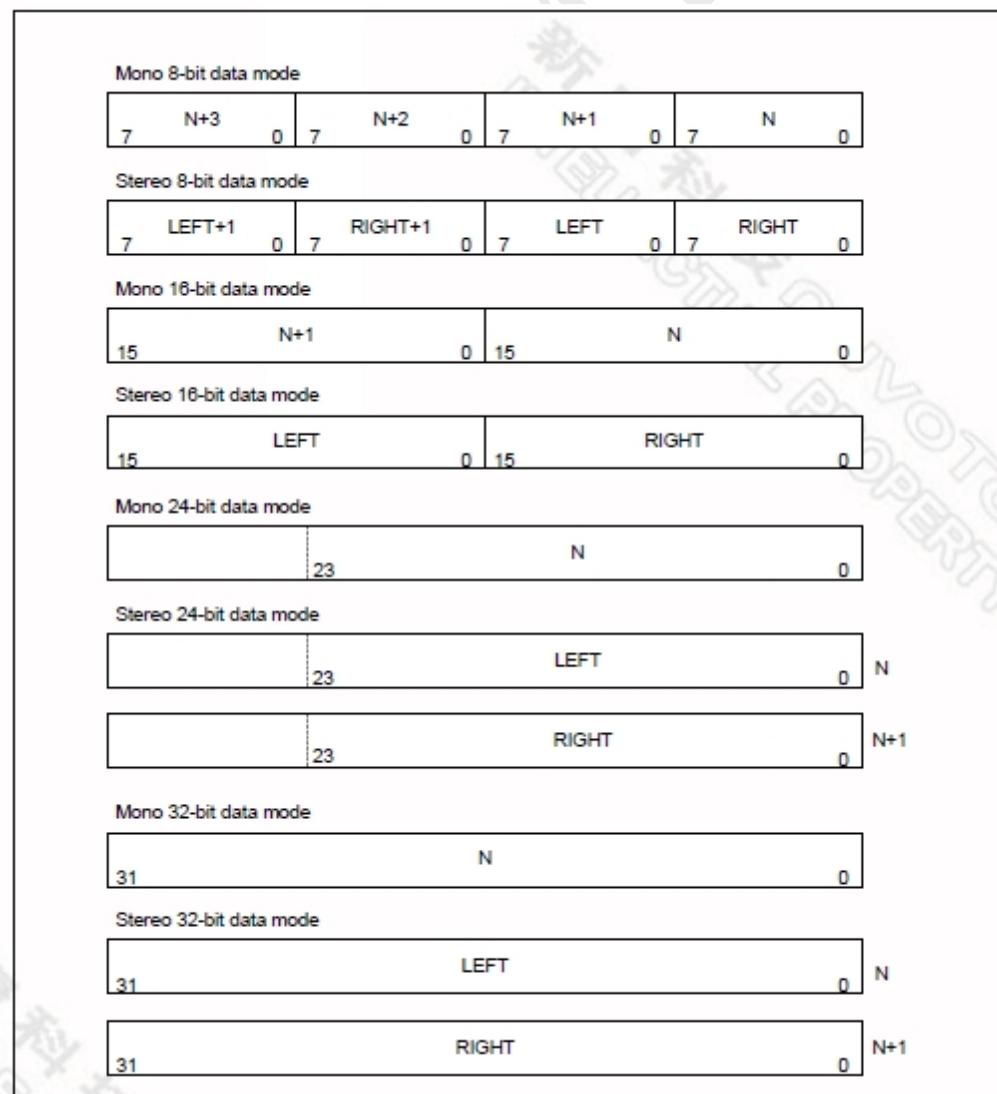


Fig 5. FIFO storage structure

The I2S module provides 2 FIFOs of 8x32bit size, one as TX FIFO and one as RX FIFO.

The storage structure of the data in the FIFO varies depending on the operating mode. When operating in mono 8-bit mode, each word in the FIFO can store 4 samples of data. When working in dual-channel 8-bit mode, the left and right channel data are stored alternately in the FIFO, with the low byte storing the right channel data and the high byte storing the left channel data. One word can store two samples. In simplex 16bit mode, one word can store two samples, in duplex 16bit mode, one word can store one sample data, the lower 16 bits are for the right channel, and the higher 16 bits are for the left channel. 24bit and 32bit modes, each word can only store one sample data or one channel data, and the specific storage method is shown in Figure 5. The specific storage method is shown in Figure 5.

16.6 I2S Module Operating Clock Configuration

The operating clock configuration of the I2S module can be configured by setting the 0x40000718 register in the Clock and Reset Module to choose whether to use an external or internal 160MHz clock for the clock source, whether to turn on the MCLK clock or not, as well as the operating frequency of MCLK and BCLK.

Setting register 0x40000718[0] allows you to select whether to use the internal 160MHz clock or an external clock as the I2S module clock source. If an external clock is selected, the I2S module will use the clock input from IO- I2S_M_EXTCLK(PA_5,Option 4) as the module clock source.

And 0x40000718[1] selects whether to turn on the MCLK clock. Bits [7:2] are areas to configure the MCLK crossover ratio. The calculation formula is as follows:

$$F_{\text{mclk}} = F_{\text{I2SCLK}} / \text{MCLKDIV}$$

F_{mclk} is the actual MCLK frequency;

F_{I2SCLK} is the clock source of the I2S module. If internal clock is used, $F_{\text{I2SCLK}} = 160\text{MHz}$; if external clock is used, F_{I2SCLK} is equal to the external input clock frequency;

MCLKDIV is the clock division ratio configured in register 0x40000718[7:2]. Note that $\text{MCLKDIV} \geq 2$.

Bits [17:8] of register 0x40000718 are areas to configure the crossover ratio of the BCLK clock. The crossover ratio calculation formula is as follows:

$$F_{\text{BCLK}} = F_{\text{I2SCLK}} / \text{BCLKDIV}$$

F_{BCLK} is the frequency at which the BCLK actually operates;

F_{I2SCLK} is the clock source of the I2S module. If internal clock is used, $F_{\text{I2SCLK}} = 160\text{MHz}$; if external clock is used, F_{I2SCLK} is equal to the external input clock frequency;

BCLKDIV is the crossover ratio to be configured. Different crossover ratios need to be selected according to different operating modes. The formula for selecting the crossover ratio is as follows:

$$\text{BCLKDIV} = \text{round}(\text{F_I2SCLK}/(\text{Fs} * \text{W} * \text{F}))$$

Fs is the sampling frequency of the audio data on the I2S interface, up to 192KHz; W is the sampling bit width, 8/16/24/32 bit can be selected; F is the mono/dual channel selection. F=1 when the transmitted data is mono, F=2 when the transmitted data is dual-channel; the final calculated crossover ratio is rounded.

The following is an example of selecting BCLKDIV according to the actual operating mode:

If the internal clock is selected as the module clock source, and 128KHz sampling rate, 24bit dual-channel data needs to be transmitted, the procedure to calculate the need to set BCLKDIV is as follows:

$$\text{BCLKDIV} = \text{round}(160 * 10e6 / 128 * 10e3 * 24 * 2) = 10'd26;$$

The crossover registers are described below:

0x18	I2S_Clk_Ctrl	[0]	RW	<p>EXTAL_EN</p> <p>External clock select</p> <p>Select whether to use External or Internal clock for I2S block.</p> <p>0=internal clk</p> <p>1=external clk</p> <p>Note: When External clock is enabled, the external clk must be 2^*N^*256 fs, where fs is sample frequency and N must be integer.</p>	1'b0
------	--------------	-----	----	--	------

	[1]	RW	MCLKEN MCLK enable 0=MCLK disabled 1=MCLK enabled	1'b0
	[7:2]	RW	MCLKDIV MCLK divider If external clock is selected, this divider is used to produce proper MCLK frequency. $F_{mclk} = F_{I2SCLK}/MCLKDIV$ Where $MCLKDIV \geq 2$ $F_{mclk} = F_{I2SCLK}$ when $MCLKDIV = 0$ Where F_{I2SCLK} is external clk. Note: F_{mclk} should be configured as 256 * fs where fs is sample frequency.	6'd0
	[17:8]	RW	BCLKDIV BCLK divider $F_{BCLK} = F_{I2SCLK}/BCLKDIV$ Note: When EXTAL_EN is not selected, Internal PLL is used and $F_{I2SCLK} = 160MHz$	10'd0

			<p>When WXTAL_EN is enabled,</p> <p>$F_{I2SCLK} = \text{External crystal frequency}$</p> <p>$BCLKDIV = \text{round}(F_{I2SCLK}/(Fs * W * F))$</p> <p>Where F_s is sample frequency of audio data and W is word width.</p> <p>$F=2$ when data is stereo and</p> <p>$F=1$ when data is mono.</p> <p>For example, if internal PLL is used and the data width is 24bit, Format is stereo format, sample frequency is 128KHz. Then the BCLKDIV should be configured as $(160 * 10e6 / 128 * 10e3 * 24 * 2) = 10'd26$</p>	
	[31:18]	RO	RSV	14'd0

16.7 Other feature descriptions:

16.7.1 Over-zero detection:

In order to avoid the noise caused by the sudden change of frequency due to data corruption, the I2S module provides the function of over-zero detection for each channel. When there is a change

in the sign bit of the two neighboring data sent, the module will generate an interrupt to remind the MCU to detect and process; meanwhile, the latter

Data forced mute.

16.7.2 Mute function

When the mute function is turned on, the data will still be sent, but the output will be forced to zero;

16.7.3 disruptions

This module provides transmit/receive completion interrupt, left and right channel over-zero detection interrupt, transmit/receive FIFO threshold interrupt (data in transmit FIFO is below the threshold, data in receive FIFO is above the threshold) and underflow/overflow interrupt of transmit/receive FIFO. The interrupt status is queried in register 0x08.

16.7.4 FIFO Status Query

Register 0x10 provides the CPU with the ability to query the status of the transmit/receive FIFOs in the I2S module. The register enables the CPU to check how much data is left to be processed in the FIFO. How many bytes of valid data are in the last word in the receive FIFO.

16.8 Data transfer

process

16.8.1 Master sends audio data

1. Configure the pins used, SDO, BCLK, LRCLK
2. Refer to Section 2.4 Setting Register 0x40000718 in the Clock and Reset Module to configure the operating clock frequency;
3. Set I2S register 0x00, set to master mode, configure the transmission format, channel selection, data bit width, whether to enable over-zero detection for left and right channels, and set transmit path-txen on.
4. Set register 0x4 to enable the interrupt you want to use;
5. If you use DMA to transmit data, select the channel to be used in the DMA module, configure the address, length of the transmitted data. Enable DMA in register 0x0 of I2S module and set the FIFO threshold. When the data in the FIFO is lower than the threshold range, it will automatically request DMA to carry the data.
6. Write the data to be transmitted to the transmit FIFO address-register 0x10, enable register 0x0[0], and the module will automatically remove the data from the FIFO and send it to the I2S bus.
7. When the data in the FIFO is less than the set threshold, the module will request data from the DMA module or send a TXTHIF interrupt.
8. When all the data in the FIFO is taken out, TXDONE interrupt will be set. When the last frame is sent, TXUDIF interrupt will be set to n o t i f y the CPU that the transmission is completed and the module stops sending.

16.8.2 Receive audio data from the slave

1. Configure the pins used, SDI, BCLK, LRCLK

16.8 Data transfer

Process I2S register 0x00, set to slave mode Configure transmission format, channel selection, data bit width, whether to enable over-zero detection for left and right channels, and set receive path-rxen on.

3. Set register 0x4 to enable the interrupt you want to use;
4. If you use DMA to transmit data, select the channel to be used in the DMA module, configure the address and length of the transmitted data. And select the channel to be used in the I2S

Enable DMA in 0x0 in the module register and set the FIFO threshold. When the data is above the threshold range, DMA is automatically requested.
Data.

5. Enable register 0x0[0], when the module detects valid BCLK and LRCLK, it will automatically capture data from SDI and store it in FIFO. When the data in the FIFO is higher than the set threshold, the module will request the DMA to carry the data to memory or send RXTHIF interrupt. The RXDONE interrupt is generated when the CPU turns off RXEN. The CPU can query how many data are in the receiving FIFO and how many bytes of valid data are in the last word through register 0x10. The CPU can then process the last remaining data according to the FIFO status.

16.8.3 Master receives audio data

1. Configure the hardware to be used, SDI , SCLK, LRCLK
2. Refer to Section 2.4 Setting Register 0x40000718 in the Clock and Reset Module to configure the operating clock frequency;
3. Set I2S register 0x00 to master mode, configure transmission format, channel selection, data bit width, whether to enable over-zero detection for left and right channels, and set receive path rxen on.
4. Set register 0x4 to enable the interrupt you want to use;
5. If you use DMA to transmit data, select the channel to be used in the DMA module, configure the address, length of the transmitted data. Enable DMA in register 0x0 of the I2S module and set the FIFO threshold. When the data is higher than the threshold range, it will automatically request DMA to carry the data.
6. By enabling register 0x0[0], the module will automatically send BCLK and LRCLK, and at the same time collect data from SDI and store it in FIFO. When the data in the FIFO is higher than

Enable DMA in 0x0 in the module register and set the FIFO threshold. When the data is above the threshold, the DMA will automatically carry the data to memory or send RXTHIF interrupt. The RXDONE interrupt is generated when the CPU closes RXEN. The CPU can query how many data are in the receiving FIFO and how many bytes of valid data are in the last word through register 0x10. The CPU can then process the last remaining data according to the FIFO status.

7. Turns off i2s enable when data reception is complete.

16.8.4 Sending audio

data from the slave side

1. Configure the pins used, SDO, BCLK, LRCLK
2. Set I2S register 0x00, set to slave mode, configure transmission format, channel selection, data bit width, whether to enable over-zero detection for the left and right channels, and set transmit path txen on.
3. Set register 0x4 to enable the interrupt you want to use;
4. If you use DMA to transmit data, select the channel to be used in the DMA module, configure the address, length of the transmitted data. Enable DMA in register 0x0 of I2S module and set the FIFO threshold. When the data in the FIFO is lower than the threshold range, it will automatically request DMA to carry the data.
5. Write the data to be transmitted to transmit FIFO address-register 0x10, enable register 0x0[0], when the module detects valid BCLK and LRCLK, the module will automatically take out the data from the FIFO and send it to the SDO.
6. When the data in the FIFO is less than the set threshold, the module will request data from the DMA module or send a TXTHIF interrupt.
7. When all the data in the FIFO is taken out, TXDONE interrupt will be set. When the last frame is sent, TXUDIF interrupt will be set to notify the CPU that the transmission is completed and the module stops sending.

16.8.5 Full-duplex mode

1. Configure the pins to be used, SDI, SDO, BCLK, LRCLK.
2. If it is master mode, you need to configure the clock division frequency.
3. Set I2S register 0x00, configure the operating mode (master/slave) configure the transmission format, channel selection, data bit width, whether or not to turn on over-zero detection for

16.8.4 Sending audio

data from the slave side.

4. Set register 0x4 to enable the interrupt you want to use;
5. If you use DMA to transmit data, select the channel to be used in the DMA module, configure the address, length of the transmitted data. Enable DMA in register 0x0 of I2S module and set the FIFO threshold. When the data in the FIFO is lower than the threshold range, it will automatically request DMA to carry the data.

6. Write the data to be emitted to transmit FIFO address-register 0x10.
7. In master mode, enable register 0x0[0], the module will start to send BCLK and LRCLK, take out data from transmit FIFO and start to send out from SDO port, and at the same time, it will receive data from SDI and store it into receive FIFO.
8. In slave mode, when the module detects valid BCLK and LRCLK, the module will automatically take out the data from the transmit FIFO and send it to the SDO, and at the same time, it will receive the data from the SDI and store it in the receive FIFO.
9. When the data in the transmit FIFO is less than the set threshold, the module will request data from the DMA module or send a TXTHIF interrupt.
10. When the data in the FIFO is higher than the set threshold, the module will either request the DMA to carry the data to memory or send an RXTHIF interrupt.
11. When all the data in TXFIFO is taken out, TXDONE interrupt will be set. When the last frame is sent, TXUDIF interrupt will be set to notify the CPU that the transmission is completed and the module stops sending.
12. The RXDONE interrupt is generated when the CPU turns off RXEN. The CPU can query how many data are in the receive FIFO and how many bytes of valid data are in the last word through register 0x10. The CPU can then process the last remaining data according to the FIFO status.

16.9 register description

16.9.1 register list

Table 130 I2S Register List

offset address	name (of a thing)	abridge	intervie ws	descriptive	reset value
-------------------	----------------------	---------	----------------	-------------	-------------

0X0000	control register	I2S Control	RW/RO	To control I2S-related functions, see the following sections for details;	0X0000_4800
0X0004	Interrupt Mask Register	I2S_IMASK	RW	Control to turn on or off all interrupts in the I2S	0X0000_03FF
0X0008	Interrupt Flag Register	I2S_INT_FLAG	RW/RO	The interrupt flag bit can be used to query whether an interrupt has been generated and to Clear the associated interrupt	0X0000_0000
0X000c	status register	I2S_STATUS	RO	Queries the status of the FIFO during I2S communication. attitude	0X0000_0000

0X0010	Data Transmission Register	I2S_TX	WO	The controller sends the data inside it to the bus	0X0000_0000
0X0014	Data Receive Register	I2S_RX	RO	The controller receives the data from the bus into it	0X0000_0000

16.9.2 control register

Table 131 I2S Control Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:29]	RO	RSV	7'h0
[28]	RW	Mode Selection 0 = master mode 1 = slave mode	1'b0
[27]	RW	Duplex mode selection 1 = Enable duplex mode 0 = Duplex mode off	1'b0
[26]	RW	Timeout Count Control Bit, when this bit is set to 1 and the transmission process is forced to stop by the master device, reception will not occur. Completion (RXDONE) Interrupt	1'h0

[25:24]	RW	<p>FORMAT</p> <p>Data format</p> <p>selection 2'b00: I2S</p> <p>data format</p> <p>2'b01: MSB Justified data</p> <p>format 2'b10: PCM A sound</p> <p>data format</p> <p>2'b11: PCM B sound data format</p>	2'b0
[23]	RW	RXLCH	1'b0

		<p>Channel receive enable</p> <p>control bit 1'b0: Enable to receive right channel data</p> <p>1'b1: Enable reception of left channel data</p> <p>Note: This bit is only valid when MONO_STEREO is selected for mono mode</p>	
[22]	RW	<p>MONO_STEREO</p> <p>Mono stereo select bit</p> <p>1'b0: data is transmitted in stereo format</p> <p>1'b1: Data is transmitted in mono format</p>	1'b0
[21]	RW	<p>RXDMAEN</p> <p>Receive DMA request</p> <p>enable bit 1'b0: do not enable transmit DMA request</p> <p>1'b1: Enable sending DMA requests</p> <p>NOTE: When the transmit DMA request is enabled and the number of words in RXFIFO is equal to or greater than RXTH, the I2S controller sends a transmit request to the DMA until RXFIFO is empty before stopping the DMA transmission.</p>	1'b0

[20]	RW	<p>TXDMAEN</p> <p>Send DMA request enable</p> <p>bit 1'b0: do not enable</p> <p>send DMA request</p> <p>1'b1: Enable sending DMA requests</p> <p>NOTE: When the transmit DMA request is enabled and the number of words in TXFIFO is less than TXTH, the I2S controller sends a transmit request to the DMA until TXFIFO is full before stopping the DMA transmission.</p>	1'b0
[19]	WO	<p>RXCLR</p> <p>Empty RXFIFO</p>	1'b0

		<p>1'b0: not valid</p> <p>1'b1: Empty RXFIFO</p> <p>Note: Write 1 clears the RXFIFO, which is automatically cleared by hardware. Reading this bit always returns 0</p>	
[18]	WO	<p>TXCLR</p> <p>Empty TXFIFO</p> <p>1'b0: not valid</p> <p>1'b1: Empty TXFIFO</p> <p>Note: A write 1 clears the TXFIFO, which is automatically cleared by the hardware. Reading this bit always returns 0</p>	1'b0
[17]	RW	<p>LZCEN</p> <p>Left channel zero-crossing detection enable control bit</p> <p>1'b0: stop left channel zero-crossing detection</p> <p>1'b1: Enable left channel zero-crossing detection</p>	1'b0
[16]	RW	<p>RZCEN</p> <p>Right channel zero-crossing detection enable control bit</p> <p>1'b0: stop right channel zero-crossing detection</p> <p>1'b1: Enable right channel zero-crossing detection</p>	1'b0

[15]	RW	<p>Rx_clk_phase_sel</p> <p>receive clock</p> <p>phase selection</p> <p>1'b0: default</p> <p>mode</p> <p>Demonstrating the I2S bus</p> <p>timing mentioned above</p> <p>1'b1: Inversion Mode</p> <p>Shown as an inversion of the I2S bus timing mentioned above</p>	1'b0
------	----	---	------

[14:12]	RW	<p>RXTH</p> <p>RXFIFO</p> <p>Queue value</p> <p>3'b000: set que value to 0 characters</p> <p>3'b000: set que value to 1 word</p> <p>...</p> <p>3'b111: set the que value to 7 characters</p> <p>Note: The RXTHIF bit is set when the existing words in RXFIFO are equal to or more than the value of RXTH. The RXDMA or I2S interrupt can be triggered according to the setting.</p>	3'h4
[11:9]	RW	<p>TXTH</p> <p>TXFIFO Que</p> <p>3'b000: set que value to 0 characters</p> <p>3'b000: set que value to 1 word</p> <p>...</p> <p>3'b111: set the que value to 7 characters</p> <p>Note: The TXTHIF bit is set when the existing word in TXFIFO is equal to or less than the value of TXTH. The TXDMA or I2S interrupt can be triggered according to the setting.</p>	3'h4

[8]	RW	Tx_clk_phase_sel Select transmit clock phase mode 1'b0: default mode Demonstrating the I2S bus timing mentioned above 1'b1: Inversion Mode Shown as an inversion of the I2S bus timing mentioned above	1'b0
-----	----	---	------

[7:6]	RW	RSV	2'h0
[5:4]	RW	<p>WDWIDTH</p> <p>Transmit word</p> <p>length setting bit</p> <p>2'b00: word</p> <p>length 8 bit</p> <p>2'b01: word length 16 bit</p> <p>2'b10: word length 24 bit</p> <p>2'b11: word length 32 bit</p>	2'b0
[3]	RW	<p>MUTE</p> <p>Transmit Mute Enable Flag Bit</p> <p>1'b0: Transfer data from shift register, normal operation mode</p> <p>1'b1: Set transmission data to 0 to mute the sound</p>	1'b0
[2]	RW	<p>RXEN</p> <p>Receive enable flag bit</p> <p>1'b0: Stop I2S data reception</p> <p>1'b1: Enable I2S data reception</p>	1'b0
[1]	RW	<p>TXEN</p> <p>Transmission enable</p> <p>flag bit 1'b0: stops I2S data transmission</p> <p>1'b1: Enable I2S data transfer</p>	1'b0

[0]	RW	I2SEN I2S enable flag bit 1'b0: not enabled	1'b0
-----	----	--	------

		1'b1: enable	
--	--	--------------	--

16.9.3 Interrupt Mask Register

Table 132 I2S Interrupt Mask Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:10]	RO	RSV	22'h0
[9]	RW	<p>LZCIMASK</p> <p>Left channel zero cross</p> <p>interrupt enable flag bit</p> <p>1'b0: interrupt not enabled</p> <p>1'b1: interrupt enable</p> <p>Generates an interrupt when the interrupt is enabled and a zero-crossing is detected on the left channel</p>	1'b1
[8]	RW	<p>RZCIMASK</p> <p>Right channel zero cross</p> <p>interrupt enable flag bit</p> <p>1'b0: interrupt not enabled</p> <p>1'b1: interrupt enable</p> <p>Generates an interrupt when the interrupt is enabled and a zero-crossing is detected on the right channel</p>	1'b1

[7]	RW	TXDONEMASK Send completion interrupt enable bit 1'b0: interrupt not enabled 1'b1: interrupt enable Interrupt is generated when interrupt is enabled and TXFIFO is empty.	1'b1
[6]	RW	TXTHIMASK	1'b1

		<p>TXFIFO queue value</p> <p>interrupt enable bit</p> <p>1'b0: interrupt not</p> <p>enabled</p> <p>1'b1: interrupt enable</p> <p>Interrupt is generated when the interrupt is enabled and the number of data in TXFIFO is equal to or less than TXTH.</p>	
[5]	RW	<p>TXOVIMASK</p> <p>TXFIFO overflow</p> <p>interrupt enable bit</p> <p>1'b0: interrupt not</p> <p>enabled</p> <p>1'b1: interrupt enable</p> <p>Note: When interrupt is enabled, TXFIFO is full, and the CPU writes data to TXFIFO again, the TXOVIF flag bit will be set.</p>	1'b1
[4]	RW	<p>TXUDIMASK</p> <p>TXFIFO underflow</p> <p>interrupt enable bit</p> <p>1'b0: interrupt not</p> <p>enabled</p> <p>1'b1: interrupt enable</p> <p>Note: When the TXFIFO underflow interrupt is enabled and TXUDIF is detected as 1, an underflow interrupt is generated.</p>	1'b1

[3]	RW	<p>RXDONEMASK</p> <p>Receive completion</p> <p>interrupt enable flag bit</p> <p>1'b0: interrupt not enabled</p> <p>1'b1: interrupt enable</p> <p>The receive completion interrupt is generated when the receive completion interrupt is enabled and the receive process is complete</p>	1'b1
[2]	RW	<p>RXTHIMASK</p> <p>RXFIFO Que interrupt enable flag bit</p>	1'b1

		1'b0: interrupt not enabled 1'b1: interrupt enable When the RXFIFO queue interrupt is enabled and the number of data in RXFIFO is equal to or more than the number of queues, a raw RX interrupt is generated.	
[1]	RW	RXOVIMASK RXFIFO overflow interrupt enable bit 1'b0: interrupt not enabled 1'b1: interrupt enable Note: An overflow interrupt is generated when the RXFIFO outflow interrupt is enabled and TXOVIF is detected as a 1.	1'b1
[0]	RW	RXUDIMASK RXFIFO underflow interrupt enable bit 1'b0: interrupt not enabled 1'b1: interrupt enable Note: When the RXFIFO underflow interrupt is enabled and TXUDIF is detected as 1, an underflow interrupt is generated.	1'b1

16.9.4 Interrupt Flag Register

Table 133 I2S Interrupt Flag Registers

classifier for	intervi ews	Operating Instructions	reset value

honorific people			
[31:13]	RO	RSV	19'h0
[12]	RO	TXIF I2S transmit interrupt flag 1'b0: no I2S interrupt occurred	1'b0

		1'b1: I2S with transmit interrupt generation	
[11]	RO	RXIF I2S receive interrupt flag 1'b0: I2S interrupt not occurred 1'b1: I2S with receive interrupt generation	1'b0
[10]	RO	I2SIF I2S interrupt flag bit 1'b0: no I2S interrupt occurred 1'b1: I2S with interrupt generation Note: This bit is set whenever there is an interrupt on either RX or TX.	1'b0
[9]	RW	LZCIF Left channel zero-crossing detection sign This bit indicates that the next sample data symbol bit on the left channel has changed or that all data bits are zero. 1'b0: zero crossing not detected 1'b1: zero crossings detected Note: Write 1 to clear the interrupt flag	1'b0

[8]	RW	<p>RZCIF</p> <p>Right channel zero-crossing detection sign</p> <p>This bit indicates that the next sample data symbol bit on the right channel has changed or that all data bits are zero. 1'b0: zero crossing not detected</p> <p>1'b1: zero crossings detected</p> <p>Note: Write 1 to clear the interrupt flag</p>	1'b0
-----	----	---	------

[7]	RW	<p>TXDONEIF</p> <p>Send completion</p> <p>interrupt flag 1'b0:</p> <p>this send is not completed</p> <p>1'b1: This transmission is complete</p> <p>Note: Write 1 to clear the interrupt flag</p>	1'b0
[6]	RO	<p>TXTHIF</p> <p>RXFIFO Interrupt Flag</p> <p>1'b0: number of words in TXFIFO is greater than the queue value</p> <p>1'b1: The number of words in the TXFIFO is less than or equal to the queue value.</p> <p>Note: When the number of words in the TXFIFO (TXCNT) is equal to or less than the que value set by TXTH, this bit is set to 1 and will not change back to 0 until data is written to the TXFIFO and the value of TXCNT is greater than the value of TXTH.</p>	1'b0
[5]	RW	<p>TXOVIF</p> <p>TXFIFO overflow interrupt flag</p> <p>1'b0: no overflow interrupt occurred in TXFIFO</p> <p>1'b1: TXFIFO overflow interrupt occurred</p> <p>Note: Write 1 to clear the interrupt flag</p>	1'b0

[4]	RW	TXUDIF TXFIFO underflow interrupt flag 1'b0: no underflow interrupt occurred in TXFIFO 1'b1: TXFIFO underflow interrupt occurred Note: Write 1 to clear the interrupt flag	1'b0
-----	----	---	------

[3]	RW	<p>RXDONEIF</p> <p>Receive complete</p> <p>interrupt flag 1'b0:</p> <p>this receive is not completed</p> <p>1'b1: This reception is complete</p> <p>Note: Write 1 to clear the interrupt flag</p>	1'b0
[2]	RO	<p>RXTHIF</p> <p>RXFIFO Interrupt Flag</p> <p>1'b0: number of words in RXFIFO is less than the queue value</p> <p>1'b1: The number of words in the RXFIFO is equal to or greater than the queue value.</p> <p>Note: When the number of words in the RXFIFO is equal to or more than the queue value set by RXTH, this bit is set to 1. It does not change back to 0 until the data in the RXFIFO is read and the value of RXCNT is less than the value of RXTH.</p>	1'b0
[1]	RW	<p>RXOVIF</p> <p>RXFIFO overflow interrupt flag</p> <p>1'b0: no overflow interrupt occurred in RXFIFO</p> <p>1'b1: Overflow interrupt occurred in RXFIFO</p> <p>Note: Write 1 to clear the overflow interrupt</p>	1'b0

[0]	RW	<p>RXUDIF</p> <p>RXFIFO underflow interrupt</p> <p>flag 1'b0: no underflow interrupt</p> <p>occurred in RXFIFO</p> <p>1'b1: underflow interrupt occurred in RXFIFO</p> <p>Note: Write 1 to clear the underflow interrupt</p>	1'b0
-----	----	--	------

16.9.5 status register

Table 134 I2S Status Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:10]	RO	RSV	22'h0
[9:8]	RO	<p>VALIDBYTE</p> <p>Number of bytes available in the last word.</p> <p>2'b00: all bytes in the RXFIFO are available after reception is complete</p> <p>2'b01: 1 byte is available in RXFIFO after reception is completed</p> <p>2'b10: 2 bytes are available in RXFIFO after reception is complete</p> <p>2'b11: 3 bytes are available in the RXFIFO after reception is complete</p>	2'h0
[7:4]	RO	<p>TXCNT</p> <p>Records the number of words in the TXFIFO at the current moment. 4'b0000: no data</p> <p>4'b0001: with 1 character</p> <p>...</p> <p>4'b1000: with 8 characters</p>	4'h0

[3:0]	RO	RXCNT Records the number of words in the RXFIFO at the current moment. 4'b0000: no data 4'b0001: with 1 character ... 4'b1000: with 8 characters	4'h0
-------	----	--	------

16.9.6 Data Transmission Register

Table 135 I2S Data Transmission Registers

classifier for honori- fic people	interviews	Operating Instructions	reset value
[31:0]	WO	<p>TXFIFO</p> <p>The I2S has a built-in 8-word FIFO to store the data to be sent, and each time a word is written to the TXFIFO, one word is added to the TXFIFO. Each time a word is written to the TXFIFO, one more word is added to the TXFIFO, and the I2S controller automatically sends the first word to the TXFIFO.</p>	32'h0

16.9.7 Data Receive Register

Table 136 I2S Data Receive Registers

classifier for honori- fic people	interviews	Operating Instructions	reset value

[31:0]	RO	RXFIFO The I2S has a built-in 8 word length FIFO for storing received data. Each time a word is read from the RXFIFO, one word is missing from the RXFIFO.	32'h0
--------	----	---	-------

17 UART Module

17.1 Functional overview

UART is a universal serial data bus for asynchronous communication. The bus supports bi-directional communication and allows for full duplex transmission and reception.

W800 has 6 common UART ports, and through the fine clock frequency division combination, various baud rate settings can be realized, and it can support a maximum communication rate of 2Mbps.

W800 UART can be used with hardware DMA to realize efficient asynchronous data transmission.

17.2 Main characteristics

- Compliant with APB bus interface protocol, full duplex asynchronous communication method
- Supports interrupt or polling mode of operation
- Supports DMA Byte transfer mode, sending and receiving 32-byte FIFOs each.
- Baud rate programmable, supports up to 2Mbps
- 5-8bit data length and parity polarity configurable
- 1 or 2 stop bits configurable
- Supports RTS/CTS flow control
- Supports Break frame sending and receiving
- Supports Overrun, parity error, frame error, rx break frame interrupt indication.

17.3 Functional Description

17.3.1 UART Baud Rate

Asynchronous communication requires both sides to send and receive data according to the

negotiated baud rate because both sides do not have the same clock source for reference, and the W800 can realize fine baud rate control by using the baud rate setting register BAUD_RATE_CTRL register.

BAUD_RATE_CTRL[15:0] is named ubdiv, and BAUD_RATE_CTRL[19:16] is named ubdiv_frac, which needs to be set for wave
 The baud rate, calculated as follows:

```
ubdiv = apbclk / (16 * baudrate) - 1 // take
```

```
integer ubdiv_frac = (apbclk % (baudrate * 16)) / baudrate //
```

take integer

Using an APB clock of 40MHz and a baud

rate of 19200bps as example: ubdiv =

$40000000 / (16 * 19200) - 1 = 129$

ubdiv_frac = $(40000000 \% (19200 * 16)) / 19200 = 3$

Based on the above formula for an APB clock of 40MHz and a baud rate of 19200bps, the baud rate register should be set to: BAUD_RATE_CTRL = $(3 \ll 16) | 129 = 0x0003_0081$.

17.3.2 UART Data Format

- **data length**

The W800's UART supports configurable data lengths of 5bit, 6bit, 7bit, and 8bit. The definition of data length is as follows:

8bit single data length

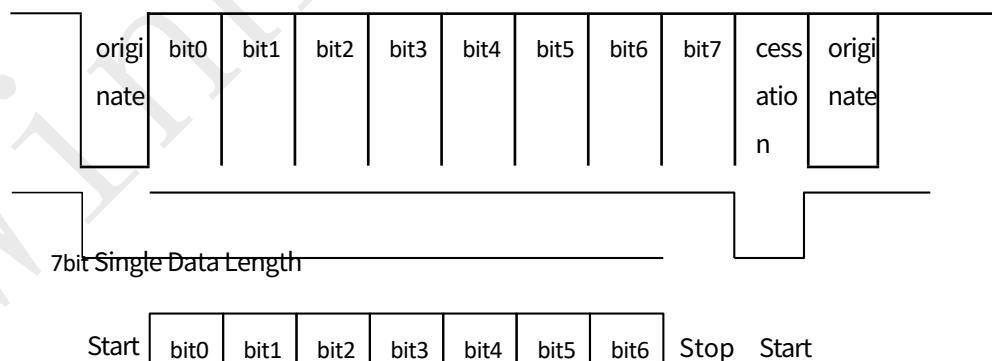


Figure 28 UART Data Length

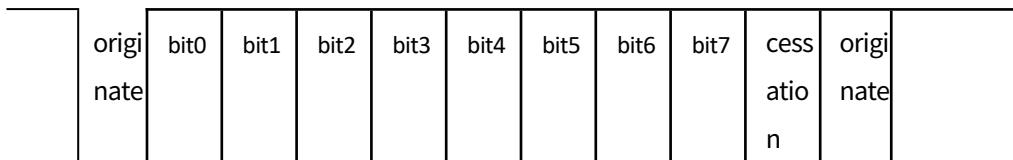
BAUD_RATE_CTRL[15:0] is named ubdiv, and BAUD_RATE_CTRL[19:16] is named ubdiv_frac,
~~Which deal with the data bit length information~~ consists of 1bit start bit, 1bit stop bit plus the middle data bit,
and the middle data bit is configurable. W800 supports 4 kinds of configurable data bit lengths,
5bit, 6bit, 7bit and 8bit, and you can select the data bit length according to the actual
application.

● stop bit

bit

The UART of W800 supports 1bit stop bit and 2bit stop bit which can be configured according to the actual needs as follows:

1bit Stop Bit



2bit Stop Bit

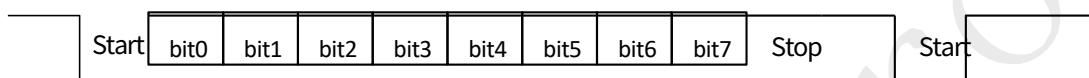


Figure 29 UART Stop Bits

● parity check digit

The parity bit serves to check the correctness of the data, and the W800 can set parity, even parity and no parity.

Calculation of odd parity: If the number of current data bits 1 is odd, the odd parity bit is 0. If the number of current data bits 1 is even, the odd parity bit is 1. In short, an odd 1 is guaranteed.

Even parity is calculated as follows: if the number of current data bits 1 is odd, the even parity bit is 1; if the number of current data bits 1 is even, the even parity bit is 0. In short, an even number of 1s is guaranteed.

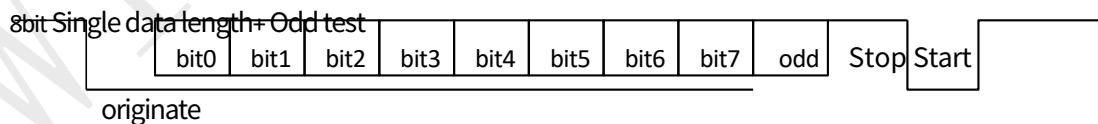


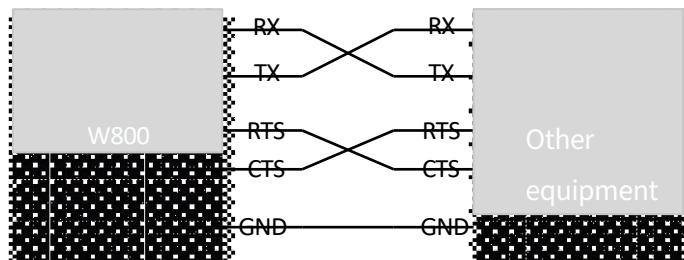
Figure 30 UART Parity Bits

17.3.3 UART

Hardware Flow

The W800 UART supports hardware flow control in the form of RTS/CTS. The main Control

purpose of the flow control is to prevent the UART fifo



The RTS and CTS are used correspondingly, as shown below:

Figure 31 UART Hardware Flow Control Connections

The hardware flow control of the W800 is controlled through the AUTO_FLOW_CTRL register. When the hardware flow control AUTO_FLOW_CTRL[0] is 1, W800 will set the flow control according to the number of data in the rxfifo set by AUTO_FLOW_CTRL[4:2], and if it is larger than the set number, the RTS will be pulled up and the other devices will no longer send data to W800, and if it is smaller than the set number, the RTS will be pulled down and the other devices will continue to send data to W800. When AUTO_FLOW is higher than the set number, RTS is pulled up and other devices stop sending data to W800. When AUTO_FLOW_CTRL[0] is 0, the software uses AUTO_FLOW_CTRL[1] to set the RTS level.

When W800 sends data, it can determine whether the current CTS has changed through interrupt and query the CTS status through FIFO_STATUS[12] to decide whether to continue to send data to other devices.

17.3.4 UART DMA Transfer

The UART of W800 supports DMA transfer mode, and for DMA transfer, you need to configure the DMA_CTRL register in the UART register list to turn on the DMA enable of the UART. At the same



time, you need to configure UART_FIFO_CTRL to configure how many bytes are left in txfifo and

Hardware Flow Control
txfifo to trigger DMA transfer.

The source or destination address of DMA is set to TX_DATA_WINDOW or RX_DATA_WINDOW, refer to the DMA Registers section for other DMA register settings.

Note: UART DMA transfer can only be set to Byte mode, half_word and word transfer modes are not supported.

17.3.5 UART interrupt

UART supports interrupt operation modes, including fifo null, fifo reaching the set trigger value, CTS change, error, etc. UART interrupt will be generated, and the desired interrupt can be set through the INT_MASK register.

When the UART interrupt is generated, you can query the current interrupt status and the reason for triggering the interrupt through INT_SRC. Software write 1 to clear 0.

17.4 register description

17.4.1 register list

Table 137 UART Register List

offset address	name (of a thing)	abridge	inter views	descriptive	reset value
0X0000	Data Flow Control Register	UART_LINE_CTRL	RW	Data format setting for uart communication	0X0000_000B
0X0004	Automatic Hardware Flow Control Register	AUTO_FLOW_CTRL	RW	uart rts/cts Hardware flow control settings	0X0000_0014
0X0008	DMA Setup Register	DMA_CTRL	RW	uart dma transfer mode setting	0X0000_0024
0X000C	FIFO Control Register	UART_FIFO_CTRL	RW	Setting the uart fifo trigger level	0X0000_0014

0X0010	Baud Rate Control Register	BAUD_RATE_CTRL	RW	Setting the uart communication baud rate	0X0003_0081
0X0014	Interrupt Mask Register	INT_MASK	RW	Setting the interrupts to be used by uart	0X0000_01FF
0X0018	Interrupt Status Register	INT_SRC	RW	uart Interrupt Status Indication	0X0000_0000

0X001C	FIFO Status Register	FIFO_STATUS	RW	fifo status, cts status query	0X0000_1000
0X0020	TX Starting address register	TX_DATA_WINDOW	WO		0X0000_0000
0X0024	reservations				
0X0028	reservations				
0X002C	reservations				
0X0030	RX Starting address register	RX_DATA_WINDOW	RO		0X0000_0000
0X0034	reservations				
0X0038	reservations				
0X003C	reservations				

17.4.2 Data Flow Control Register

Table 138 UART Data Flow Control Registers

classifier for honori- fic people	intervi- ews	Operating Instructions	reset value
[31: 8]		reservations	
[7]	RW	uart_rx_enable Receive enable, active high	1'b0
[6]	RW	uart_tx_enable Transmit enable, active high.	1'b0

[5]	RW	send break enable The Uart will send a break packet after this bit is set, and clear 0 automatically when finished.	1'b0
[4]	RW	parity polarity	1'b0

		1'b0: even checking 1'b1: odd checksums	
[3]	RW	parity en Parity Check Enable, High Valid	1'b1
[2]	RW	number of stop bits 1'b0: 1 stop bit 1'b1: 2 stop bits	1'b0
[1 : 0]	RW	uart Bit length. 2'h0: 5bit 2'h1: 6bit 2'h2: 7bit 2'h3: 8bit	2'h3

17.4.3 Automatic Hardware Flow Control Register

Table 139 UART Automatic Hardware Flow Control Registers

classi fier for hono rific peop le	intervi ews	Operating Instructions	reset value
[31: 5]		reservations	

[4 : 2]	RW	RTS trigger level Determines when RTS needs to be disabled when afc_enable is active. 3'h0: rxfifo with more than 4 bytes 3'h1: rxfifo with more than 8 bytes	3'h5
---------	----	---	------

		3'h2: rxfifo has more than 12 bytes 3'h3: rxfifo has more than 16 bytes 3'h4: rxfifo has more than 20 bytes 3'h5: rxfifo has more than 24 bytes 3'h6: rxfifo has more than 28 bytes 3'h7: rxfifo has more than 31 bytes	
[1]	RW	RTS set When AFC_enable is not valid, software can accomplish receive flow control by setting this bit. When AFC_enable is valid, this bit does not care.	1'b0
[0]	RW	afc enable 1'b1: Valid, receive condition rts is generated using the rts_trigger_level control.	1'b0

17.4.4 DMA Setup Register

Table 140 UART DMA Setup Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:8]		reservations	

[7 : 3]	RW	<p>rxfifo timeout num</p> <p>The rxfifo timeout interrupt is generated if no new data is received within N packets if there is less than rxfifo_trigger_level data in rxfifo.</p> <p>When this timing function is enabled, both the first timing and the last timing completion are only received when at least 1</p> <p>The timer starts only after one packet.</p>	5'h04
[2]	RW	rxfifo timeout en	1'b1

		rxfifo timeout enable	
[1]	RW	rx dma enable Receive DMA enable, active high. 0: Indicates that the receive process uses an interrupt.	1'b0
[0]	RW	tx dma enable Send DMA enable, active high. 0: Indicates that the send process uses an interrupt.	1'b0

17.4.5 FIFO Control Register

Table 141 UART FIFO Control Registers

classi fier for hono rific peop le	intervi ews	Operating Instructions	reset value
[31: 6]		reservations	

[5 : 4]	RW	<p>rxfifo trigger level</p> <p>Triggers an interrupt when the number of data bytes in rxfifo is greater than or equal to this value, or triggers rxdma req. 2'h0: 1byte 2'h1: 4byte 2'h2: 8byte 2'h3: 16byte</p>	2'h1
[3 : 2]	RW	<p>txfifo trigger level</p> <p>Triggers an interrupt when the number of data bytes in txfifo is less than or equal to this value, or triggers a txdma req. 2'h0: empty 2'h1: 4byte</p>	2'h1

		2'h2: 8byte 2'h3: 16byte	
[1]	RW	rxfifo reset Reset rxfifo to clear the rxfifo state	1'b0
[0]	RW	txfifo reset Reset txfifo to clear the txfifo state	1'b0

17.4.6 Baud Rate Control Register

Table 142 UART Baud Rate Control Registers

classifier for honori- fic people	intervi- ews	Operating Instructions	reset value
[31:20]		reservations	
[19:16]	RW	<p>ubdiv_frac</p> <p>Indicates the decimal portion of the system clock divided by the quotient of the 16x baud rate clock. The exact value is $\text{frac} \times 16$.</p> <p>(Refer to Section 2.3.2, Baud Rate Calculation Method)</p>	4'h3
[15:0]	RW	<p>ubdiv</p> <p>The system clock is divided by the integer part of the 16 times baud rate clock quotient minus 1. The default system clock is 40 MHz and the baud rate is 19200.</p> <p>(Refer to Section 2.3.2, Baud Rate Calculation Method)</p>	16'h81

17.4.7 Interrupt Mask Register

Table 143 UART Interrupt Mask Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value

[31: 9]		reservations	
[8]	RW	overrun error int mask, rxfifo Overrun interrupt mask bit, valid high.	1'b1
[7]	RW	parity error int mask, parity check interrupt mask bit, valid high.	1'b1
[6]	RW	frame error int mask, data frame error interrupt mask bit, valid high.	1'b1
[5]	RW	break detect int mask, break signal detect interrupt mask bit, valid high.	1'b1
[4]	RW	cts changed indicate mask, CTS signal changed interrupt mask bit, valid high.	1'b1
[3]	RW	rxfifo data timeout int mask, rxfifo receive data timeout interrupt mask bit, valid high.	1'b1
[2]	RW	rxfifo trigger level int mask, rxfifo reach trigger value interrupt mask bit, valid high.	1'b1
[1]	RW	txfifo trigger level int mask, txfifo reach trigger value interrupt mask bit, valid high.	1'b1
[0]	RW	txfifo empty int mask, txfifo is the air break mask bit, valid high.	1'b1

17.4.8 Interrupt Status Register

Table 144 UART Interrupt Status Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31: 9]		reservations	
[8]	RW	overrun error rxfifo overflowed. Software-initiated write 1 to clear 0.	1'b0

[7]	RW	<p>parity error</p> <p>Received packet checksum bit error.</p> <p>In the DMA case, this interrupt is still generated. However,</p> <p>DMA operation does not care about this interrupt.</p> <p>Software-initiated write 1 clears 0.</p>	1'b0
-----	----	---	------

[6]	RW	<p>frame error</p> <p>Received packet stop bit error.</p> <p>In the DMA case, this interrupt is still generated. However,</p> <p>DMA operation does not care about this interrupt.</p> <p>Software-initiated write 1 clears 0.</p>	1'b0
[5]	RW	<p>break detect</p> <p>A break packet was received.</p> <p>In the DMA case, this interrupt is still generated. However,</p> <p>DMA operation does not care about this interrupt.</p> <p>Software-initiated write 1 clears 0.</p>	1'b0
[4]	RW	<p>cts changed</p> <p>This interrupt is generated</p> <p>when the cts signal changes.</p> <p>Software-initiated write 1</p> <p>clears 0.</p>	1'b0
[3]	RW	<p>rxfifo data timeout</p> <p>An interrupt is generated if the data length in rxfifo is less than the rxfifo trigger level but no data is received for N data cycles.</p> <p>The software actively writes 1 clear 0.</p>	1'b0

[2]	RW	<p>rxfifo trigger level interrupt</p> <p>This interrupt is generated when the number of data in rxfifo changes from less than the number specified in rxfifo trigger level to greater than or equal to that number.</p> <p>The current data frame size should be determined from rxfifo count.</p> <p>The software actively writes 1 clear 0.</p>	1'b0
[1]	RW	<p>txfifo trigger level interrupt</p> <p>When the number of data in the txfifo changes from greater than the number specified in the txfifo trigger level to less than or equal to that number</p>	1'b0

		When an interrupt is generated. The software actively writes 1 clear 0.	
[0]	RW	tx fifo empty interrupt This interrupt is generated when the current packet has been sent and txfifo is empty. Software-initiated write 1 to clear 0.	1'b0

17.4.9 FIFO Status Register

Table 145 UART FIFO Status Registers

classifier for honori c people	intervi ews	Operating Instructions	reset value
[31:13]		reservations	
[12]	RW	cts status Current cts status	1'b0
[11: 6]	RW	rxfifo count Number of data in rxfifo	6'h0
[5 : 0]	RW	txfifo count Number of data in txfifo	6'h0

address registers

Table 146 UART TX start address registers

classifier	intervi	Operating Instructions	reset value
------------	---------	------------------------	-------------

for honorifi c people	ews		
[31:0]	WO	tx data window Send data start address.	32'h0

		<p>Note: uart sends and receives data in bytes only, when burst transmission is used, it is possible to use words to send and receive data.</p> <p>The section address incremental method is designed to support up to 16-burst operation, i.e., 16 bytes, so a total of 16 bytes (4 words) from the transmit/receive start address are reserved for the transmit/receive data window.</p>	
--	--	--	--

17.4.11RX Starting address registers

Table 147 UART RX start address registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:0]	RO	rx data window Receive data start address. Note: uart send and receive data only support byte operation, when burst transmission is used, it is possible to use byte address incremental method, the design supports up to 16-burst operation, i.e., 16byte. therefore, a total of 16byte (4 words) from the send/receive start address are reserved for the send/receive data window.	32'h0

18 UART&7816 Module

18.1 Functional overview

The UART&7816 modules are compatible with UART functionality, as well as 7816 interface functionality.

W800 supports 1 group UART&7816 multiplexing interface (uart2), when used as UART, it can realize various baud rate settings through fine clock division frequency combination, and can support maximum communication rate of 2Mbps.

The W800 UART&7816 interface can be used with hardware DMA for efficient data transfer.

18.2 Main characteristics

- Conforms to the APB bus interface protocol and supports UART asynchronous full-duplex and 7816 asynchronous half-duplex communication;
- Supports interrupt or polling mode of operation;
- Supports DMA Byte transfer mode with 32-byte FIFO for transmit and receive;
- Serial port function:
 - Baud rate programmable, supports up to 2Mbps
 - 5-8bit data length and parity polarity configurable
 - 1 or 2 stop bits configurable
 - Supports RTS/CTS flow control
 - Supports Break frame sending and receiving
 - Supports Overrun, parity error, frame error, rx break frame interrupt indication.
- 7816 Interface Functions:

-
- Compatible with ISO-7816-3 T=0.T=1 mode

- Compatible with EVM2000 protocol
- Configurable guard time (11 ETU-267 ETU)
- Forward/reverse convention, software configurable
- Support send/receive parity and retransmission function
- Supports 0.5 and 1.5 stop bits configurable

18.3 UART Function Description

Refer to Chapter 16, UART Function Module Description

18.4 7816 Functional Description

18.4.1 7816 Introduction

ISO7816 is an international standard smart card protocol that specifies the physical characteristics, dimensions and interfaces, electrical signals and transmission protocols, commands, security and other aspects of smart card information.

W800 mainly realizes the part of ISO 7816-3 telecommunication signals and transmission protocols, and supports T0 and T1 cards. Through the 7816 interface of W800, users can directly interact with the smart card for data and commands without caring about the signal communication logic of clock and data. Regarding the smart card data and command interaction, users need to refer to the ISO7816-4 protocol to realize it by themselves.

18.4.2 7816 Interface

W800 mainly integrates the clock and data interfaces of smart card to realize the communication signal logic of data and command. In actual smart card applications, there are also three signals,

RST, VCC and GND. RST can be controlled by ordinary GPIOs, mainly used when the smart card is powered on and reset, while VCC can be directly connected to the 3.3V power supply, or through the GPIOs with other circuits to control the smart card VCC on and off.

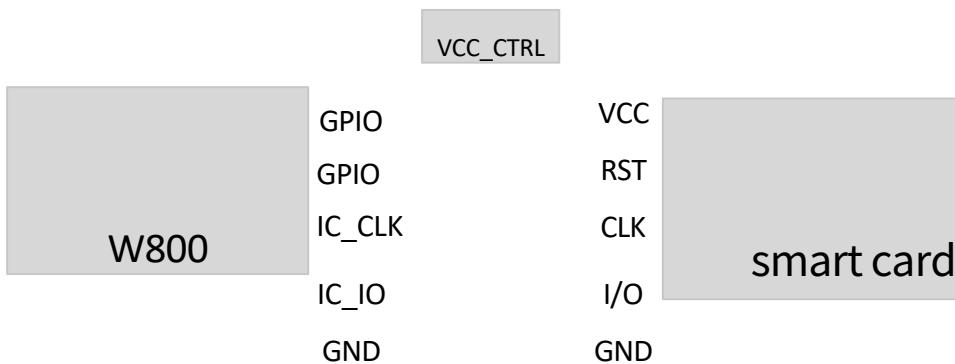


Figure 32 7816 Connection Schematic

18.4.3 7816 Configuration

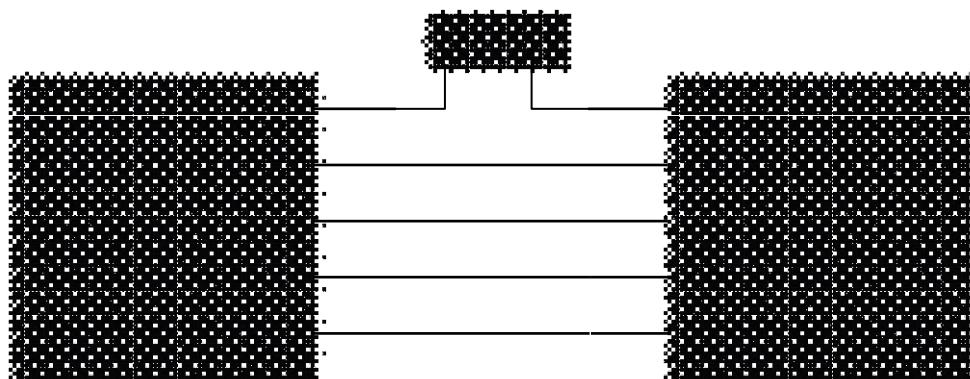
Configuration is required for use as a 7816 interface:

- Set the interface working mode, UART_LIN_CTRL[24] is set to 1 to select the current interface as 7816 mode;
- Set 7816 MSB or LSB transmission mode, UART_LIN_CTRL[3] is set to 1. Select whether the 7816 interface is in MSB mode (bit7 is transmitted first) or LSB mode (bit0 is transmitted first) by UART_LIN_CTRL[3];
- Set stop bit, UART_LIN_CTRL[2] can select smart card 0.5 or 1.5 stop bits;
- Select the card type, UART_LIN_CTRL[8] can select T0 card or T1 card;
- Configure the smart card communication timeout time, through the WAIT_TIME to set the timeout time, receive data timeout not received data will generate a timeout interrupt.

18.4.4 7816 Clock Configuration

The smart card clock is the clock provided to the smart card via the CLK pin and is set via BAUD_RATE_CTRL[21:16], calculated as follows:

$$= \frac{f_{clk_apb}}{1} - 1$$



fclk_apb: System APB bus clock;

clk_div: clock divider to be set for BAUD_RATE_CTRL[21:16].

Because clk_div can only take the integer, in order to reduce the error, we had better take the rounding calculation method, C language calculation will lose the decimal part of the calculation, C language to take the rounding conversion method is as follows:

$$\text{clk_div} = (\text{fclk_apb} + \text{fsc_clk}) / (2 * \text{fsc_clk}) - 1;$$

18.4.5 7816 Rate Setting

The smart card has a time unit, ETU, according to which the smart card transmits data and commands. the ETU is set by BAUD_RATE_CTRL[15:0] and is calculated as follows:

$$= \frac{F}{D} \times \frac{1}{f}$$

f: i.e. the CLK of our smart card;

F and D are both parameters given with smart cards.

In fact, the ubdiv of BAUD_RATE_CTRL[15:0] we need to set is F/D, the above formula is just to calculate the ETU for your reference. The above formula is just to calculate the ETU for your reference, when we actually set it, we only need to set ubdiv = F/D. F and D can be queried

Bits 8 to 5	0000	0001	0010	0011	0100	0101	0110	0111
from the following table.								
f(max.) MHz	4	5	6	8	12	16	20	—
Bits 8 to 5	1000	1001	1010	1011	1100	1101	1110	1111
fi	RFU	512	768	1024	1536	2048	RFU	RFU
f(max.) MHz	—	5	7.5	10	15	20	—	—

— According to Table 8, bits 4 to 1 encode Di.

Table 8 — Di

Bits 4 to 1	0000	0001	0010	0011	0100	0101	0110	0111
Di	RFU	1	2	4	8	16	32	64
Bits 4 to 1	1000	1001	1010	1011	1100	1101	1110	1111
Di	12	20	RFU	RFU	RFU	RFU	RFU	RFU

18.4.6 7816 Power-on reset

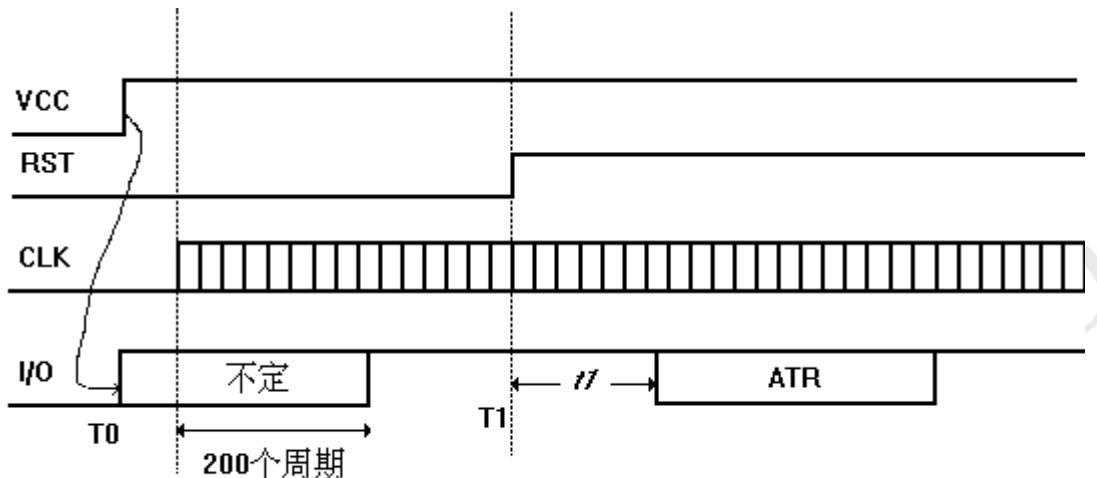


Figure 33 7816 Power-On Reset Timing

The above figure shows the smart card power-on reset timing, CLK and IO are initially low, we need to configure them to GPIO mode and pull them low, VCC is pulled high, CLK and IO are configured to 7816 mode and controlled by 7816. Finally, we need to pull the RST pin high manually to complete the reset process. The configuration steps are as follows:

- I/O, CLK, and RST are configured to normal GPIO mode and held low;
- Set 7816 to T=0 mode;
- Controls VCC power-up via GPIO;
- Configure I/O, CLK to 7816 mode, with the 7816 driving the clock and data;
- Configure the 7816 clock frequency and allow clock output;
- Set the RST pin and wait for receiving ATR data, if no ATR data is received within 40000 clocks, the deactivation process is executed and the card is deactivated.

18.4.7 7816 Thermal Reset

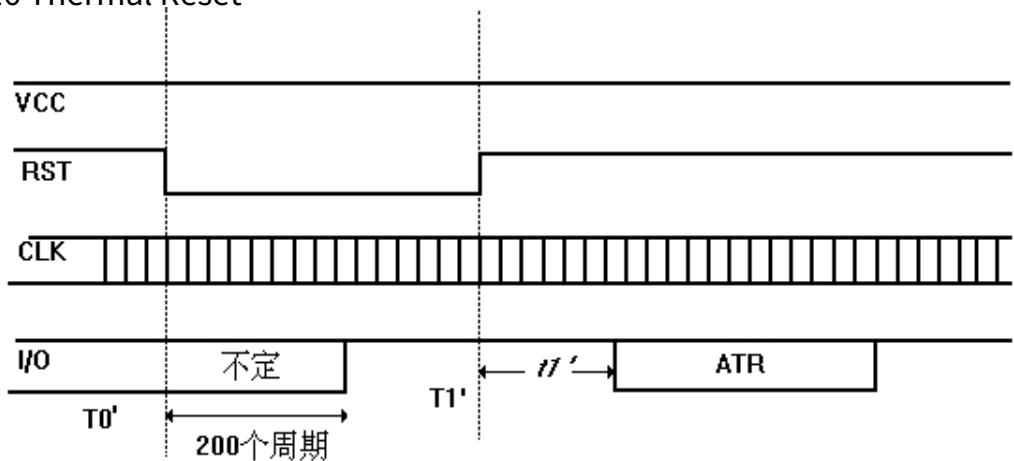


Figure 34 7816 Thermal Reset

As shown in the above figure, the procedure for a hot reset is simple, in normal operating mode, pulling the RST pin low for 400 cycles is sufficient. The configuration steps are as follows:

- Keeps VCC powered up;
- Pull down the RST pin for at least 400 clock cycles;
- Pull up the RST pin and wait for receiving ATR data, if ATR data is not received within 40000 clocks, the deactivation process will be executed and the card will be deactivated.

18.4.8 7816 Inactivation process

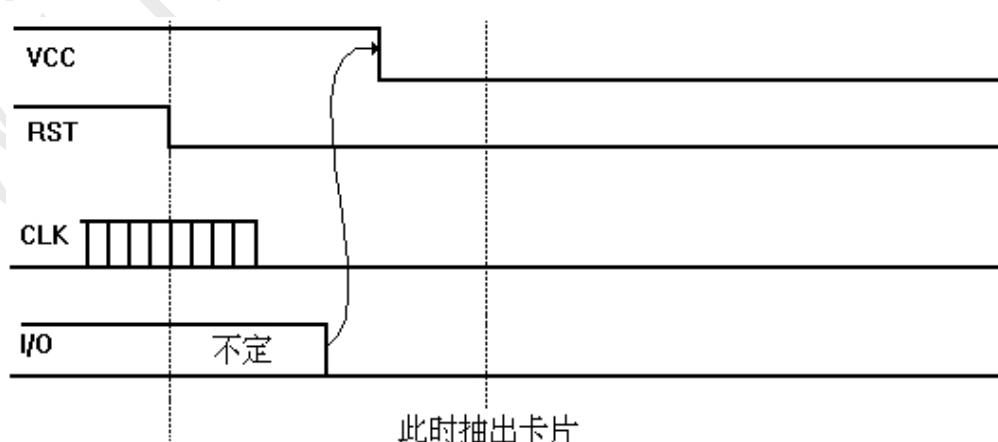


Figure 35 7816 Deactivation process

As shown in the above figure, after RST is pulled down, you need to configure CLK and IO into normal IO mode and pull them down, and finally turn off the VCC power supply.
 Below:

- Keeps VCC powered up;
- Pull down the RST pin;
- Configure CLK and IO to GPIO mode and pull low;
- Controls VCC power down via GPIO;

18.4.9 7816 Data transmission

The timing of data transmission of 7816 has been completed by W800 hardware, no user operation is required, if users want to know the specific content of this part, please refer to the provisions in the ISO7816-3 protocol.

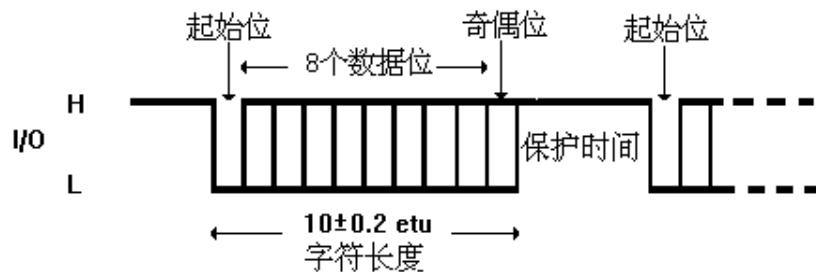


Figure 36 7816 Data Transfer

18.4.10 UART&7816 DMA Transmission

The UART&7816 of W800 supports DMA transfer mode, and for DMA transfer, you need to configure the DMA_CTRL register in the UART&7816 register list to turn on the DMA enable of the UART. At the same time, you need to configure UART_FIFO_CTRL to configure how many bytes are left in txfifo and rxfifo to trigger DMA transfer.

As shown in the above figure, after RST is pulled down, you need to configure CLK and IO into ~~the bus~~ mode and set the address of the DMA, set off the ~~VACT~~ power supply or RX_DATA_WIDNOW, and the other DMA registers are set as follows

Exam DMA Registers chapter.

Note: UART&7816 DMA transfer can only be set to Byte mode, half_word and word transfer modes are not supported.

18.4.11 UART&7816 Interrupt

UART&7816 supports interrupt operation modes, including fifo null, fifo reaching the set trigger value, CTS change, error, etc. will generate UART&7816 interrupt, and you can set the desired interrupt through INT_MASK register.

When the UART&7816 interrupt is generated, the current interrupt status can be queried by INT_SRC, the reason for triggering the interrupt. Software write 1 to clear 0.

18.5 register description

18.5.1 register list

Table 149 UART&7816 Register List

offset address	name (of a thing)	abridge	inter views	descriptive	reset value
0X0000	Data Flow Control Register	UART_LINE_CTRL	RW	Data-related settings for uart&7816 communication install	0X0033_520B
0X0004	Automatic Hardware Flow Control Register	AUTO_FLOW_CTRL	RW	uart rts/cts Hardware flow control settings	0X0000_0014
0X0008	DMA Setup Register	DMA_CTRL	RW	uart&7816 dma transfer mode setting	0X0000_0024
0X000C	FIFO Control Register	UART_FIFO_CTRL	RW	Setting the uart&7816 fifo trigger level	0X0000_0014

0X0010	Baud Rate Control Register	BAUD_RATE_CTRL	RW	Setting the uart baud rate and 7816 clock	0X0003_0082
0X0014	Interrupt Mask Register	INT_MASK	RW	Setting up the uart&7816 to be used in the decidedly	0X0000_03FF

0X0018	Interrupt Status Register	INT_SRC	RW	uart&7816 Interrupt status indication	0X0000_0000
0X001C	FIFO Status Register	FIFO_STATUS	RW	fifo status, cts status query	0X0000_0000
0X0020	TX Starting address register	TX_DATA_WINDOW	WO		0X0000_0000
0X0024	reservations				
0X0028	reservations				
0X002C	reservations				
0X0030	RX Starting address register	RX_DATA_WINDOW	RO		0X0000_0000
0X0034	reservations				
0X0038	reservations				
0X003C	reservations				
0X0040	7816 Protection Time Register	GUARD_TIME	RW	7816 Inter-data protection time	0X0000_0000
0X0044	7816 Timeout Time Register	WAIT_TIME	RW	7816 Receive Data Timeout	0X0007_8000

18.5.2 Data Flow Control Register

Table 150 UART&7816 Data Flow Control Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:25]		reservations	

[24]	RW	sc_mode 1'b0: uart mode 1'b1: 7816 Mode	1'b0
[23]	RW	7816 Card T0 Mode rx_retrans_en 1'b0: Rx auto retransmission not working	1'b0

		1'b1: Rx auto retransmit enable	
[22:20]	RW	7816 Card T0 Mode rx_retrans_cnt	3'h3
[19]	RW	<p>7816 Card T0 Mode</p> <p>tx_retrans_en 1'b0: Tx auto-retransmission not valid</p> <p>1'b1: Tx auto retransmission enable</p>	1'b0
[18:16]	RW	<p>7816 card t0 mode tx_retrans_cnt</p> <p>tx Number of automatic retransmissions</p>	3'h3
[15:11]	RW	<p>7816 Card Minimum MIN_BGT (Min Block Guard Time)</p> <p>Min Block Guard Time Calculation: 10+stop bits (default 2 bits) + configured value MIN_BGT Note.</p> <p>T=0: The minimum time interval between the falling edge of the start bit of two consecutive characters sent and received in opposite directions must not be less than 16 ETUs. it must be possible to correctly interpret a received character whose falling edge of the start bit is 15 ETUs from the falling edge of the start bit of the last byte transmitted.</p> <p>T=1: Minimum time interval between the falling edges of the start bits of two consecutive characters sent and received in opposite directions (Block Guard Time, BGT) must be 22 ETUs. must be able to correctly interpret characters received within 21 ETUs of the interval between the falling edge of their start bit and the falling edge of the start bit of the last byte sent.</p>	5'ha
[10]	RW	<p>7816 Card Clock Control Configuration</p> <p>1'b0: card clock output is generated when configured for card mode, otherwise card clock output is invalidated</p> <p>1'b1: Clock stop</p>	1'b0

[9]	RW	7816 Whether to receive data in case of card parity error 1'b0: not received 1'b1: Receiving	1'b1
-----	----	--	------

[8]	RW	7816 card's T0/T1 mode configuration. 1'b0: T0 mode 1'b1: T1 mode	1'b0
[7]	RW	uart_rx_enable uart/7816 mode, receive enable, active high	1'b0
[6]	RW	uart_tx_enable Transmit enable in uart/7816 mode, active high.	1'b0
[5]	RW	send break enable The Uart will send a break packet after this bit is set, and clear 0 automatically when finished.	1'b0
[4]	RW	parity polarity (UART mode) 1'b0: even parity 1'b1: odd checksums Forward and reverse (7816 mode) 1'b0: LSB (b0 bit) transmitted first 1'b1: MSB (b7 bit) transmitted first	1'b0
[3]	RW	parity enable, active high (UART mode)	1'b1
[2]	RW	Number of stop bits (UART mode) 1'b0: 1 stop bit 1'b1: 2 stop bits Number of stop bits (7816 mode) 1'b0: 0.5 stop bits	1'b0

		1'b1: 1.5 stop bits	
[1 : 0]	RW	uart bit length (UART mode) 2'h0: 5bit 2'h1: 6bit 2'h2: 7bit 2'h3: 8bit	2'h3

18.5.3 Automatic Hardware Flow Control Register

Table 151 UART&7816 Automatic Hardware Flow Control Registers

classi fier for hono rific peop le	intervi ews	Operating Instructions	reset value
[31: 5]		reservations	

[4 : 2]	RW	<p>RTS trigger level (UART mode)</p> <p>Determines when RTS needs to be disabled when afc_enable is active.</p> <p>3'h0: rxfifo with more than 4 bytes</p> <p>3'h1: rxfifo has more than 8 bytes</p> <p>3'h2: rxfifo has more than 12 bytes</p> <p>3'h3: rxfifo has more than 16 bytes</p> <p>3'h4: rxfifo has more than 20 bytes</p> <p>3'h5: rxfifo has more than 24 bytes</p> <p>3'h6: rxfifo has more than 28 bytes</p> <p>'h5: rxfifo has more than 24 bytes</p> <p>3'h6: rxfifo has more than 28 bytes</p> <p>3'h7: rxfifo has more than 31 bytes</p>	3'h5
---------	----	---	------

[1]	RW	RTS set (UART mode) When AFC_enable is not valid, software can accomplish receive flow control by setting this bit. When AFC_enable is valid, this bit does not care.	1'b0
[0]	RW	afc enable (UART mode) Receive condition rts Generated using the rts_trigger_level control, valid high.	1'b0

18.5.4 DMA Setup Register

Table 152 UART&7816 DMA Setting Registers

classifier for honori- fication people	intervi- ews	Operating Instructions	reset value
[31: 8]		reservations	
[7 : 3]	RW	rxfifo timeout num (UART mode) The rxfifo timeout interrupt is generated if no new data is received within N packets if there is less than rxfifo_trigger_level data in rxfifo. When this timing function is enabled, both the first timing and the last timing completion are only received when at least 1 The timer starts only after one packet.	5'h4
[2]	RW	rxfifo timeout en (UART&7816 mode) rxfifo timeout enable, active high	1'b1
[1]	RW	rx dma enable (UART&7816 mode) Receive DMA enable, active high. 0 Indicates that the receive process uses an interrupt.	1'b0

[0]	RW	tx dma enable (UART&7816 mode) Transmit DMA Enable, active high.	1'b0
-----	----	---	------

		0 Indicates that an interrupt is used for the sending process.	
--	--	--	--

18.5.5 FIFO Control Register

Table 153 UART&7816 FIFO Control Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31: 6]		reservations	
[5 : 4]	RW	rxfifo trigger level(UART&7816 mode) Triggers an interrupt when the number of data bytes in rxfifo is greater than or equal to this value, or triggers rxdma req. 2'h0: 1byte 2'h1: 4byte 2'h2: 8byte 2'h3: 16byte	2'h1
[3 : 2]	RW	txfifo trigger level(UART&7816 mode) Triggers an interrupt when the number of data bytes in txfifo is less than or equal to this value, or triggers a txdma req. 2'h0: empty 2'h1: 4byte 2'h2: 8byte 2'h3: 16byte	2'h1
[1]	RW	rxfifo reset(UART&7816 mode) Reset rxfifo, clear the rxfifo state.	1'b0

[0]	RW	txfifo reset(UART&7816 mode) Reset txfifo to clear the txfifo state	1'b0
-----	----	--	------

18.5.6 Baud Rate Control Register

Table 154 UART&7816 Baud Rate Control Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:20]		reservations	
[19:16]	RW	<p>ubdiv_frac</p> <p>UART mode:</p> <p>Indicates the decimal portion of the system clock divided by the quotient of the 16x baud rate clock. The exact value is $\text{frac} \times 16$.</p> <p>(Refer to section Baud Rate Calculation Methods) 7816</p> <p>Mode:</p> <p>$\text{ubdiv_frac} = (\text{fclk_apb} + \text{fsc_clk}) / (2 * \text{fsc_clk}) - 1;$</p> <p>(refer to 7816 clock calculations)</p>	4'h3

[15:0]	RW	<p>ubdiv</p> <p>UART mode:</p> <p>The system clock is divided by the integer part of the 16 times baud rate clock quotient minus 1. The default system clock is 40 MHz and the baud rate is 19200.</p> <p>(Refer to Baud Rate Calculation Method)</p> <p>7816 Mode:</p> <p>ubdiv=Fi/Di (Fi, Di are parameters for smart card feedback,edu frequency: $f_{etuclk} = f_{sc_clk}/(ubdiv+1)$)</p> <p>(Refer to Section 7816 Rate Calculation Methods)</p>	16'h82
--------	----	---	--------

18.5.7 Interrupt Mask Register

Table 155 UART&7816 Interrupt Mask Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:10]		reservations	
[9]	RW	7816 Card received error signal errorsignalwhile transmitting.(7816 mode)	1'b1
[8]	RW	overrun error int mask, rxfifo Overrun interrupt mask bit, valid high. (UART&7816 mode)	1'b1
[7]	RW	parity error int mask, parity check interrupt mask bit, valid high. (UART&7816 mode)	1'b1
[6]	RW	frame error int mask, data frame error interrupt mask bit, valid high. (UART mode)	1'b1
[5]	RW	break detect int mask, break signal detect interrupt mask bit, valid high. (UART mode)	1'b1
[4]	RW	cts changed indicate mask, CTS signal changed interrupt mask bit, valid high. (UART mode)	1'b1
[3]	RW	rxfifo data timeout int mask, rxfifo receive data timeout interrupt mask bit, valid high. (UART&7816 (Mode))	1'b1
[2]	RW	rxfifo trigger level int mask, rxfifo reach trigger value interrupt mask bit, valid high. (UART&7816 Module) style)	1'b1
[1]	RW	txfifo trigger level int mask, txfifo reach trigger value interrupt mask bit, valid high. (UART&7816 module) style)	1'b1
[0]	RW	txfifo empty int mask, txfifo is the air break mask bit, valid high. (UART&7816 mode)	1'b1

18.5.8 Interrupt Status Register

Table 156 UART&7816 Interrupt Status Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31: 9]		reservations	

[9]	RW	7816 The card received an error signal while transmitting. (7816 mode)	
[8]	RW	<p>overrun error (UART&7816 mode)</p> <p>rxfifo overflowed.</p> <p>The software actively writes 1 clear 0.</p>	1'b0
[7]	RW	<p>parity error (UART&7816 mode)</p> <p>Received packet parity bit error.</p> <p>This interrupt is still generated in the DMA case. However, the DMA operation does not care about this interrupt.</p> <p>The software actively writes 1 clear 0.</p>	1'b0
[6]	RW	<p>frame error (UART mode)</p> <p>Received packet stop bit error.</p> <p>In the DMA case, this interrupt is still generated. However, the DMA operation does not care about this interrupt.</p> <p>The software actively writes 1 clear 0.</p>	1'b0
[5]	RW	<p>break detect (UART mode)</p> <p>A break packet is received.</p> <p>In the DMA case, this interrupt is still generated. However, the DMA operation does not care about this interrupt.</p> <p>The software actively writes 1 clear 0.</p>	1'b0
[4]	RW	<p>cts changed (UART mode)</p> <p>This interrupt is generated when the cts signal changes.</p>	1'b0

		The software actively writes 1 clear 0.	
[3]	RW	<p>rxfifo data timeout (UART&7816 mode)</p> <p>An interrupt is generated if the data length in rx FIFO is less than the rx FIFO trigger level but no data is received for N data cycles.</p>	1'b0

		The software actively writes 1 clear 0.	
[2]	RW	<p>rxfifo trigger level interrupt (UART&7816 mode)</p> <p>This interrupt is generated when the number of data in rxfifo changes from less than the number specified in rxfifo trigger level to greater than or equal to that number.</p> <p>The current data frame size should be determined from rxfifo count.</p> <p>The software actively writes 1 clear 0.</p>	1'b0
[1]	RW	<p>txfifo trigger level interrupt (UART&7816 mode)</p> <p>An interrupt is generated when the number of data in txfifo changes from greater than the number specified in txfifo trigger level to less than or equal to that number.</p> <p>The software actively writes 1 clear 0.</p>	1'b0
[0]	RW	<p>tx fifo empty interrupt (UART&7816 mode)</p> <p>This interrupt is generated when the current packet has been sent and txfifo is empty. Software-initiated write 1 to clear 0.</p>	1'b0

18.5.9 FIFO Status Register

Table 157 UART&7816 FIFO Status Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value

[31:13]		reservations	
[12]	RW	cts status (UART mode) Current cts status	1'b0

[11: 6]	RW	rxfifo count (UART&7816 mode) Number of data in rxfifo	6'h0
[5 : 0]	RW	txfifo count (UART&7816 mode) Number of data in txfifo	6'h0

18.5.10 TX Starting address register

Table 158 UART&7816 TX start address registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:0]	WO	tx data window (UART&7816 mode) Transmit data start address. Note: uart sends and receives data only supports byte operation, when burst transmission is used, it is possible to use byte address incrementing, the design supports up to 16-burst operation, i.e., 16byte. therefore, from the send/receive data, it is possible to use byte address incrementing. A total of 16 bytes (4 words) after the receive start address are reserved for the transmit/receive data window.	32'h0

18.5.11 RX Starting address register

Table 159 UART&7816 RX Starting Address Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value

[31:0]	RO	<p>rx data window (UART&7816 mode)</p> <p>Receive data start address.</p> <p>Note: uart sends and receives data in bytes only, when burst transmission is used, it is possible to use words to send and receive data.</p> <p>The section address incremental method is designed to support up to 16-burst operation, i.e., 16 bytes, so a total of 16 bytes (4 words) from the transmit/receive start address are reserved for the transmit/receive data window.</p>	32'h0
--------	----	--	-------

18.5.12 7816 Protection Time Register

Table 160 7816 Protection Time Registers

classifier for honori- fic peop- le	intervi- ews	Operating Instructions	reset value
[31:8]		reservations	
[7 : 0]	RW	ex_gt_num 7816 mode, guard time calculation: 10 + stop bit + configured value ex_gt_num	8'h0

18.5.13 7816 Timeout Time Register

Table 161 7816 Timeout Time Registers

classifier for honori- fic people	intervi- ews	Operating Instructions	reset value
[31:24]		reservations	
[23:0]	RW	wait time counter (in ETUs) In 7816 mode: CWT and BWT time, configured to the maximum default value. (in T1 mode: BWT = (11 etu+ 2BWI*960*Fd/fsc))	24'h78000

19 Timer Module

19.1 Functional overview

The timer consists of a 32-bit auto-loading counter driven by the system clock after dividing the frequency. The W800 has 6 fully independent timers. The W800 has six fully independent timers, which enable precise timing and interrupt functions for delayed or periodic event processing.

19.2 Main characteristics

- 6 fully independent timers
- 32-bit auto-load counter
- Timing unit can be configured as ms, us
- Single time or repeated time function can be realized.
- Timed Interrupt Function
- The timer count value can be checked at any time;

19.3 Functional Description

The timer module consists of 6 completely independent timers that do not affect each other, and all 6 can work at the same time.

The system clock is divided by a frequency divider to obtain the us standard clock, which is used as the input clock of the counter. The timing unit can be configured as us and ms.

The timer value is a 32-bit configurable register for different timing durations. Each timer corresponds to an interrupt. When the timing time is satisfied, an interrupt request is generated if the interrupt function is enabled, which can be used to handle periodic events.

19.3.1 timing function

Timing function is based on the user to set the time, when the time to generate hardware interrupt, notify the user to realize a specific function. Timing triggers support both single and periodic, one can be used to handle single events and one can be used to handle periodic events.

The user obtains the APB bus clock frequency based on the division factor of the system clock and sets the reference microsecond count configuration register for the timer. (TMR_CONFIG) sets the timing value, configures the timing unit, working mode, enables the interrupt, and then, starts the timing function. When the timing time is up, the program enters the timer interrupt handling function to clear the interrupt.

19.3.2 time-delay function

The delay function means that the user can rely on the countdown function of the timer to keep the program in a waiting state until the timer finishes and the program continues to run.

19.4 register description

19.4.1 register list

Table 162 Timer Register List

offset address	name (of a thing)	abridge	inter views	descriptive	reset value
0X0000	Standard us Configuration Registers	TMR_CONFIG	RW	The standard us timing divider value, which is derived from the bus clock divider, is equal to the APB bus frequency in MHz minus the standard us timing.	0X0000_0027

				-	
0X0004	Timer Control Register	TMR_CSR	RW	Timer Control Register	0X0631_8C63
0X0008	Timer 1 Timer Value Configuration Register	TMR1_PRD	RW	Timer1 Timer value configuration register	0X0000_0000
0X000C	Timer 2 Timer Value Configuration Register	TMR2_PRD	RW	Timer2 Timer Value Configuration Register	0X0000_0000
0X0010	Timer 3 Timer Value Configuration Registers	TMR3_PRD	RW	Timer3 Timer Value Configuration Register	0X0000_0000

0X0014	Timer 4 Timer Value Configuration Registers	TMR4_PRD	RW	Timer4 Timer Value Configuration Register	0X0000_0000
0X0018	Timer 5 Timer Value Configuration Registers	TMR5_PRD	RW	Timer5 Timer Value Configuration Register	0X0000_0000
0X001C	Timer 6 Timer Value Configuration Registers	TMR6_PRD	RW	Timer6 Timer Value Configuration Register	0X0000_0000
0X0020	Timer 1 Current count value	TMR1_CNT	RO	Timer1 Current count value	0X0000_0000
0X0024	Timer 2 Current count value	TMR1_CNT	RO	Timer2 Current count value	0X0000_0000
0X0028	Timer 3 Current count value	TMR1_CNT	RO	Timer3 Current count value	0X0000_0000
0X002C	Timer 4 Current count value	TMR1_CNT	RO	Timer4 Current count value	0X0000_0000
0X0030	Timer 5 Current count value	TMR1_CNT	RO	Timer5 Current count value	0X0000_0000
0X0034	Timer 6 Current count value	TMR1_CNT	RO	Timer6 Current count value	0X0000_0000

19.4.2 Standard us configuration registers

Table 163 Timer Standard Us Configuration Registers

class fier for hono rific peop le	intervi ews	Operating Instructions	reset value
[31: 7]		reservations	
[6 : 0]	RW	Clock divider configuration prescale. for example: apb_clk=40MHz	7'h27

		prescale= 40 - 1= 8'd39	
--	--	-------------------------	--

19.4.3 Timer Control Register

Table 164 Timer Timer Control Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
---	----------------	------------------------	-------------

[31:30]	RW	reservations	2'h0
[29:25]	RW	TMR6_CSR, same as TMR1_CSR	5'h3
[24:20]	RW	TMR5_CSR, same as TMR1_CSR	5'h3
[19:15]	RW	TMR4_CSR, same as TMR1_CSR	5'h3
[14:10]	RW	TMR3_CSR, same as TMR1_CSR	5'h3
[9 : 5]	RW	TMR2_CSR, same as TMR1_CSR	5'h3
[4 : 0]	RW	[4 : 0] for TMR1_CSR as follows:	
		[4]: Interrupt status register, write 1 clear	
		1'b0: Timer generates no interrupt;	1'b0
		1'b1: Timer generates interrupt;	
		[3]: Interrupt enable registers	
		1'b0: No interrupt is generated after the timing time is completed;	1'b0
		1'b1: Generate an interrupt when the timing time is completed;	
		[2]: timer enable register	
		1'b0: timer not working;	1'b0
		1'b1: enable timer	
		[1]: timer operating mode	
		1'b0: timer repeat timing;	1'b1
		1'b1: The timer is timed only once and is automatically turned off when the timer is finished;	

	[0]: timer timing unit 1'b0: timing unit is us; 1'b1: Timing unit in ms;	1'b1
--	--	------

19.4.4 Timer 1 Timer Value Configuration Register

Table 165 Timer 1 Timer Value Configuration Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:0]	RW	Configure the timer value for Timer 1	32'b0

19.4.5 Timer 2 Timer Value Configuration Register

Table 166 Timer 2 Timer Value Configuration Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:0]	RW	Configure the timer value for Timer 2	32'b0

19.4.6 Timer 3 Timer Value Configuration Registers

Table 167 Timer 3 Timer Value Configuration Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:0]	RW	Configure the timer value for Timer 3	32'b0

19.4.7 Timer 4 Timer Value Configuration Registers

Table 168 Timer 4 Timer Value Configuration Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:0]	RW	Configure the timer value for Timer 4	32'b0

19.4.8 Timer 5 Timer Value Configuration Registers

Table 169 Timer 5 Timer Value Configuration Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:0]	RW	Configure the timer value for Timer 5	32'b0

19.4.9 Timer 6 Timer Value Configuration Registers

Table 170 Timer 6 Timer Value Configuration Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:0]	RW	Configure the timer value for Timer 6	32'b0

19.4.10 Timer 1 Current count value register

Timer 1 Current count value register

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:0]	RO	Reads the current count value of Timer 1;	32'b0

19.4.11 Timer 2 Current Count Register

Timer 2 Current Count Register

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:0]	RO	Reads the current count value of Timer 2;	32'b0

19.4.12 Timer 3 Current Count Register

Timer 3 Current Count Register

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:0]	RO	Reads the current count value of Timer 3;	32'b0

19.4.13 Timer 4 Current Count Value Register

Timer 4 Current Count Value Register

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:0]	RO	Reads the current count value of Timer 4;	32'b0

19.4.14 Timer 5 Current Count Register

Timer 5 Current Count Value Register

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:0]	RO	Reads the current count value of Timer 5;	32'b0

19.4.15 Timer 6 Current Count Register

Timer 6 Current Count Register

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:0]	RO	Reads the current count value of Timer 6;	32'b0

Winner Micro

20 Power Management Module

20.1 Functional overview

The PMU realizes the switching of the hardware operating state of the chip and the power management during the state switching process, and at the same time provides a timer, a real-time clock, and a 32K clock.

20.2 Main characteristics

- Provides chip power control
- Provides timer function
- Provides real-time clock control
- 32K RC oscillator calibration available
- Provides a wake-up function;

20.3 Functional Description

20.3.1 Full chip power control

The PMU module controls the power switching of the chip, including the 40M startup circuit, BandGap, digital PLL, voltage detection circuit, and digital circuit LDO.

When the chip is powered up, the PMU module guides each module to turn on the power in turn according to the preset power-up sequence;

When the software configuration register enters the sleep mode, it guides each functional module to turn off the power in sequence according to the safe power-down sequence; when the software configuration register enters the sleep mode, it turns off the clock, the crystal starting circuit and the related power supply

according to the sequence;

Three wake-up modes are provided in hibernate/sleep mode: Timer timer, RTC timer, or wake-up by pulling the special WAKEUP pin high.

20.3.2 Low Power Mode

There are 2 low power modes that can be selected by configuring the PMU register chip; Standby mode:

In this mode, the power supply of the digital power domain will be shut down, and only the PMU module works on the whole chip to provide wake-up and reset functions; at this time, the power consumption of the whole chip is about 15uA. After the power is turned off, all the data and contents stored in the memory will be lost, and the firmware will be reloaded after waking up, which is equivalent to a reboot;

Sleep mode:

In this mode, the power supply of the digital power domain will be retained, only the DPLL and crystal oscillator circuit will be shut down, and the clock will be cut off, at this time, the power consumption of the whole chip is around 1mA; the data and code stored in the memory will still be retained; the program will continue to run after waking up;

20.3.3 wake-up call (computing)

The PMU supports three wake-up modes, Timer wake-up, RTC wake-up and external IO wake-up.

Timer Wakeup

Configure the Timer0 module in the PMU to set the hibernation time before setting the hibernation/sleep mode in the software. When the system enters the sleep mode, Timer0 will wake up the system and generate the corresponding timer interrupt when the timer reaches the sleep time. After the system resumes operation, it is necessary to clear the interrupt status by writing '1'

to the corresponding status bit in the interrupt status register, otherwise, the next time the system enters hibernation mode, it will be woken up immediately by the interrupt;

RTC wakeup

Configure the RTC module in the PMU to set the hibernation time before setting the hibernation/sleep mode in software. When the system enters the hibernation mode, it will wake up the system and give the corresponding RTC interrupt when the RTC register reaches the hibernation time. After the system resumes operation, please write '1' to the corresponding status bit in the interrupt register 0x14 to clear the interrupt status, otherwise, the next time the system enters into sleep mode, it will be woken up immediately by the interrupt;

External IO Wakeup

After software hibernation/sleep, PMU will detect a specific Wakeup pin, the external controller can wake up the system by pulling this IO high and give the corresponding IO wakeup interrupt.PMU will not detect this IO status after leaving hibernation mode. After the system resumes operation, please clear the interrupt status by writing '1' to the corresponding status bit in the interrupt register 0x14, otherwise, the next time the system enters the sleep mode, it will be woken up immediately by the interrupt;

20.3.4 Timer0 Timer

The timer enable signal and timing time are configured through the AHB register. First set the timer value, then set the timer enable BIT to start the timer, when the timer time is reached, an interrupt is generated and the software clears the interrupt flag by writing BIT0 of the status register.

20.3.5 real time clock function

Reference Real-Time Clock Module

20.3.6 32K clock source switching and calibration

The W800 chip integrates a 32K RC oscillator as the clock source for the PMU module.

Due to changes in the operating environment and temperature, the output frequency of the 32K RC oscillator may change, resulting in timing deviations. Therefore, a 32K RC oscillator calibration function and a 32K clock switching function have been introduced in the PMU module to correct timing deviations.

1) 32K clock source switching

The 32K clock can be switched from 32K RC oscillator to 32K clock by setting bit3 of PS_CR register to 1. However, when the chip enters into sleep mode, the 40M clock will be turned off and bit3 will be cleared to 0 automatically. However, when the chip enters sleep mode, bit3 will be cleared to 0 automatically because the 40M clock will be turned off, and if the firmware still wants to use the precision timing function after waking up, it needs to reset bit3 to 1.

2) Calibration of 32K RC Oscillator Circuits

First set bit2 of PS_CR register to 0, then set bit2 of PS_CR register to 1.

Once calibrated, the 32K RC oscillator will be relatively accurate. However, if you want more accurate timing, it is recommended to use a 32K clock divided by a 40M clock, which will give you an accurate RTC clock, and the deviation will only be related to the frequency bias of the crystal.

20.4 register description

20.4.1 register list

Table 171 PMU Register List

offset address	name (of a thing)	abridge	intervi ews	descriptive	reset value
0X0000	PMU Control Register	PS_CR	RW	Used to configure 32K calibration, configure 32K clock source, set the STANDBY function of the chip	0X0000_0002
0X0004	PMU Timer 0	TIMER0	RW	Configure the timing value (in seconds) to enable timing tool	0X0000_0000
0X0008	reservations				
0X0014	PMU Interrupt Source Register	INT_SRC	RW	Provide PMU interrupt flag	0X0000_0000

20.4.2 PMU Control Register

Table 172 PMU Control Registers

classifier for honorific people	inter views	Operating Instructions	reset value

[31: 11]	RO		24'b0
[10]	RW	<p>Wake-up key interrupt polarity selection:</p> <p>0: Set interrupt when wakeup key is high;</p> <p>1: Set interrupt when wakeup key is high; valid only in Sleep interrupt;</p>	1'b0
[9:6]	RW	<p>Minimum number of hold times for key wakeup:</p> <p>Unit: 128ms;</p>	4'd01
[5]	RW	<p>DLDO_CORE Reference</p> <p>Voltage Source Selection 1:</p> <p>ABG</p> <p>0: DBG</p>	1'b1
[4]	RW	<p>32K Oscillator BYPASS Signal</p> <p>Valid high, set to 1, 32K is the frequency division of 40M clock. 2○</p>	1'b0
[3]	RW	<p>RC 32K oscillator calibration circuit start switch; 1'b0: calibration circuit reset;</p> <p>1'b1: Activate the calibration circuit;</p> <p>To activate the calibration function, this bit needs to be set to 0 first and then 1.</p>	1'b0
[2]	RW	<p>Enable key-triggered entry to sleep function 0: Not enabled;</p> <p>1: Pressing the io_wakeup button in active mode for the threshold time will trigger the io_wakeup button.</p>	1'b0

		io_sleep_flag Interrupt, reported to MCU.	
[1]	RW	Sleep Enable signal, high active. 1'b0: Chip wake-up state	1'b1

	<p>1'b1: Chip enters Sleep state</p> <p>If the WAKEUP pin is at an invalid level and no TIMER0/1 interrupt wakeup is configured, the chip enters the Sleep state when this register is valid;</p> <p>If the wake-up interrupt is generated, the chip will switch from Sleep state to wake-up state, and this bit will be cleared automatically when the wake-up condition is satisfied.</p> <p>Wake-up source: WAKEUP pin, TIMER0/TIMER1, RTC</p> <ol style="list-style-type: none"> 1) WAKEUP pin, active high; for the chip to enter the Sleep state, the WAKEUP pin must be low. To wake up, pull up WAKEUP pin to generate a wake-up interrupt to make the chip leave the sleep state. 2) TIMER0, timer wake-up interrupt. When the WAKEUP pin is low, TIMER0 sets the timing time and enables it, and when the timing time is up, a wake-up interrupt will be generated to make the chip leave the Sleep state. 3) RTC, timed time to wake up When the WAKEUP pin is low and the RTC timer expires, a wake-up interrupt is generated, causing the chip to leave the Sleep state. 	
--	---	--

[0]	RW	<p>STANDBY enable signal, active high. 1'b0: Chip wake-up state</p> <p>1'b1: Chip enters STANDBY state</p> <p>If the WAKEUP pin is at an invalid level and the TIMER0/1 interrupt wakeup is not configured, the chip enters the STANDBY state when this register is valid;</p> <p>If a wakeup interrupt is generated, the chip switches from the STANDBY state to the wakeup state, and the wakeup condition is full</p> <p>This bit is automatically cleared to 0.</p>	1'b0
-----	----	---	------

	<p>Wake-up source: WAKEUP pin, TIMER0/TIMER1, RTC</p> <p>4) WAKEUP pin, active high; for the chip to enter the STANDBY state, the WAKEUP pin must be low. To wake up, pull WAKEUP pin high to generate a wake-up interrupt, which causes the chip to leave the STANDBY state.</p> <p>5) TIMER0, timer wake-up interrupt.</p> <p>When the WAKEUP pin is low, TIMER0 sets the timing time and enables it, and a wake-up interrupt is generated when the timing time is up to make the chip leave the STANDBY state.</p> <p>6) RTC, timed time to wake up</p> <p>When the WAKEUP pin is low and the RTC timer expires, a wake-up interrupt is generated, causing the chip to leave the STANDBY state.</p>	
--	---	--

20.4.3 PMU Timer 0

Table 173 PMU Timer 0 Registers

classifier for honorific people	inter- views	Operating Instructions	reset value
[31:17]	RO	reservations	15'b0

[16]	RW	Timer0 enable bit 1'b0: Bit enable. 1'b1: Enable;	1'b0
[15:0]	RW	Timer0's timer value, unit: sec.	16'b0

20.4.4 PMU Interrupt Source Register

Table 174 PMU Interrupt Source Registers

classifier for honorific people	inter- views	Operating Instructions	reset value
[31: 9]	R	reservations	
[8]	RW	Displays the current power-up status: 1'b0: Power-up or reset initiated 1'b1: Wake from sleep, write 1 clear	1'b0
[7]	RO	reservations	1'b0
[6]	RO	reservations	1'b0
[5]	RW	RTC Timing interrupt flag bit: 1'b0: Timing interrupt generated 1'b1: no timed interrupt generation, write 1 clear	1'b0
[4]	RW	reservations	1'b0
[3]	RW	reservations	1'b0
[2]	RW	WAKEUP pin wake-up interrupt flag bit 1'b0: No WAKEUP wake-up interrupt generation 1'b1: WAKEUP interrupt generated, write 1 clear	1'b0

[1]	RW	reservations	1'b0
[0]	RW	Timer0 Timer interrupt flag bit: 1'b0: no Timer0 interrupt generation 1'b1: Timer0 interrupt generated, write 1 clear	1'b0

Winner Micro

21 Real Time Clock Module

21.1 Functional overview

The RTC is a BCD counter/timer provided by the PMU module. The two 32-bit registers contain seconds, minutes, hours, days, months, and years in binary-coded decimal format (BCD) which automatically corrects for months of 28, 29 (leap years) 30, and 31 days.

With the appropriate software configuration, the RTC can provide both clock and calendar functions, and can also be used as a timer, after the timer reaches the set time, it will generate an RTC interrupt, which can be used to wake up the system in the sleep state.

The RTC module has two clock sources that can be configured: a 40M clock division and an internal 32K clock. During normal operation, the software can configure which clock source to use; during sleep state, only the 32K clock can be used. If the RTC clock source is derived from the 40M clock division during normal operation, it will automatically switch to the 32K clock when it enters the sleep state, and the 32K clock will still be used when the system is woken up. Therefore, as long as the supply voltage remains within the operating range, the RTC module will not stop working regardless of whether the module is in normal operation or in sleep state.

21.2 Main characteristics

- Provides timing function
- Provide timer function
- Provides timed interrupts

21.3 Function

- Wake-up call
- Descripti system on

21.3.1 timekeeping function

The initial values of day, hour, minute and second can be configured in RTC Configuration Register 1, the initial values of year and month can be configured in RTC Configuration Register 2, and the timing function can be enabled in RTC Configuration Register 2.

After the RTC timing function is enabled, read RTC Configuration Register 1 to get the current day, hour, minute, and second values, and read RTC Configuration Register 1 to get the current day, hour, minute, and second values, and read RTC Configuration Register 1 to get the current day, hour, minute, and second values.

The current year and month values are available in memory 2.

21.3.2 timing function

Day, hour, minute, and second timer values can be configured in RTC Configuration Register 1, year and month timer values can be configured in RTC Configuration Register 2, and the timing function can be enabled in RTC Configuration Register 1.

An RTC interrupt is generated when the RTC timer reaches the timer time, and the interrupt status can be cleared by setting the RTC interrupt bit of the PMU interrupt source register to 1.

After the system enters sleep mode, the interrupt generated by the RTC timer wakes up the system.

21.4 register description

21.4.1 register list

The RTC module has a total of two 32-bit exclusive registers, and the RTC interrupt status requires querying the PMU interrupt source register.

Table 175 RTC Register List

offset address	name (of a thing)	abridg e	intervi ews	descriptive	reset value
0X000C	RTC Configuration Register 1	RTC_R1	RW	Configure the RTC day, minute, and second values, and configure the enable timing	0X0000_0000
0X0010	RTC Configuration Register 2	RTC_R2	RW	Configure RTC year and month values, configure enable timing	0X0000_0000

21.4.2 RTC Configuration Register 1

Table 176 RTC Configuration Register 1

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31]	RW	RTC Timer Interrupt Enable	1'b0

		1'b0: not enabled 1'b1: enable	
[30:29]		reservations	
[28:24]	RW	Initial daily value/timed daily value	5'b0
[23:21]		reservations	
[20:16]	RW	Hourly initial value/hourly timed value	5'b0
[15:14]		reservations	
[13: 8]		reservations	
[7 : 6]		reservations	
[5 : 0]	RW	Seconds initial value/seconds timer value	6'b0

21.4.3 RTC Configuration Register 2

Table 177 RTC Configuration Registers 2

classifier for honori- fic people	intervi- ews	Operating Instructions	reset value
[31:17]		reservations	
[16]	RW	RTC timing function enable bit 1'b0: not enable 1'b1: enable	1'b0
[15]		reservations	
[14: 8]	RW	Beginning of year/annual fixed value	7'b0

[7 : 4]		reservations	
[3 : 0]	RW	Initial monthly value/monthly fixed value	4'b0

22 watchdog module

22.1 Functional overview

Implements a "watchdog" function. Designed for global reset in case of system crash.

The watchdog generates a periodic interrupt, and the system software clears its interrupt flag after the interrupt is generated. If the interrupt is not cleared for more than a set period of time, a hard reset signal is generated to reset the system.

22.2 Main characteristics

- Provide timer function
- Provides reset function
- Provides timed interrupts

22.3 Functional Description

22.3.1 timing function

After setting the timing value to register WD_LD, set BIT0 of WDG_CTRL to 1 to start the timer, and the WDG module will generate a timing interrupt when the timing time is up to notify the program to handle. If BIT0 of register WD_CLR is not cleared, timing interrupt will be generated periodically.

The value of WD_LD is based on the APB clock unit, which is clocked off the 160M clock.

22.3.2 reset function

After setting the chip timing value WD_LD, start the timing and reset function (set BIT1/BIT0 of

WDG_CTRL) the WDG module starts the countdown, and when the timing time is up, the WDG generates a timing interrupt, and at the same time, if the BIT0 of WD_CLR is not cleared, the chip generates a reset signal in the next cycle of the timing time.

22.4 register description

22.4.1 register list

Table 178 WDG Register List

offset address	name (of a thing)	abridge	intervi ews	descriptive	reset value
0X0000	WDG Timer Load Register	WD_LD	RW	Configure timed values for repeated loading	0XFFFF_FFFF
0X0004	WDG Current Value Register	WD_VAL	RO	Get the current timer value	0XFFFF_FFFF
0X0008	WDG Control Register	WD_CTRL	RW	control register	0X0000_0000
0X000C	WDG Interrupt Clear Register	WD_CLR	WO	Interrupt Clear Register	0X0000_0000
0X0010	WDG Interrupt Source Register	WD_SRC	RO	Interrupt Source Register	0X0000_0000
0X0014	WDG Interrupt output register	WD_STATE	RO	Interrupt Output Status Register	0X0000_0000

22.4.2 WDG Timer Value Load Register

Table 179 WDG Timer Value Load Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value

[31:0]	RW	<p>Configure timed values for repeated loading</p> <p>The value of this register is counted in APB clock.</p> <p>For example, if the APB clock is 40MHZ, the maximum duration of the timing value is about 107s, i.e. 0xFFFFFFFF/40000000.</p>	32'hffff_ffff
--------	----	--	---------------

22.4.3 WDG Current Value Register

Table 180 WDG Current Value Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:0]	RO	<p>Get the current timer value</p> <p>To calculate the remaining time, simply read the current value.</p> <p>To calculate the elapsed time, simply subtract the value of register WD_LD from the value of register WD_VAL</p>	32'hffff_ffff

22.4.4 WDG Control Register

Table 181 WDG Control Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31: 2]		reservations	30'h0
[1]	RW	<p>reset enable bit</p> <p>1'b0: No reset signal is generated when the WDG reset condition is generated</p> <p>1'b1: Reset signal generated when WDG reset condition is generated</p>	1'b0
[0]	RW	<p>Timing enable bit</p> <p>1'b0: timer not working</p> <p>1'b1: Timer operation, generating periodic interrupts</p>	1'b0

22.4.5 WDG Interrupt Clear Register

Table 182 WDG Interrupt Clear Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31: 1]		reservations	31'h0

[0]	WO	Interrupt status clear bit, write any value to clear the current interrupt status	1'b0
-----	----	---	------

22.4.6 WDG Interrupt Source Register

Table 183 WDG Interrupt Source Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31: 1]		reservations	31'h0
[0]	RO	Interrupt source register, the timer function is turned on, will generate the interrupt at the same time	1'b0

22.4.7 WDG Interrupt Status Register

Table 184 WDG Interrupt Status Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31: 1]		reservations	31'h0
[0]	RO	Interrupt output status register. This interrupt is not generated after the timer is turned off, but WD_SRC may be 1	1'b0

23 PWM Controller

23.1 Functional overview

PWM is a method of digitally encoding the level of an analog signal. Through the use of high-resolution counters, the duty cycle of a square wave is modulated to encode the level of a specific analog signal. The PWM signal is still digital because a full amplitude DC supply is either fully available (ON) or fully absent (OFF) at any given moment. The voltage or current source is applied to the analog load in a repetitive pulse sequence that is either ON or OFF. On is when the DC supply is applied to the load and off is when the supply is disconnected. Any analog value can be encoded using PWM as long as the bandwidth is sufficient.

23.2 Main characteristics

- Supports 2-channel input signal capture function (PWM0 and PWM4 channels)
- Input signal capture function supports interrupt interaction mode and DMA transfer mode; DMA mode supports word-by-word operation.
- Supports 5-channel PWM signal generation
- 5-channel PWM signal generation supports one-shot generation mode and auto-load mode
- Supports 5-channel braking
- PWM output frequency range: 3Hz~160kHz
- Maximum accuracy of duty cycle: 1/256, Counter width with deadband inserted: 8bit
- Supports channel 0 and channel 1 synchronization, channel 2 and channel 3 synchronization.
- Supports complementary and non-complementary modes for channel 0 and channel 1, and complementary and non-complementary modes for channel 2 and channel 3
- Supports 5-channel synchronization

23.3

Functional

~~Description~~ Output Signal Capture

The PWM controller supports the signal capture function of two channels. The capture function of channel 0 can be activated by setting Bit24 of the PWM_CTL register, and the capture function of channel 4 can be activated by setting Bit1 of the PWM_CAP2CTL register. The level of the captured signal can also be set to flip or not. After the channel captures the corresponding signal, the capture number is updated to the corresponding capture registers PWM_CAPDAT (channel 0 capture number) and PWM_CAP2DAT (channel 4 capture number).

23.3.2 DMA Transfer Capture Count

When the capture function is enabled on channel 0 or channel 4, the counts in the capture registers can be quickly transferred to memory through the DMA channel to speed up user processing.

23.3.3 Supports single and automatic loading modes

All five output channels of the PWM controller support single output mode and auto-load mode. In single loading mode, after the channel outputs a specified number of cycles of waveforms, it will no longer output PWM waveforms; in auto-loading mode, after the channel outputs a specified number of cycles of waveforms, it will reload the number of cycles automatically, so as to continue to generate PWM waveforms.

23.3.4 Multiple output modes

The PWM controller supports independent output mode, i.e., each channel outputs independently without interfering with each other; dual-channel synchronization mode, i.e., the output of one channel is completely consistent with the output of another channel; five-channel synchronization

23.3

~~functional mode~~ The outputs of channels 1 to 4 are completely consistent with the output of channel 0; and ~~description~~ dual-channel complementary output, i.e., the waveform of the output of one channel is completely opposite to the waveform of the output of the other channel; It supports the deadband setting which is often used in complementary mode, and the deadband length can be set up to 256 clock cycles; it supports the braking mode, when the braking port detects the specified level, the output channel will output the braking level which has already been set.

A variety of output modes can be flexibly configured to meet the user's various PWM-related application scenarios.

23.4 register description

23.4.1 PWM Register List

Table 185 PWM Register List

offset address	name (of a thing)	abridge	interv iews	descriptive	reset value
0X0000	Clock Divider Register_01	PWM_CLKDIV01	RW	Dividing the clock for channel 0 and channel 1	0X0000_0000
0X0004	Clock Divider Register_23	PWM_CLKDIV23	RW	Dividing the clocks of channel 2 and channel 3	0X0000_0000
0X0008	control register	PWM_CTL	RW	Used to configure or control a number of configurable items	0X0000_0000
0X000C	cycle register	PWM_PERIOD	RW	Used to set the period of channel 0 to channel 4	0X0000_0000
0X0010	Periodicity Register	PWM_PNUM	RW	Used to set the signal generation of channel 0 to channel 4. Number of cycles	0X0000_0000
0X0014	Comparison Register	PWM_CMPDAT	RW	Used to store channel 0 through channel 4 comparison values. to produce different duty cycles	0X0000_0000
0X0018	Deadband Control Register	PWM_DTCTL	RW	Used to configure or control deadband-related configurable parameters. install	0X0000_0000
0X001C	Interrupt Control Register	PWM_INTEN	RW	To enable the relevant interrupts	0X0000_0000
0X0020	Interrupt Status Register	PWM_INTSTS	RW	Used to query the status of related interrupts	0X0000_0000

0X0024	Channel 0 Capture Register	PWM_CAPDAT	RO	Captures and counts the rise to channel 0. Along and Down	0X0000_0000
0X0028	Brake Control Register	PWM_BRKCTL	RW	Used to control the braking mode	0X0000_0000
0X002C	Clock Divider Register_4	PWM_CH4_reg1	RW	Dividing the clock of channel 4	0X0000_0000

0X0030	Channel 4 Control Register_1	PWM_CH4_reg2	RW	Setting the configuration items related to channel 4	0X0000_0000
0X0034	Channel 4 Capture Register	PWM_CAP2DAT	RO	Captures and counts the rise to channel 4. Along and Down	0X0000_0000
0X0038	Channel 4 Control Register_2	PWM_CAP2CTL	RW	Configuration items for channel 4 are set.	0X0000_0000

23.4.2 Clock Divider Register_01

Table 186 PWM Clock Division Register_01

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:16]	RW	CLKDIV1 CH1 Crossover Counter The number of divisions is determined by the counter value Note: The crossover frequency range is (0~65535), if no crossover frequency is required, enter 0 or 1.	16'h0
[15:0]	RW	CLKDIV0 CH0 Crossover counters a m e as CH1	16'h0

23.4.3 Clock Divider Register_23

Table 187 PWM Clock Division Register_23

classifier for honorifi c people	intervi ews	Operating Instructions	reset value

c people			
[31:16]	RW	CLKDIV3 CH3 Crossover counters same as CH1	16'h0
[15:0]	RW	CLKDIV2	16'h0

		CH2 Crossover Counter Same as CH1	
--	--	--	--

23.4.4 control register

Table 188 PWM Control Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:27]	RW	CNTEN Counter count enable 1'b0: stop counting 1'b1: start counting Note: Each bit controls each channel individually, in descending order of CH4, CH3, CH2, CH1, and CH0.	5'b0
[26]	--	reservations	1'b0
[25]	RW	CAPINV Capture reverse enable identification bit 1'b0: Capture mode input signal reverse disabled 1'b1: Capture mode input signal inverse valid, inverse to input signal	1'b0

[24]	RW	<p>CPEN</p> <p>Capture Function Enable Flag Bit</p> <p>1'b0: The CH0 capture function is disabled and the RCAPDAT and FCAPDAT values will not be updated;</p> <p>1'b1: CH0 Capture function active, captures and latches the PWM counters, which are stored in RCAPDAT (Rising Edge Latched) and FCAPDAT (Falling Edge Latched), respectively</p>	1'b0
[23:22]	RW	CNTTYPE3	2'b0

		<p>CH3 Counter counting method</p> <p>2'b00: Edge alignment mode (counter counting is incremental, only for capture mode)</p> <p>2'b01: Edge alignment mode (counter counting in decreasing mode, only for PWM mode)</p> <p>2'b10: Central alignment mode (for PWM mode only)</p> <p>Note: In PWM mode, when the counter is set to Align Along mode, you need to set the counting method to Decreasing.</p>	
[21:20]	RW	<p>CNTTYPE2</p> <p>CH2 Counter counts in the same way as CH3</p>	2'b0
[19:18]	RW	<p>CNTTYPE1</p> <p>CH1 Counter counts in the same way as CH3</p>	2'b0
[17:16]	RW	<p>CNTTYPE0</p> <p>CH0 Counter counts in the same way as CH3</p>	2'b0

[15:14]	RW	TWOSYNCEN 2 Channel Synchronization Mode Enable Signal 1'b0: 2-channel synchronization not allowed 1'b1: allows 2 channels to be synchronized. PWM_CH0 and PWM_CH1 have the same phase and the phase is determined by PWM_CH0; PWM_CH2 and PWM_CH3 have the same phase and the phase is determined by PWM_CH2 15bit Control CH3 and CH2	2'b0
---------	----	---	------

		14bit Control CH1 and CH0	
[13]	--	reservations	1'b0
[12]	RW	POEN PWM pin output enable bit 1'b0: PWM pin set to output state 1'b1: PWM pin set to tri-state state Note: only for CH0	1'b0
[11:8]	RW	CNTMODE PWM Generate Cycle Mode 1'b0: Single Mode 1'b1: Automatic loading mode Note: PWM_CMPDAT zeroes out during CNTMODE changes; each bit controls each channel individually, PW3, PW2, PW1, and PW0 in descending order.	4'h0
[7]	--	reservations	1'b0
[6]	RW	ALLSYNCEN All-channel synchronization mode enable signal 1'b0: All-channel synchronization not allowed 1'b1: Allows all channels to be synchronized, with PWM_CH0, PWM_CH1, PWM_CH2 and PWM_CH3 having a	1'b0

		The same phase is determined by PWM_CH0.	
[5:2]	RW	PINV PWM output signal polarity enable 1'b0: PWM output polarity flip not enabled	4'h0

		1'b1: PWM output polarity flip enable Note: Each bit controls each channel separately, from high to low, PW3, PW2, PW1, and PW0.	
[1:0]	RW	OUTMODE Output Mode 1'b0: Non-complementary mode for each two channels 1'b1: every two channels form a complementary pattern BIT1 controls CH2 and CH3 BIT0 Controls CH0 and CH1.	2'b0

23.4.5 cycle register

Table 189 PWM Cycle Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value

[31:24]	RW	<p>PERIOD3</p> <p>CH3 PERIOD REGISTER VALUE (NOTE: PERIOD cannot be greater than 255) "Along Alignment Mode (Counter Counting Method is Decreasing)":</p> <ul style="list-style-type: none"> ➤ PERIOD register value with cycle value (PERIOD + 1) ➤ Duty cycle = (CMP+1)/(PERIOD + 1) ➤ CMP>=PERIOD: PWM output fixed high ➤ CMP<PERIOD: PWM low width is (PERIOD-CMP), high width is (CMP+1) ➤ CMP=0: PWM low width is PERIOD, high width is 1; <p>"Center Alignment Mode":</p> <ul style="list-style-type: none"> ➤ PERIOD register value: period is 2*(PERIOD+1) 	8'h0
---------	----	--	------

		<ul style="list-style-type: none"> ➤ Duty cycle = $(2 \times \text{CMP} + 1) / (2 \times (\text{PERIOD} + 1))$ ➤ CMP > PERIOD: PWM continuously high ➤ CMP <= PERIOD: PWM low = $2 \times (\text{PERIOD} - \text{CMP}) + 1$. High level = $(2 \times \text{CMP}) + 1$ ➤ CMP = 0: The width of PWM low level is $2 \times \text{PERIOD} + 1$, the width of high level is 1. Note: The number of cycles should not be 255 in the "Middle Alignment Mode". No matter which alignment mode is selected, the channel period is determined by the number of divisions (N) and the number of periods (P), i.e., the input clock is 40MHz, and the clock frequency f_{div} after dividing is: $f_{\text{div}} = 40\text{MHz}/N$, N is the number of divisions (16bit). The output frequency f_{output} is: $f_{\text{output}} = f_{\text{div}} / P$, P is the number of cycles. Note: In PWM mode, when the counter is set to Align Along mode, you need to set the counting method to Decrementing. Way. 	
[23:16]	RW	PERIOD2 CH2 Period register value (Note: period can not be greater than 255) Same as PERIOD3	8'h0
[15:8]	RW	PERIOD1 CH1 Period register value (Note: period can not be greater than 255) Same as PERIOD3	8'h0
[7:0]	RW	PERIOD0 CH0 Period register value (Note: period can not be greater than 255) Same as PERIOD3	8'h0

23.4.6 Periodicity Register

Table 190 PWM Cycle Number Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:24]	RW	<p>PNUM3</p> <p>PWM3 Number of Generation Cycles</p> <p>Set PWM3 cycle number PNUM3, when PWM generates PNUM3 PWM signals, stop generating signals, and at the same time trigger the interrupt and set the interrupt status word.</p>	8'h0
[23:16]	RW	<p>PNUM2</p> <p>PWM2 generates the same number of cycles as PNUM3.</p>	8'h0
[15:8]	RW	<p>PNUM1</p> <p>PWM1 generates the same number of cycles as PNUM3.</p>	8'h0
[7:0]	RW	<p>PNUM0</p> <p>PWM0 generates the same number of cycles</p>	8'h0

		as PNUM3.	
--	--	-----------	--

23.4.7 Compare

Registers

Table 191 PWM Comparison Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:24]	RW	CMP3 PWM3 Compare Register Value	8'h0

		<p>"Along Alignment Mode (Counter Counting by Decreasing)":</p> <ul style="list-style-type: none"> ➤ PERIOD register value with cycle value (PERIOD + 1) ➤ Duty cycle = $(\text{CMP}+1)/(\text{PERIOD} + 1)$ ➤ CMP>=PERIOD: PWM output fixed bit high ➤ CMP<PERIOD: PWM low width is (PERIOD-CMP), high width is (CMP+1) ➤ CMP=0: PWM low width is PERIOD, high width is 1; <p>"Center Alignment Mode":</p> <ul style="list-style-type: none"> ➤ PERIOD register value: period is $2*(\text{PERIOD}+1)$ ➤ Duty cycle = $(2*\text{CMP}+1)/2*(\text{PERIOD}+1)$ ➤ CMP>PERIOD: PWM continuously high ➤ CMP<=PERIOD: PWM low = $2*(\text{PERIOD}-\text{CMP})+1$, high = $(2*\text{CMP})+1$ ➤ CMP=0: PWM low level width is $2*\text{PERIOD}+1$, high level width is 1. <p>No matter which alignment mode is selected, the channel period is determined by the number of divisions (N) and the number of periods (P), i.e., the input clock is 40MHz, and the clock frequency f_{div} after dividing is: $f_{\text{div}} = 40\text{MHz}/N$, and N is the number of divisions.</p> <p>(16bit). The output frequency f_{output} is: $f_{\text{output}} = f_{\text{div}} / P$, P is the number of periods.</p> <p>Note: In PWM mode, when the counter is set to Align Along mode, you need to set the counting method to Decreasing.</p>	
[23:16]	RW	CMP2 PWM2 Comparison Register Value Same as CMP3	8'h0

[15:8]	RW	CMP1 PWM1 Compare Register Value	8'h0
--------	----	-------------------------------------	------

		Same as CMP3	
[7:0]	RW	CMP0 PWM0 Compare register value same as CMP3	8'h0

23.4.8 Deadband Control Register

Table 192 PWM Deadband Control Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:22]	--	reservations	10'h0
[21]	RW	DTEN23 Whether channel 2 and channel 3 can be inserted with a valid deadband marker The Insert Deadband Valid signal is valid only when the complementary mode of the channel is turned on. And, if the insert valid signal is 0, the complementary signals of the two channel outputs are inserted without deadband 1'b0: invalid insertion of dead zone 1'b1: Insertion of deadband validity	1'b0
[20]	RW	DTEN01 Whether channel 0 and channel 1 can be inserted into the deadband is the same as DTEN23.	1'b0
[19:18]	--	reservations	2'b0

[17:16]	RW	DTDIV Deadband clock divider control 2'b00: deadband clock equal to reference clock (40MHz)	2'b0
---------	----	--	------

		2'b01: Deadband clock equal to the reference clock (40MHz) divided by two 2'b10: Deadband clock is equal to the reference clock (40 MHz) quadrature 2'b11: Deadband clock equal to octave of reference clock (40MHz)	
[15:8]	RW	DTCNT23 Deadband interval for channel 3 and channel 2 8bit Determines the deadband interval value, the deadband clock is determined by DTDIV.	8'h0
[7:0]	RW	DTCNT01 Deadband interval for channel 1 and channel 0 8bit Determines the deadband interval value, the deadband clock is determined by DTDIV.	8'h0

23.4.9 Interrupt Control Register

Table 193 PWM Interrupt Control Registers

classifier for honori- fic people	intervi- ews	Operating Instructions	reset value
[31:8]	--	reservations	24'h0
[7]	RW	DMA_request_EN DMA_request enable 1'b0: DMA_request is not active 1'b1: DMA_request is valid	1'b0

[6]	RW	FLIEN Falling edge cache interrupt enable bit 1'b0: Falling edge cache interrupt disabled 1'b1: Falling edge cache interrupt active	1'b0
-----	----	--	------

		Note: For CH0	
[5]	RW	<p>RLIEN</p> <p>Rising edge cache</p> <p>interrupt enable bit</p> <p>1'b0: Rising cache</p> <p>interrupt disabled</p> <p>1'b1: Rising edge cache interrupt active</p> <p>Note: For CH0</p>	1'b0
[4:0]	RW	<p>PIEN</p> <p>PWM cycle interrupt</p> <p>enable bit 1'b0: cycle</p> <p>interrupt disabled</p> <p>1'b1: cycle interrupt active</p> <p>Note: The interrupt is triggered when the counter reaches 0 and the number of PWM cycles satisfies PWM_PNUM.</p>	5'b0

23.4.10 Interrupt Status Register

Table 194 PWM Interrupt Status Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:10]	--	reservations	12'h0

[9]	RW	<p>OVERFL</p> <p>Counter Overflow Flag</p> <p>1'b0: Capture mode, counter counting process, counter not overflowed</p> <p>1'b1: Capture mode, counter overflow during counter counting</p> <p>Note: When the user clears CFLIF or CRLIF, this bit is also cleared at the same time</p>	1'b0
[8]	RW	FLIFOV	1'b0

		Falling edge delay interrupt identifies overrun status 1'b0: no falling edge delay interrupt generation when CFILF is 1 1'b1: When CFILF is 1, another falling edge delay interrupt occurs Note: When the user clears CFILF, this bit is also cleared at the same time.	
[7]	RW	RLIFOV Rising edge delay interrupt identifies the overrun state 1'b0: no rising edge delay interrupt generation when CRILF is 1 1'b1: When CRILF is 1, another rising edge delay interrupt occurs Note: When the user clears CRILF, this bit is also cleared at the same time.	1'b0
[6]	RW	CFLIF Capture Falling Edge Interrupt Identifier 1'b0: No Falling Edge Captured 1'b1: This bit is set to 1 when the falling edge is captured Note: This identification bit is cleared by writing 1; Note: For CH0	1'b0

[5]	RW	<p>CRLIF</p> <p>Capture rising edge</p> <p>interrupt flag 1'b0: no</p> <p>rising edge captured</p> <p>1'b1: This bit is set to 1 when a rising edge is captured Note: Clear this identification bit by writing 1;</p> <p>Note: For CH0</p>	1'b0
[4:0]	RW	PIF	5'b0

		<p>PWM cycle interrupt identifier</p> <p>When the PWM generates a PWM signal of the specified period, the marker is positioned 1; the marker is cleared by writing 1 to the software Note: Each bit identifies each channel separately, controlling PW4, PW3, PW2, PW1, and PW0 in descending order.</p>	
--	--	--	--

23.4.11 Channel 0 Capture Register

Table 195 PWM Channel 0 Capture Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:16]	RO	<p>PWM_FCAPDAT</p> <p>Capture Falling Edge Register</p> <p>Stores the current counter value when a falling edge of the input signal is present</p>	16'h0
[15:0]	RO	<p>PWM_RCAPDAT</p> <p>Capture Rising Edge Register</p> <p>Stores the current counter value when a rising edge of the input signal is present</p>	16'h0

23.4.12 Brake Control Register

Table 196 PWM Brake Control Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
---	------------------------	-------------------------------	--------------------

[31:16]	--	reservations	16'h0
[15:11]	RW	<p>BRKCTL</p> <p>Brake mode enable</p> <p>1'b0: brake mode disabled</p> <p>1'b1: Brake mode activation</p>	5'b0

		[7:3] correspond to CH4, CH3, CH2, CH1, and CH0 respectively	
[10:8]	--	reservations	3'b0
[7:3]	RW	<p>BKOD</p> <p>Brake Output Control Register</p> <p>1'b0: PWM output low when brake mode is active</p> <p>1'b1: When the braking mode is active, the PWM output goes high [7:3] corresponds to CH4, CH3, CH2, CH1 and CH0, respectively.</p>	5'b0
[2:0]	--	reservations	3'b0

23.4.13 Clock Divider Register_4

Table 197 PWM Clock Divider Register_4

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:16]	RW	<p>CLKDIV4</p> <p>CH4 Crossover Counter</p> <p>The number of divisions is determined by the counter value</p> <p>Note: The crossover frequency range is (0~65535), if no crossover frequency is required, enter 0 or 1.</p>	16'h0

[15:8]	RW	<p>PERIOD4</p> <p>CH4 PERIOD REGISTER VALUE (Note: PERIOD cannot be greater than 255) "Along Alignment Mode (Counter Counting Method is Decreasing)":</p> <ul style="list-style-type: none"> ➤ PERIOD register value with cycle value (PERIOD + 1) ➤ Duty cycle = (CMP+1)/(PERIOD + 1) ➤ CMP>=PERIOD: PWM output fixed bit high 	8'h0
--------	----	---	------

		<ul style="list-style-type: none"> ➤ CMP<PERIOD: PWM low width is (PERIOD-CMP), high width is (CMP+1) ➤ CMP=0: PWM low width is PERIOD, high width is 1; <p>"Center Alignment Mode":</p> <ul style="list-style-type: none"> ➤ PERIOD register value: period is $2*(PERIOD+1)$ ➤ Duty cycle = $(2*CMP+1)/(2*(PERIOD+1))$ ➤ CMP>PERIOD: PWM continuously high ➤ CMP<=PERIOD: PWM low = $2*(PERIOD-CMP)+1$, high = $(2*CMP)+1$ ➤ CMP=0: The width of PWM low level is $2*PERIOD+1$, the width of high level is 1. Note: The number of cycles should not be 255 in the "Middle Alignment Mode". <p>No matter which alignment mode is selected, the channel period is determined by the number of divisions (N) and the number of periods (P), i.e., the input clock is 40MHz, and the clock frequency f_{div} after dividing is: $f_{div} = 40MHz/N$, N is the number of divisions (16bit). The output frequency f_{output} is: $f_{output} = f_{div} / P$, P is the number of cycles.</p> <p>Note: In PWM mode, when the counter is set to Align Along mode, you need to set the counting method to Decrementing.</p> <p>Way.</p>	
[7:0]	RW	<p>CH4 Number of generation cycles</p> <p>Set the number of PWM4 cycles to PNUM4, and when the PWM generates PNUM4 PWM signals, it stops generating signals, and at the same time triggers an interrupt and sets the interrupt status word.</p>	8'h0

23.4.14 Channel 4 Control Register_1

Table 198 PWM Channel 4 Control Register_1

classifier for honorifi c people	intervi ews	Operating Instructions	reset value
[31:16]	--	reservations	16'h0

[15:8]	RW	<p>CMP4</p> <p>CH4 Cycle Register Value</p> <p>"Along Alignment Mode (Counter Counting by Decreasing)":</p> <ul style="list-style-type: none"> ➤ PERIOD register value with cycle value (PERIOD + 1) ➤ Duty cycle = $(\text{CMP}+1)/(\text{PERIOD} + 1)$ ➤ $\text{CMP} \geq \text{PERIOD}$: PWM output fixed bit high ➤ $\text{CMP} < \text{PERIOD}$: PWM low width is $(\text{PERIOD}-\text{CMP})$, high width is $(\text{CMP}+1)$ ➤ $\text{CMP}=0$: PWM low width is PERIOD, high width is 1; <p>"Center Alignment Mode":</p> <ul style="list-style-type: none"> ➤ PERIOD register value: period is $2^*(\text{PERIOD}+1)$ ➤ Duty cycle = $(2^*\text{CMP}+1)/2^*(\text{PERIOD}+1)$ ➤ $\text{CMP} > \text{PERIOD}$: PWM continuously high ➤ $\text{CMP} \leq \text{PERIOD}$: PWM low = $2^*(\text{PERIOD}-\text{CMP})+1$, high = $(2^*\text{CMP})+1$ ➤ $\text{CMP}=0$: PWM low level width is $2^*\text{PERIOD}+1$, high level width is 1. <p>No matter which alignment mode is selected, the channel period is determined by the number of divisions (N) and the number of periods (P), i.e., the input clock is 40MHz, and the clock frequency f_{div} after dividing is: $f_{\text{div}} = 40\text{MHz}/N$, and N is the number of divisions.</p> <p>(16bit). The output frequency f_{output} is: $f_{\text{output}} = f_{\text{div}} / P$, P is the number of periods.</p> <p>Note: In PWM mode, when the counter is set to Align Along mode, you need to set the counting method to Decreasing.</p>	8'h0
[7:5]	--	reservations	3'b0

[4:3]	RW	CNTTYPE4 CH4 Counter counting method	2'b0
-------	----	---	------

		<p>2'b00: Edge alignment mode (counter counting is incremental, only for capture mode)</p> <p>2'b01: Edge Alignment Mode (counter counting in decreasing mode, only for PWM mode)</p> <p>2'b10: Central alignment mode (for PWM mode only)</p> <p>Note: In PWM mode, when the counter is set to Align Along mode, you need to set the counting method to Decreasing.</p>	
[2]	--	reservations	1'b0
[1]	RW	<p>CNTMODE4</p> <p>CH4 Generate</p> <p>cycle mode 1'b0:</p> <p>Single mode</p> <p>1'b1: Automatic loading mode</p> <p>Note: PWM_CMPDAT zeroes out during CNTMODE changes</p>	1'b0
[0]	RW	<p>PINV4</p> <p>CH4 Output signal polarity</p> <p>enable 1'b0: PWM output</p> <p>polarity flip not enabled</p> <p>1'b1: PWM output polarity flip enable</p>	1'b0

23.4.15 Channel 4 Capture

Registers

Table 199 PWM Channel 4 Capture Registers

classifier for honorifi c people	intervi ews	Operating Instructions	reset value

[31:16]	RO	PWM_FCAP2DAT Capture Falling Edge Register Stores the current counter value when a falling edge of the input signal is present	16'h0
---------	----	--	-------

[15:0]	RO	PWM_RCAP2DAT Capture Rising Edge Register Stores the current counter value when a rising edge of the input signal is present	16'h0
--------	----	--	-------

23.4.16 Channel 4 Control Register_2

Table 200 PWM Channel 4 Control Register_2

classifier for honori- fic people	intervi- ews	Operating Instructions	reset value
[31:11]	--	reservations	21'h0
[10]	RW	DMA_request2_mask DMA_request2 enable 1'b0: DMA_request2 not valid 1'b1: DMA_request2 valid Note: only for CH4	1'b0
[9]	RW	FLIEN2 Falling edge cache interrupt enable bit 1'b0: 1'b0: Falling edge cache interrupt disabled 1'b1: Falling edge cache interrupt active Note: only for CH4	1'b0

[8]	RW	RLIEN2 Rising edge cache interrupt enable bit 1'b0: Rising cache interrupt disabled 1'b1: Rising edge cache interrupt active Note: only for CH4	1'b0
-----	----	---	------

[7]	RW	<p>OVERFL2</p> <p>Counter Overflow Flag</p> <p>1'b0: Capture mode, counter counting process, counter not overflowed</p> <p>1'b1: Capture mode, counter overflow during counter counting</p> <p>Note: When the user clears CFLIF or CRLIF, this bit is also cleared at the same time Note: Only for CH4.</p>	1'b0
[6]	RW	<p>FLIFOV2</p> <p>Falling edge delay interrupt identifies overrun status</p> <p>1'b0: no falling edge delay interrupt generation when CFILF is 1</p> <p>1'b1: When CFILF is 1, another falling edge delay interrupt occurs Note: When the user clears CFILF, this bit is also cleared at the same time.</p> <p>Note: only for CH4</p>	1'b0
[5]	RW	<p>RLIFOV2</p> <p>Rising edge delay interrupt identifies the overrun state</p> <p>1'b0: no rising edge delay interrupt generation when CRILF is 1</p> <p>1'b1: When CRILF is 1, another rising edge delay interrupt occurs Note: When the user clears CRILF, this bit is also cleared at the same time.</p> <p>Note: only for CH4</p>	1'b0

[4]	RW	CFLIF2 Capture Falling Edge Interrupt Identifier 1'b0: No Falling Edge Captured 1'b1: This bit is set to 1 when the falling edge is captured	1'b0
-----	----	---	------

		Note: Clear this bit by writing a 1. Note: only for CH4	
[3]	RW	<p>CRLIF2</p> <p>Capture rising edge</p> <p>interrupt flag 1'b0: no</p> <p>rising edge captured</p> <p>1'b1: This bit is set to 1 when a rising edge is captured Note: This identification bit is cleared by writing a 1</p> <p>Note: only for CH4</p>	1'b0
[2]	RW	<p>POEN2</p> <p>PWM pin output enable bit</p> <p>1'b0: PWM pin set to output state</p> <p>1'b1: PWM pin set to tri-state state</p> <p>Note: only for CH4</p>	1'b0
[1]	RW	<p>CPEN2</p> <p>Capture Function Enable Flag Bit</p> <p>1'b0: The CH4 capture function is disabled and the RCAPDAT and FCAPDAT values will not be updated;</p> <p>1'b1: CH4 capture function active, captures and latches PWM counters, which are stored in RCAPDAT (rising edge latch) and FCAPDAT (falling edge latch) respectively</p> <p>Note: only for CH4</p>	1'b0

[0]	RW	CAPINV2 Capture reverse enable identification bit 1'b0: Capture mode input signal reverse disabled	1'b0
-----	----	--	------

		1'b1: Capture mode input signal inverse valid, inverse to input signal Note: only for CH4	
--	--	--	--

24 QFLASH Controller

24.1 Functional overview

The W800 has a built-in QFLASH controller, which provides bus-based QFLASH read, write and erase operations, system bus and data bus access arbitration, and CACHE-based QFLASH read operations.

24.2 Main characteristics

- Provides bus access to the FLASH interface
- Supports direct SPI command access to FLASH through register configuration.
- Supports 24-bit or 32-bit address access to FLASH
- Supports automatic decryption and reading of code and data stored in FLASH.

24.3 Functional Description

24.3.1 bus access

The FLASH address in the system starts from 0x08000000, and read operations can be performed directly on this address space.

24.3.2 register access

SPI commands can be sent directly to the FLASH through the configuration of registers to realize more flexible access to the FLASH and support most of the FLASH control commands.

24.3.3 Command configuration and startup

The Flash control command code is written to the command bit of FLASH_CMD, and the other bits of the register are configured according to the command format, e.g., if there is an address bit in

the command, set the address bit to 1. If there is a data bit in the command, set the DAT bit to 1, and the command format is configured according to the command format, e.g., if there is a data bit in the command, set the DAT bit to 1.

Body reference

register description.

After the command configuration is complete, set the command_b of the CMD_START register to position 1, then the controller starts to send the command to the FLASH.

24.3.3.1 Communication Mode and Configuration

First send a command to the FLASH to configure the FLASH to QIO or QPI mode, then set the QIOM or QPIM position of the FLASH controller to 1 to communicate with the FLASH using QIO mode or QPI mode.

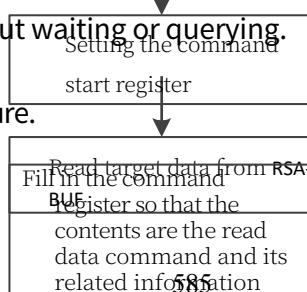
24.3.3.2 data cache

When reading or writing data to FLASH through registers, the storage space of address 0x4000 0000 ~ 0x4000 03ff is used as data cache, and this address space is also used as data cache for RSA encryption and decryption module. The address space is also used as the data cache for RSA encryption and decryption module. That is, when reading data, the controller will sequentially store the data read from FLASH in the address space starting from 0x4000 0000 for caching. When writing data to FLASH, the controller also reads data from 0x4000 0000 and sends it to flash.

24.3.3.3 read operation

Each read operation can read up to 256 bytes of data, and can read data from any location in the QFlash. Once the read command is initiated, the data is stored in the cache without waiting or querying.

The operation flow is shown in the figure.



Body reference
register description.

24.3.3.4 write operation

The following commands are all write operations:

WRSR: Write Status

Register

PP: Page

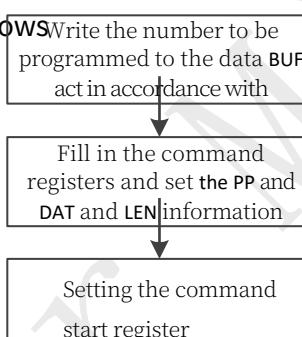
Programming SE:

Sector Erase BE: 32K

Block Erase HBE: 64K

Block Erase
CE: Full Chip Erase

The erase command does not require data, just configure the command registers directly to the appropriate command and start it. The procedure for programming commands is as follows



```
graph TD; A[Write the number to be programmed to the data BUF act in accordance with] --> B[Fill in the command registers and set the PP and DAT and LEN information]; B --> C[Setting the command start register]
```

24.4 register description

24.4.1 register list

Table 201 QFLASH Controller Register List

offset address	name (of a thing)	abridge	descriptive	default value
0X0000_0000	Command Message Register	CMD_INFO	Commands and data words required for QFLASH operation classifier for segments, e.g. lessons, train wagons, biblical verses	0X0000_0000
0X0000_0004	Command Launch Register	CMD_START	command activation	0X0000_0000
0X0000_0008	Flash Control Register	Flash_CR	QFLASH Operation Mode Control	0X0000_0000
0X0000_000C	Remap Register	Remap	Remap Option Control	0X0000_0000
0X0000_0010	ADDR Register	Flash_ADDR	operating address	0X0000_0000
0X0000_0014	Decryption Control Register	DECRYPT_CR	Firmware Confidential Mode Control	0X0000_0000
0X0000_0018	Decryption Status Register	DECRYPT_STA	Firmware decryption status;	0X0000_0000
0X0000_0200 ... 0X0000_0600	Read and Write Data Buffer	RSA_BUF	<p>It is used to cache QFLASH read/write data or RSA encryption/decryption data.</p> <p>This memory is shared by the QFLASH controller and the RSA module. Therefore, QFLASH read/write and RSA operations cannot be used at</p>	0X0000_0000

			<p>the same time, and need to be realized in the driver.</p> <p>Adding Mutually Exclusive Operation Protection</p>	
--	--	--	--	--

24.4.2 Command Message Register

Table 202 QFLASH Command Message Store

classification for honorific people	interviews	Operating Instructions	reset value

[31]	RW	1: Indicates that CMD_START carries the Address, and CMD_START register holds the Address in the lower 20 percent of the register. The upper 4 bits of this Address are fixed to 0, for a total of 24 bits.	1'b0
[30]	RW	1: Indicates that CRM carries contents in the CMD_START register	1'b0
[29]	RW	1: Instructs CMD_INFO to carry the Dummy cycle	1'b0
[28:26]	RW	Its content +1 indicates how many Dummy cycles CMD_INFO carries.	3'b0
[25:16]	RW	Its content +1 indicates how many bytes of data CMD_INFO carries.	10'b0
[15]	RW	1: Indicates that CMD_INFO carries the data	1'b0
[14]	RW	1: Indicates that the command currently filling in the CMD_INFO command field is a read command, and the command contains RDSR, FR, and QIOFR, RDPDRID, RSR, etc.	1'b0
[13]	RW	1: Indicates that the current CMD_INFO command field is filled with the WRSR command.	1'b0
[12]	RW	1: Indicates that the current command in the CMD_INFO command field is a PP command.	1'b0
[11]	RW	1: Indicates that the current CMD_INFO command field is filled with the SE command.	1'b0
[10]	RW	1: Indicates that the current command in the CMD_INFO command field is the BE command.	1'b0
[9]	RW	1: Indicates that the current CMD_INFO command field is filled with the HBE command.	1'b0
[8]	RW	1: Indicates that the current command in the CMD_INFO command field is a CE command.	1'b0
[7 : 0]	RW	CMD_INFO command field, fill in the commands for QFlash, refer to the QFlash chip manual.	8'b0

24.4.3 Command

Launch

Register

Table 203 QFLASH Command Startup Registers

classifier for honorific people	inter views	Operating Instructions	reset value
[31:29]	RO	Reserved	3'b0

[28]	RW	Setting 1 indicates a startup command, which is cleared by the hardware after the command operation is completed.	1'b0
[27: 8]	RW	Address[19:0]: If the A flag in the command register is 1, then the lower 20 bits of Address are stored here.	20'b0
[7 : 0]	RW	CRM[7:0]: If the CRM flag in the command register is 1, the CRM content is stored here.	8'b0

24.5 Frequently used commands in QFLASH

Table 204 QFLASH Common Commands

command name	command word	Command Abbreviations
Write Enable	06H	WREN
Write Disable	04H	WRDI
Read Status	05H	RDSR
Write Status	01H	WRSR
Fast Read	0BH	FR
Quad IO Fast Read	EBH	QIOFR
Page Program	02H	PP
Sector Erase	20H	SE

For other commands, see the QFLASH manual.

Winner Micro

25 PSRAM Interface Controller

25.1 Functional overview

W800 has a built-in PSRAM controller with SPI/QSPI interface, supports external PSRAM device access, and provides PSRAM read, write and erase operations in bus mode. The maximum read/write speed is 80MHz.

25.2 Main characteristics

- Supports read/write access to external PSRAM.
- Configurable for SPI and QSPI
- SPI/QSPI clock frequency configurable
- Supports BURST INC mode access
- Supports semi-sleep mode for PSRAM

25.3 Function al

Descripti
PSRAM is called Pseudo static random access memory, which refers to pseudo static random memory. Compared with traditional SRAM

It has the advantages of small package, large capacity, low cost, etc. It is mainly used in the direction of data caching in IoT applications. The interfaces are mostly SPI, QSPI, etc. The interface pins are mainly SCK, CS and 4 bi-directional data IOs. The interface pins mainly include the clock signal SCK, the chip select signal CS and 4 bi-directional data IOs, and the PSRAM controller provided by W800 can support the bus access of the PSRAM of the SPI/QSPI interface with a maximum operating clock rate of 80MHz and a maximum capacity of 64Mb.

25.3.1 Pin Description

SCLK: SPI interface clock, SCLK period is set by PSRAM_CTRL[7:4], the minimum settable divider value is 3, the default is 4 divisions of AHB clock.

CS: SPI interface chip select signal

SIO0: SPI Mode Data Input, QSPI Mode

SD[0]

SIO1: SPI Mode Data Output, QSPI Mode SD[1] SIO2:

QSPI Mode SD[2]

SIO3: QSPI Mode SD[3]

25.3.2 Access Mode Setting

The PSRAM controller supports two access modes, SPI for external PSRAM and QSPI for 4-wire, the default is SPI. configure the mode of SPI by setting PSRAM_CTRL[1].

PSRAM_CTRL[1] defaults to 0, i.e., SPI mode, when it takes 64 SCLK cycles to complete a write operation and 72 SCLK cycles to complete a read operation.

If the PSRAM is working in SPI mode, when writing 1 to PSRAM_CTRL[1] at this time, the controller will send command 35H to the PSRAM.

When reading PSRAM_CTRL[1] is 1, it means that the command sending is completed and PSRAM enters the QPI mode, at this time, to complete a write operation needs to be

16 SCLK cycles, and it takes 22 SCLK cycles to complete a read operation.

If PSRAM is working in QPI mode, when writing 0 to PSRAM_CTRL[1] at this time, the controller will send command F5H to PSRAM, when reading PSRAM_CTRL[1] as 0, it means that the command sending is completed, and PSRAM enters into SPI mode.

Setting the PSRAM operating mode must be done after the initialization operation and cannot be done at the same time.

25.3.3 PSRAM

Initialization

Before the first use, you need to write 1 to register PSRAM_CTRL[0] after the PSRAM is powered on and stabilized to start the PSRAM reset initialization operation, i.e., send 66H and 99H commands to the PSRAM. By default, the commands are sent in SPI mode, i.e. it takes 8 SCLK + tCPH + 8 SCLK. After the initialization is completed, the hardware automatically clears PSRAM_CTRL[0] to

SIO0: SPI Mode Data Input, QSPI Mode

SD[0]

The initialization operation restores the PSRAM to SPI mode.

The recommended initialization process is:

- (1) Write PSRAM_CTRL[0] to 1
- (2) Wait until PSRAM_CTRL[0] is automatically cleared to zero
- (3) Reset PSRAM controller by soft reset
- (4) Reset other required parameters of PSRAM_CTRL.

25.3.4 PSRAM Access Methods

PSRAM is read/written in the same way as normal SRAM, i.e., data is written/read to/from the corresponding bus address.

25.3.5 BURST function

Setting PSRAM_CTRL[2] enables the controller to support AHB bus-initiated BURST, i.e., when HBURST on the AHB bus is

1/3/5/7 indicates that the AHB bus initiates a sequential read/write with incremental address, at The CS does not pull up the CS after reading/writing a word, but reads/writes the next word directly.

OVERTIMER register is used to set the maximum time for CS to be low, and the unit is the number of cycles of HCLK. Every time a BURST operation is started, the internal counter starts counting from 0.

When the counter value is larger than the set value, the controller automatically stops the BURST operation and changes CS to high level directly after finishing reading/writing of the current word, if there is still data needing to be read/written in AHB bus at this time, it will be treated as a separate WORD. If there is still data to be read/written on the AHB bus at this time, it will be treated as a separate WORD.

The PSRAM controller does not support the WRAP form of BURST, so if accessing the PSRAM would require a WRAP BURST, set the

PSRAM_CTRL[2] is set to 0.

25.4 register description

25.4.1 register list

Table 205 PSRAM Controller Register List

offset address	name (of a thing)	abridge	descriptive	default value
0X0000_0000	control register	PSRAM_CTRL	PSRAM Controller Settings	0X0000_0000
0X0000_0004	Timeout Control Register	OVERTIMER	CS Timeout Control	0X0000_0000

25.4.2 Command Message Register

Table 206 PSRAM Control Setup Registers

classifier for honori fic people	inter views	Operating Instructions	reset value
[31:12]	RO	RSV	'b0
[11]	RW	HSM, Halfsleep mode Enable 1: Enable PSRAM half-sleep mode 0: Clear half-sleep mode	1'b0
[10:8]	RW	tCPH,CS High level shortest time setting, the unit is the number of AHB clock cycles, must be greater than 1, the specific time according to the different The same psram manual instructions to set up, not clear can not change the default values	3'd6
[7:4]	RW	SPI Crossover Setting Can only be configured to a value above 2, the value written is the multiple of the	4'd4

		crossover frequency.	
[3]	RO	RSV	
[2]	RW	INC_EN BURST Function Enable 1: Supports BURST function on the AHB bus 0: BURST function is not supported	1'b0

[1]	RW	QUAD Write 1 to enable QPI mode for PSRAM, write 0 to enable SPI mode. Read this flag bit to know which mode the PSRAM is currently in.	1'b0
[0]	RW	PSRAM reset Write 1 to initiate a reset operation of the PSRAM, which is automatically cleared when the reset is completed.	1'b0

25.4.3 Timeout Control Register

Table 207 CS Timeout Control Registers

classifier for honorific people	inter views	Operating Instructions	reset value
[11:0]	RW	Timeout register setting, sets the maximum time for CS to be low, for BURST mode	12'd0

26 ADC

26.1 ADC Functional Overview

Sigma-Delta ADC-based acquisition module, complete up to 4 analog signal acquisition, the sampling rate through the external input clock control, can be collected input voltage, can also be collected chip temperature, support input calibration and temperature compensation calibration.

26.2 ADC Key Features

- Supports up to 4 channels of data acquisition
- Supports DMA module for data buffering;
- Interrupt interaction mode is supported;
- Supports the function of comparing the collected data with the input data
- Maximum sampling frequency 1KHz;
- Supports single-ended input mode and differential input mode;

26.3 ADC Functional Description

26.3.1 Module Basic Structure

The ADC module is a PGA+SDADC structure, the input signal is pre-amplified by the PGA and then input to the SDADC for processing, the SDADC is designed as a 16-bit ADC, it should be noted that the ADC in the chip is powered by a 2.5V LDO, in order to protect the internal circuitry, the input signal has to satisfy the $V_{inmax} < 2.5V$, for other limitations, please refer to Section 26.3. For other restrictions, please refer to Section 26.3.8.

26.3.2 Channel Selection

Register 0x04[11:8] provides the channel selection function. The ADC of W800 provides 4-

channel signal acquisition function. By setting register 0x04[11:8], you can select any one of the 4 channels for single-ended signal data acquisition, or select 2 pairs of pre-paired differential channels among the 4 channels for differential signal data acquisition. See the register description for specific channel selection.

In addition to the 4 channels, the ADC module provides temperature detection, voltage detection, and offset calibration functions. These functions correspond to the channel selection of the 4'b1100, 4'b1101, 4'b1110.

If you need to use the corresponding function, configure the channel selection to the corresponding channel, and then follow the process of the above usage instructions.

26.3.3 data comparison

The ADC module provides a data comparison function, and the switch for this function can be set via BIIT[5] in configuration register 0x10.

Users can configure the value they wish to compare via BIT[17:0] of result register 0x18. This value needs to be converted to 18bit signed number, and the highest bit is the sign bit. BIT[6] of ADC configuration register 0x10 can set the direction of data comparison, when this bit is 0, if the value captured by ADC is greater than or equal to Comp_data, INT_CMP will be set to 1 and an interrupt will be generated; when this bit is 1, only when the value captured by ADC is less than Comp_data, INT_CMP will be set to 1 and an interrupt will be generated. When this bit is 1, INT_CMP will be set to 1 and an interrupt will be generated only if the value captured by ADC is less than Comp_data.

26.3.4 PGA Gain Adjustment Description

BIT[8:4] of register address 0X08, 5-bit gain control

signal, where 0X08[8:7]=GAIN_CTRL_PGA<4:3>

(corresponds to GAIN2)

0X08[6:4]=GAIN_CTRL_PGA<2:0> (corresponds to GAIN1, not used for

110 and 111) PGA overall gain GAIN_PGA=GAIN1*GAIN2

PGA Gain Configuration Table (expressed in magnification)

In addition to the 4 channels, the ADC module provides temperature detection, voltage detection, and offset calibration functions. These functions correspond to the channel selection of the [6:4]

[6:4]	00	01	10	11
000	1	2	3	4
001	16	32	48	64
010	32	64	96	128
011	64	128	192	256
100	128	256	384	512
101	256	512	768	1024

According to the simulation results, for the same amplification, it is recommended to prioritize the configuration in which GAIN2 amplification is larger.

26.3.5 Single-Ended Mode VCMIN Adjustment Description

The PGA internal VCMIN voltage, in single-ended mode, serves as the negative op-amp input.

The purpose of adjusting VCMIN is to accommodate a variety of signal sources with different bias points and to prevent saturation of the PGA output at certain bias points that are too high or too low.

BIT[16:11] of register address 0X40000D20, default 100000, i.e. VCMIN default=1.311V, step \approx 39.1mV

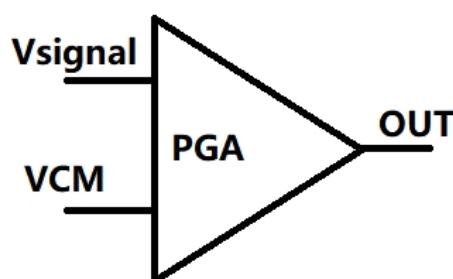


Figure 37 Principle of Single-Ended Operation Mode

26.3.6 Bypass Mode

Bypass_pga register address 0X08 BIT[3], default 0, not bypassed, PGA normal operation; if 1, at this time, the PGA is bypassed, the external signal source is directly connected to the SDADC input, mainly used for internal module testing, normal use does not need bypass. Bypass_ref register address 0X08 BIT[2], default 0, not bypassed, SDADC normal operation; if 1, at this time, the SDADC internal ref is bypassed, external reference voltage needs to be added through the test pin, mainly used for internal module testing, normal use does not need bypass. 0, no bypass, SDADC works normally; if it is 1, SDADC internal ref is bypassed at this time, need to add external reference voltage through the test pin, mainly used for internal module test, no bypass is needed for normal use.

26.3.7 SDADC Output Code Supplement

The SDADC output result is stored in the ADC_RESULT register at register address 0X00 BIT[17:0].

According to the simulation results, for the same amplification, it is
~~The output code of SDADC is not fixed, so the configuration signal can only amplify the digital processing of the larger.~~
output code of SDADC (inversion of the highest bit turns it into a signed number, and the 0 code of the output code of SDADC is not fixed (and it is not exactly 10 0000 0000 0000 0000) it is a mistake to use the value of the register for the subsequent calculations directly.

When calculating the output result of the SDADC, the value of register ADC_RESULT[17:0] (the read value is a hexadecimal number) needs to be calculated by the software first. Convert back to an unsigned number (the highest bit of the binary number, ADC_RESULT[17], is inverted) and then perform subsequent calculations.

The SDADC in this design is a 16bit ADC (design effective number of bits is 16bit) and the register data is 18bit, the lowest two bits are increased by digital filter, and the 16bit number is converted to 18bit number ($18\text{bit number} \approx 16\text{bit number} * 4$), so the effective number of bits is 16bit higher (i.e. ADC_RESULT[17:2])

In the read value back to the unsigned number, the calculation of the LSB, but also need to software 18bit data into 16bit data, that is, the lowest two data (ADC_RESULT[1:0]) only retain the high 16bit (back to the unsigned number of the ADC_RESULT[17:2]), and then use the 16bit data to carry out the subsequent calculations, at this time, LSB The simulation result of LSB is about 68.5uV (still need to test to determine the actual LSB of the chip)

26.3.8 Input signal voltage range

Because the NTO version of the PGA and SDADC are powered by a 2.5V LDO, there is a limitation on the input signal voltage range to prevent saturation of the internal circuitry.

Assuming the input differential signals are V_{inP} and V_{inN} (in single-ended mode, the signal V_{inN} is V_{CMIN}) the input differential voltage $V_{diff}=V_{inP}-V_{inN}$, and the input common-mode voltage $V_{cm}=(V_{inP}+V_{inN})/2$. The PGA has a gain of GAIN1 for the first stage, GAIN2 for the second stage, and an overall gain of the PGA, $GAIN_PGA=GAIN1*GAIN2$ (see Section 2.2.2 for details)

The input signal must satisfy both
of the following limitations:

$0V < V_{inP} < 2.5V$, $0V < V_{inN} < 2.5V$;

$0V < V_{cm} - (V_{diff}*GAIN1)/2 < 2.5V$;

$V_{cm} + (V_{diff}*GAIN1)/2 < 2.5V$;

When calculating the output result of the SDADC, the value of register ADC_RESULT[17:0] (the read value is a hex code) needs to be calculated by the software first.

$$0.12V + (V_{diff} * GAIN_PGA) / 2 < 2.5V;$$

For known approximate ranges of input signals, the available gain settings can be selected by directly substituting into the above equation calculation.

For unknown input signals, the x1 gain setting can be used first to determine the approximate range, and then substituted into the above equation to calculate and select the available gain setting.

26.4 Register Configuration for Typical Use Cases

The following configuration words are hexadecimal numbers; the high 0's are omitted.

26.4.1 Offset Measurement

1. Set module working status, set register 0X04 to 0xE03, turn on channel selection (offset detection)/SDADC_CHOP enable/SDADC enable/LDO enable.
2. Set module register 0X08=0x3, PGA is x1 Gain Configuration/PGA_CHOP Enable/PGA Enable. Among them, gain configuration

0X08 BIT[8:4] should be modified accordingly to the requirements of the test being performed (i.e., whatever gain configuration is used for the later test, the same gain configuration should be used for the OFFSET measurement)

3. Set clock state 0X40000E14 BIT[15:8]=0x28, confirm SDADC clock is configured to 1MHz, the default value is 0x28 after the chip is powered on, if there is no change, this step can be omitted.
4. Wait for 2ms, read out the output of ADC_RESULT register, refer to section 26.3.7, the software needs to convert back to an unsigned number (the highest bit of ADC_RESULT[17] is inverted), then discard the lowest two bits to keep only the high 16bit data. Repeat the reading for 10 times, with 2ms interval between readings, and calculate the average value of the code number as code_offset.

26.4.2 Differential Input Mode

1. The offset measurement is performed according to section 26.4.1 to obtain code_offset.
2. Set module register 0X04=0x803, channel selection (positive channel 0, negative channel 1)/SDADC_CHOP enable /SDADC enable/LDO enable. where channel selection 0X04[11:8] can be configured as a differential input for other channels, refer to section 26.3.2.
3. Set module register 0X08=0x3, PGA is x1 Gain Configuration/PGA_CHOP Enable/PGA Enable. Among them, gain configuration

26.4 Register Configuration for
~~0X081841 Select the~~ appropriate configuration according to the input signal voltage range, refer to sections 26.3.4 and 26.3.8.

4. Set clock state 0X40000E14[15:8]=0x28 to confirm that the SDADC clock is configured to 1MHz. the default value after the chip is powered up is

28, this step may be omitted if no changes are made

5. Wait for 2ms and read the output of the ADC_RESULT register, register address 0X00[17:0].

Refer to Section 26.3.7, which requires

The software first converts back to an unsigned number (the highest bit ADC_RESULT[17] is

inverted) and then rounds off the lowest two bits, retaining only the high 16bit data.

6. Repeat the readings for a total of 10 times, with readings spaced at intervals of 2ms, and

then calculate the average value of the code number as code_out.

7. Differential input signal Vindiff=(code_out-code_offset)*LSB/GAIN_PGA, where LSB is

approximately equal to 68.5uV according to the simulation results

26.4.3 Single-ended input mode

1. The offset measurement is first performed according to Section 26.4.1, resulting in code_offset.

2. Set module register 0X04=0x3, channel select (positive channel 0, negative VCMIN)/SDADC_CHOP enable/SDADC enable/LDO enable. The channel selection 0X04[11:8] can be configured as a single-ended input for other channels, refer to Section 26.3.2.

Set module register 0X408=0x3, PGA is x1 Gain Configuration/PGA_CHOP Enable/PGA Enable.

Among them, VCMIN configuration 0X40000D20[16:11] and gain configuration 0X08[8:4], select the appropriate configuration according to the input signal voltage range, refer to 26.3.4,

Sections 26.3.5 and 26.3.8

3. Set clock state 0X40000E14[15:8]=0x28 to confirm that the SDADC clock is configured to 1MHz. the default value after the chip is powered up is

28, this step may be omitted if there are no changes.

4. Wait for 2ms and read out the output of ADC_RESULT register, register address 0X00[17:0].

Refer to section 26.3.7, the software needs to convert back to an unsigned number (invert the

highest ADC_RESULT[17]) then discard the lowest two bits and keep only the high 16bit data. Repeat the

reading for 10 times, with 2ms interval between readings, and calculate the average value of the

code as code_out.

5. Wait for 2ms and read the output of the ADC_RESULT register, register address 0X00[17:0].
Refer to section 26.3.7 for more details.
where LSB is approximately equal to 68.5uV according to simulation results, and the default value of VCMIN 1.311V;

26.4.4 Temperature Detection Mode

1. Set module register 0X04=0xC03, channel selection (temperature detection)/SDADC_CHOP enable/SDADC enable/LDO enable;
2. Set module register 0X08=0x183, PGA for x4 gain configuration/PGA_CHOP enable/PGA enable;

3. Set the clock state 0X40000E14[15:8]=0x28 to confirm that the SDADC clock is configured to 1MHz. the default value after the chip is powered up is 28, this step may be omitted if no changes are made;
4. Set tempsensor module operating state 0X0C=0x1 , TempSensor enable/cal_offset_temp12=0 , TempSensor configured for x2 gain.
5. Wait for 2ms, read the output result of ADC_RESULT register, refer to section 26.3.7, it needs to be converted back to an unsigned number first by the software (the highest bit ADC_RESULT[17] is inverted) which is written as output code code_out1;
6. Set tempsensor module operating state 0X0C=0x3 , TempSensor enable/cal_offset_temp12=1 , TempSensor is x2 gain configuration;
7. Wait for 2ms, read the output result of ADC_RESULT register, refer to section 26.3.7, software is required to convert back to unsigned number first (the highest bit ADC_RESULT[17] is inverted) which is recorded as output code code_out2;
8. Repeat steps 4 through 7 for a total of 10 times to calculate the average values of code_out1 and code_out2, code_out1avg and code_out2avg, respectively;
9. From the above steps, $V_{temp} = (code_out1avg - code_out2avg) / 16$, and TempSensor output voltage= V_{temp} code number*(LSB/4) where LSB is about 68.5uV according to the simulation result (this LSB is the result of simulation according to 16bit, and the temperature detection in the above steps 5 and 7 is 18bit data, 18bit number≈16bit number*4, then the lowest valid bit corresponding to 18bit is ≈LSB/4). (This LSB is the result of the 16bit simulation. In steps 5 and 7 of the temperature measurement, the 18bit data is read, and the 18bit number is ≈ 16bit number * 4, so the least significant bit corresponding to 18bit is ≈ LSB/4).
10. According to the test results, the fitting formula is now organized, which can directly estimate the chip temperature

$$V_{temp} \text{ code} = (15.548 * \text{chip temperature}) + 4444.1$$

3. Set the clock state 0X40000E14[15:8]=0x28 to confirm that the SDADC clock is configured to
~~Substitute the default value calculated in step 9 into the above equation to obtain the chip~~

temperature. (Note that the fitting equation is in decimal) Also, add an optional measure

to improve accuracy:

For different chips, it is assumed that the coefficient k in the fitting equation $y=kx+b$ is unchanged (all are 15.548) and the coefficient b is somewhat different for different chips. According to the above instructions, we can first calculate the Vtemp code at a known ambient temperature (e.g., room temperature 25°C) and then substitute into the Vtemp code

$= (15.548 * (\text{known ambient temperature} + 11.8)) + b$, which yields the coefficient b for different chips, saved in rom or flash, replacing the above equation's

4444.1. then, the temperature is calculated according to the coefficients of the new formula, which can improve the temperature detection accuracy of different chips to some extent.

26.4.5 Voltage Detection Mode

1. The offset measurement is first performed according to Section 26.4.1, resulting in code_offset.
2. Set module register 0X04=0xD03, channel select (voltage detect)/SDADC_CHOP enable/SDADC enable/LDO enable
3. Set module register 0X08=0x83, PGA for x2 gain configuration/PGA_CHOP enable/PGA enable.
4. Set clock state 0X40000E14[15:8]= 28 confirm the SDADC clock configuration is 1MHz. the default value is 28 after the chip is powered on, if there is no change, this step can be omitted
5. Wait for 2ms, read out the output of ADC_RESULT register, refer to section 26.3.7, the software needs to convert back to an unsigned number (the highest bit of ADC_RESULT[17] is inverted), then discard the lowest two bits to keep only the high 16bit data. Repeat the reading for 10 times, with 2ms interval between readings, and calculate the average value of the code number as code_out;
6. Chip supply voltage $V_{power} = (code_out - code_offset) * LSB / 2 + 1.2V$, where LSB is approximately equal to according to simulation results

68.5uV

26.5 ADC Register Description

26.5.1 register list

Table 208 ADC Register List

offset address	name (of a thing)	abridge	descriptive	reset value
0X0000	ADC Result Register	ADC_RESULT	Storing ADCs Acquisition values and data comparison (be) worth	0X0000_0000
0X0004	ADC Analog Register	ADC_ANA_CTRL	Configuring ADC Related Functions	0X0000_0004
0X0008	PGA Configuration Register	PGA_CTRL	Configure PGA-related functions;	0x0000_0000
0X000c	TempSensor Configuration Hosting tool	TEMP_CTRL	Configure temperature sensor related functions;	0x0000_0000
0x0010	ADC Configuration Register	ADC_CTRL	Configure ADC module functions;	0x0050_0500
0x0014	ADC Interrupt Register	ADC_INT_STATUS	ADC module interrupt status register;	0x0000_0000
0x0018	Comparison Value Register	CMP_VALUE	Comparison threshold setting	0x0000_0000

26.5.2 ADC Result Registers

209 ADC Result Register

m

classifier for honorific people	inter views	Operating Instructions	reset value
ng	st	617	h)

[17:0]	RW	ADC conversion result value, valid bits Bit width 18 bits. signed number, the highest bit is the sign bit.	1'b0
--------	----	--	------

26.5.3 ADC Analog Configuration Register

Table 210 ADC Configuration Registers

classifier erfor honorif ic people	intervie ws	Operating Instructions	reset value
[11 : 8]	RW	<p>ADC operating channel selection signal:</p> <p>4'b0000: AIN0 channel operation, in DMA mode, corresponds to DMA channel 0 operation.</p> <p>4'b0001: AIN1 channel operation, in DMA mode, corresponds to DMA channel 1 operation.</p> <p>4'b0010: AIN2 channel operation, in DMA mode, corresponds to DMA channel 2 operation.</p> <p>4'b0011: AIN3 channel operation, in DMA mode, corresponds to DMA channel 3 operation.</p> <p>4'b0100: RSV</p> <p>4'b0101: RSV</p> <p>4'b0110: RSV</p> <p>4'b0111: RSV</p> <p>4'b1000: AIN0/AIN1 Differential Signal Inputs. in DMA mode, corresponds to DMA channel 0 operation.</p> <p>4'b1001: AIN2/AIN3 Differential Signal Inputs. corresponding to DMA channel 2 operation in DMA mode.</p> <p>4'b1010: RSV</p> <p>4'b1011: RSV</p> <p>4'b1100: Temperature sensor input, corresponding to DMA channel 2</p>	4'b1000

		<p>4'b1101: input for voltage detection module, corresponding to DMA channel 3 4'b1110: offset detection input.</p> <p>4'b1111: RSV</p>	
[7]	RO	RSV	1'b0
[6:5]	RW	chop_enr	2'b00

		LDO Chopper Signal PD Signal	
[4]	RW	<p>Chop_ens</p> <p>sdadc Chopper Signal</p> <p>PD Signal 0: Enable</p> <p>1: Not enabled</p>	1'b0
[3]	RO	RSV	1'b0
[2]	RW	<p>Pd_sdadc</p> <p>sdadc analog module</p> <p>power-down enable 1:</p> <p>power-down</p> <p>0: Work</p>	1'b1
[1]	RW	<p>Rstn_sdadc</p> <p>Analog Module Digital Logic</p> <p>Section Reset Signal 0: Reset</p> <p>1: Normal operation</p>	1'b0
[0]	RW	<p>en_ldo_sdadc</p> <p>ADC LDO</p> <p>enable</p> <p>0: power down 1: working</p>	1'b0

26.5.4 PGA Configuration Register

Table 211 PGA Configuration Registers

classifier for honorific people	intervie ws	Operating Instructions	reset value
[8:4]	RW	<p>Gain_ctrl_pga</p> <p>PGA gain configuration;</p> <p>BIT[8:7] configure GAIN2, BIT[6:4] configure GAIN1, refer to Section26.3.4 for the specific gain table.</p>	5'd0
[3]	RW	<p>Bypass_pga</p> <p>pga bypass signal 1:</p> <p>bypass pga</p> <p>0: No bypass</p>	1'b0
[2]	RW	<p>Bypass_ref</p> <p>Internal reference voltage bypass signal</p> <p>1: Bypass internal reference voltage</p> <p>0: No bypass</p>	1'b0

[1]	RW	<p>Chop_enp</p> <p>PGA Chopper</p> <p>Enable Signal 1:</p> <p>Enable</p> <p>0: not enabled</p>	1'b0
[0]	RW	<p>En_pga</p> <p>Pga enable</p> <p>signal 1:</p> <p>enable</p> <p>0: not enabled</p>	1'b0

26.5.5 TEMP Configuration Register

Table 212 Temperature Sensor Configuration Registers

classifier erfor honorif ic people	intervie ws	Operating Instructions	reset value
[5:4]	RW	Gain_temp temp Gain control code Gain_temp temp 00 -- 2 01 -- 4 10-- 6 11 -- 8	2'd0
[3:2]	RO	RSV	2'b0
[1]	RW	Cal_offset_temp12 TEMPSEN offset Calibration function	1'b0
[0]	RW	ON_TEMP 1: temp enabled 0: temp disable	1'b0

26.5.6 ADC Function Configuration Register

Table 213 Temperature Sensor Configuration Registers

classifier for honorific people	interviews	Operating Instructions	reset value
[29:20]	RW	<p>ana_swth_time</p> <p>The time it takes for the analog circuit to stabilize after the software switches the data channel is 80 pclk by default, i.e. 2us.</p>	10'h050

[19:18]	RO	RSV	2'b0
[17:8]	RW	<p>ana_init_time</p> <p>The time it takes for the analog circuit to stabilize after the software starts adc_start is 80 pclk by default, i.e. 2us.</p>	10'h050
[7]	RO	RSV	1'b0
[6]	RW	<p>Cmp_pol</p> <p>0: Interrupt generated at adc_result >= cmp_value</p> <p>1: Interrupt generated at adc_result < cmp_value</p>	1'b0
[5]	RW	<p>Cmp_int_ena 1:</p> <p>compare</p> <p>interrupt enable</p> <p>0: Compare interrupt not enabled</p>	1'b0
[4]	RW	<p>Adc_cmp_enable 1:</p> <p>adc compare</p> <p>function enable</p> <p>0: adc compare function not enabled</p>	1'b0
[3:2]	RO	Reserved	2'b0
[1]	RW	<p>Adc_int_ena</p> <p>1: ADC data conversion interrupt enable</p> <p>0: ADC data conversion interrupt not enabled</p>	1'b0
[0]	RW	<p>Adc_dma_enable</p> <p>1: dma enable</p> <p>0: dma not enabled</p>	1'b0

26.5.7 ADC Interrupt Status Register

Table 214 ADC Interrupt Status Registers

classification for honorific people	interview ws	Operating Instructions	reset value
[1]	RW	Cmp_int Compare Interrupt Flag Bit, Hardware Set, Software Write 1 Clear	1'b0
[0]	RW	Adc_int Data conversion complete interrupt, hardware set, software write 1 to zero	1'b0

26.5.8 Comparison Threshold Register

Table 215 Comparison Threshold Registers

classification for honorific people	interview ws	Operating Instructions	reset value
[17:0]	R/W	Cmp_value Values to be compared	18'h00000

27 Touch Sensor

27.1 Module Functionality Overview

The basic functions of the module are as follows:

- Supports up to 16 Touch Sensor scans;
- Records the results of each Touch Sensor scan;
- Report scan results via interrupt;

27.2 Description

of the use of

functions

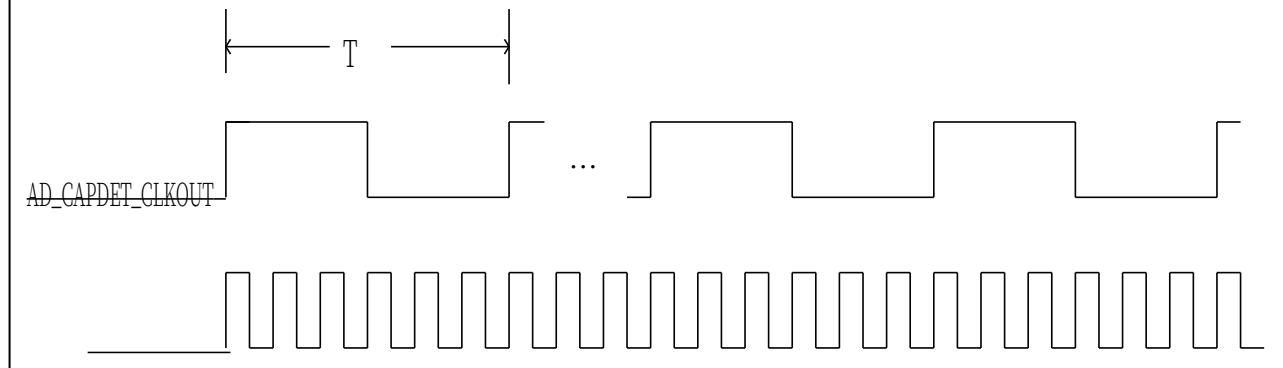
The basic principle of the integrated touch key module in the W800 system is that when the key is touched, the capacitance value of the circuit on the key will change, thereby

Affects the frequency of the output clock in the module. By detecting the frequency of the output clock in the module, it is possible to determine whether the capacitance value has changed, and thus determine the state of the key.

The basic function of this digital module is to scan the status of each touch button one by one at certain intervals, counting and recording within the set time window.

If the threshold is exceeded, it is judged that the key has been touched and reported to the MCU

system through an interrupt. If the set threshold is exceeded, the key is judged to have been touched and reported to the MCU system through an interrupt.



27.2.1 Basic workflow

1. Set Touch_CR register to configure scan_period, scan_window and select IOs to be scanned. scan_period in Touch_CR is the interval period of each scanning, the unit is 16ms, that is to say, if the register is set to 10, 16 touch keys will be scanned one by one in every 160ms. capdet_cnt is the window for counting when scanning each IO state, that is N in the above figure. CAPDET_CNT is the window for counting when scanning each IO state, i.e. N in the above figure, it should be noted that in order to avoid the jitter caused by switching channels, the counting will not start until the third pulse after switching the scanning IOs. therefore, if this register is set to N, the actual counting window is N-2.
 - + threshold, the key can be considered to be touched;
2. Sets the count threshold corresponding to each Touch IO. If the result of the scan exceeds the base actual counting window is N-2.
3. Enable Touch_CR[0], the module starts working.
4. After enabling the touch key module, the hardware will first scan the selected IOs one by one and store the count value of each IO as the base. Then it will scan the selected IOs one by one every scan_period and store the current value. After all IOs have been scanned, the count value of each IO will be compared with the base value, if the current value > base value + threshold, the key is considered to have been touched and the corresponding bit in PAD_STATUS in register 0x44 will be set to 1 and reported to the system via interrupt.

27.3 Register list:

Table 216 Touch Sensor Controller Register List

offset address	name (of a thing)	abridge	descriptive	default value
0X0000_0000	control register	Touch_CR	Touch Sensor Controller Settings	0X0000_0000

0X0000_0004 ~ 0X0000_0040	Touch key single channel control	Touch_Sensor x	Threshold control and counting values per touch key	0X0000_0000
0x0000_0044	Interrupt Controller	Int_Source	Interrupt Controller	0x0000_0000

27.3.1 Touch Sensor Control Register

Table 217 Touch Sensor Control Setting Register

classifier for honorific people	inter views	Operating Instructions	reset value
[31:26]	RW	<p>Scan_period</p> <p>Scanning period, unit is 16ms; for example, if scan_period is set to 6'd10, the key state will be scanned every 160ms;</p>	6'd10
[25:20]	RW	<p>CAPDET_CNT</p> <p>Select the number of CAPDET output pulses as the counting window.</p> <p>Note: To avoid jitter caused by switching channels, counting does not start until the beginning of the third pulse, so if this register is set to N, the actual counting window is N-2. e.g., if it is set to 6'd20, the counting window will be the period of 18 CAPDET pulses.</p>	6'd20

[19:4]	RW	Touch Sensor key selection; scanning the corresponding bit of touch button status. 0x0000: No scanning; 0x0001: Scan the first key; 0x0002: Scanning the second key;	16'd0
--------	----	--	-------

		0x0003: Scans the first and second keys; ... 0xFFFF: Scans all 16 keys;	
[3:1]	RW	RSV	3'd0
[0]	RW	Touch Key Controller Enable 1: Enable touch key scanning; 0: Disables touch key scanning;	1'b0

27.3.2 Touch Key Single Control Register

Table 218 Touch Key Single Setting Registers

classifier for honorific people	interviews	Operating Instructions	reset value
[22:8]	RO	Touch Senor 1 Count	14'd0
[7]	RO	RSV	1'b0
[6:0]	RW	Touch Senor 1 Threshold	7'd50

27.3.3 Interrupt Control

Registers

Table 219 Touch Key Interrupt Control Registers

classifier for honorific	interviews	Operating Instructions	reset value
--------------------------	------------	------------------------	-------------

ic people			
[31:16]	RW	<p>INT_EN</p> <p>If the corresponding bit is 1, it means the corresponding IO will generate interrupt when triggered; if the corresponding bit is 0, it means the corresponding IO will not generate interrupt when triggered;</p>	16'd0

[15:0]	RW	PAD_STATUS/INT_SOURCE A bit of 1 indicates that the corresponding PAD is triggered; a bit of 0 indicates that the corresponding PAD is not triggered; write 1 to clear 0.	16'd0

28 W800 Security Architecture Design

28.1 Functional overview

The XT804 indicates the security features of bus access by means of the HPROT signal.

	0	1
HPROT [3]	uncacheable	cacheable
HPROT [2]	un-security	security
HPROT [1]	User	super
HPROT[0]	Code	data

The W800's support for security architecture is divided into security access control for srams and access control for peripherals.

28.1.1 SRAM Security Access Controller (SASC)

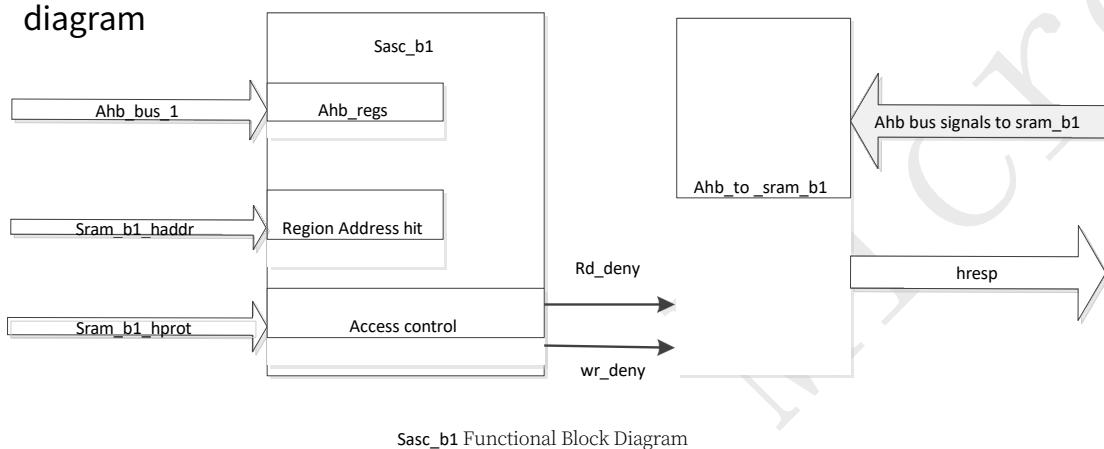
The primary bus SRAM, secondary bus SRAM, and FLASH each have a secure access controller, and each SASC-controlled memory space has the following security features

- 8 configurable security zones
- Each security zone can be configured to a minimum of 4 bytes
- Security-super allows unrestricted access to all zones
- If the CPU is denied access over the AHB bus an error answer signal (HRESP=1s) is generated, which triggers an access exception.
- No exception is thrown if access is denied to other master devices on the bus, such as DMA.

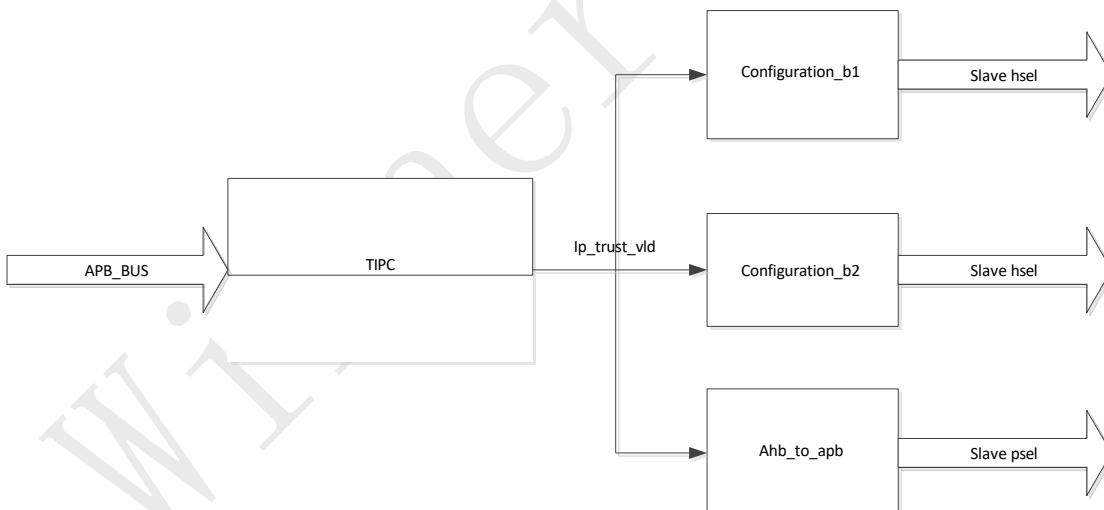
28.1.2 Trusted IP Controller (TIPC)

Hooks up to the APB bus and is used to configure trust permissions for all peripherals on the primary bus, secondary bus and APB bus. If the peripheral is configured as a trusted device (`ip_trust_vld=1`) the peripheral can only be accessed normally if access on the bus is security (HPROT[2]=1 or PPROT[2]=1), otherwise reads and writes are denied.

28.2 Security architecture block diagram



Sasc_b1 Functional Block Diagram



TIPC Functional Block Diagram

28.3 Register Description

28.3.1 SASC Register List

name (of a thing)	misal ignm ent addr ess	potent ial thresh old	clarification	reset value
CAR	0x0	[31:16]	Retention of unused	--
		[15:14]	sram region7 configuration attributes, same as region0	2'b00
		[13:12]	sram region6 configuration attributes, same as region0	2'b00
		[11:10]	sram region5 configuration attributes, same as region0	2'b00
		[9:8]	sram region4 configuration attributes, same as region0	2'b00
		[7:6]	sram region3 configuration attributes, same as region0	2'b00
		[5:4]	sram region2 configuration attributes, same as region0	2'b00
		[3:2]	sram region1 configuration attributes, same as region0	2'b00
		[1:0]	sram region0 configuration attribute 00: un-security-user 01: un-security-super 10: security-user	2'b00

			11: security-super	
CR	0x4	31:8	Unused reservations	--
		7	region7 Attribute control register, same as region0	
		6	region6 Attribute control register, same as region0	
		5	region5 Attribute control register, same as region0	
		4	region4 Attribute control register, same as region0	
		3	region3 Attribute control register, same as region0	

AP0	0x8	2	region2 Attribute control register, same as region0	
		1	region1 Attribute control register, same as region0	
		0	region0 Attribute Control Register 0: Attribute not valid 1: Attribute valid	
		[31:16]	Retention of unused	
		[15:14]	region7 AP configuration for un-security-user	
		[13:12]	region6 AP configuration for un-security-user	
		[11:10]	region5 AP configuration for un-security-user	
		[9:8]	region4 AP configuration for un-security-user	
		[7:6]	region3 AP configuration for un-security-user	
		[5:4]	region2 AP configuration for un-security-user	

	[3:2]	region1 AP configuration for un-security-user	
	[1:0]	region0 AP configuration for un-security- user 00:R/W 01:RO 10:WO 11: No	

			Access	
CD0	0xC	[31:16]	Retention of unused	
		[15:14]	region7 CD configuration for un-security-user	
		[13:12]	region6 CD configuration for un-security-user	
		[11:10]	region5 CD configuration for un-security-user	
		[9:8]	region4 CD configuration for un-security-user	
		[7:6]	region3 CD configuration for un-security-user	
		[5:4]	region2 CD configuration for un-security-user	
		[3:2]	region1 CD configuration for un-security-user	

	[1:0]	region0 CD configuration for un-security- user 00: Date Access , Opcode Feche 01: Data Access 10: Opcode Feche 11: Data, Opcode all deny	
--	-------	--	--

AP1	0x10	[31:16]	Retention of unused	
		[15:14]	region7 AP configuration for un-security-super	
		[13:12]	region6 AP configuration for un-security-super	
		[11:10]	region5 AP configuration for un-security-super	
		[9:8]	region4 AP configuration for un-security-super	
		[7:6]	region3 AP configuration for un-security-super	
		[5:4]	region2 AP configuration for un-security-super	
		[3:2]	region1 AP configuration for un-security-super	
		[1:0]	region0 AP configuration for un-security-super 00:R/W 01:RO 10:WO 11: No Access	

CD1	0x14	[31:16]	Retention of unused	
		[15:14]	region7 CD configuration for un-security-super	

		[13:12]	region6 CD configuration for un-security-super	
		[11:10]	region5 CD configuration for un-security-super	
		[9:8]	region4 CD configuration for un-security-super	
		[7:6]	region3 CD configuration for un-security-super	
		[5:4]	region2 CD configuration for un-security-super	
		[3:2]	region1 CD configuration for un-security-super	
		[1:0]	region0 CD configuration for un-security-super 00: Date Access , Opcode Feche 01: Data Access 10: Opcode Feche 11: Data, Opcode all deny	
		[31:16]	Retention of unused	

AP2	0x18	[15:14]	region7 AP configuration for security-user	
		[13:12]	region6 AP configuration for security-user	
		[11:10]	region5 AP configuration for security-user	

		[9:8]	region4 AP configuration for security-user	
		[7:6]	region3 AP configuration for security-user	
		[5:4]	region2 AP configuration for security-user	
		[3:2]	region1 AP configuration for security-user	
		[1:0]	region0 AP configuration for security-user 00:R/W 01:RO 10:WO 11: No Access	
CD2	0x1C	[31:16]	Retention of unused	
		[15:14]	region7 CD configuration for security-user	
		[13:12]	region6 CD configuration for security-user	
		[11:10]	region5 CD configuration for security-user	
		[9:8]	region4 CD configuration for security-user	
		[7:6]	region3 CD configuration for security-user	
		[5:4]	region2 CD configuration for security-user	
		[3:2]	region1 CD configuration for security-user	

		[1:0]	region0 CD configuration for security-user 00: Date Access , Opcode Feche 01: Data Access 10: Opcode Feche 11: Data, Opcode all deny	
REGION0	0x20	31:n+1	reservations	0
		n:8	BAddr0, region0 base address, n with sram large	

			Minor correlation. sram=32kB, n=20 sram=64kB, n=21	
		7:5	reservations	3'b000
		4:0	RSize, region0 size 00101: 4B 00110: 8B 10001: 16KB 10010: 32KB 	
		31:n+1	reservations	0
REGION1	0x24	n:8	Baddr1, region1 Base address	
		7:5	reservations	3'b000
		4:0	region1 size	
		31:n+1	reservations	0
REGION2	0x28	n:8	region2 base address	
		7:5	reservations	3'b000
		4:0	region2 size	
		31:n+1	reservations	0
REGION3	0x2C	n:8	region3 base address	
		7:5	reservations	3'b000

		4:0	region3 size	
REGION4	0x30	31:n+1	reservations	0
		n:8	region4 base address	
		7:5	reservations	3'b000
		4:0	region4 size	
REGION5	0x34	31:n+1	reservations	0
		n:8	region5 base address	
		7:5	reservations	3'b000
		4:0	region5 size	
REGION6	0x38	31:n+1	reservations	0
		n:8	region6 Base address	
		7:5	reservations	3'b000
		4:0	region6 size	
REGION7	0x3C	31:n+1	reservations	0
		n:8	region7 Base address	
		7:5	reservations	3'b000
		4:0	region7 size	

28.3.2 TIPC Register

name (of a thing)	misalignm ent address	potenti al threshold	clarification	reset value
ip_trust_vld0	0x0	31:18	unexpended	0
		17	BT modem Trusted Attributes 1: Credible 0: not credible	0
		16	I2S Trusted Attributes	0
		15	PWM Trusted Properties	0
		14	LCD driver Trusted attributes	0
		13	RF controller Trusted Attributes	0
		12	timer Trusted Attributes	0
		11	watch dog Trusted Properties	0
		10	PORTB Trusted Attributes	0
		9	PORTA Trusted Properties	0
		8	UART5 Trusted Attributes	0
		7	UART4 Trusted Attributes	0
		6	UART3 Trusted Attributes	0
		5	UART2 Trusted Attributes	0

	4	UART1 Trusted Attribute	0
	3	UART0 Trusted Attribute	0

		2	SPI MASTER Trusted Attributes	0
		1	SAR ADC Trusted Attributes	0
		0	I2C Trusted Attributes	0
ip_trust_vld1	0x4	31:18	unexpended	0
		17	RF BIST Trusted Attribute Configuration 1: Credible 0: not credible	0
		16	SDIO Wrapper Trusted Attributes	0
		15	SPI_HS Trusted Attributes	0
		14	SDIO Trusted Attributes	0
		13	unexpended	0
		12	SEC Trusted Properties	0
		11	MAC Trusted Attributes	0
		10	BBP Trusted Properties	0
		9	MMU Trusted Attributes	0
		8	Clock Reset Control Module Trusted Attributes	0
		7	PMU Trusted Properties	0
		6	BT Trusted Properties	0
		5	GPSEC Trusted Properties	0
		4	DMA Trusted Attributes	0
		3	RSA Trusted Attributes	0

	2	PSRAM Controller Trusted Attributes	0
	1	FLASH Controller Trusted Attributes	0

	0	SDIO HOST Trusted Attributes	0
--	---	------------------------------	---

28.4 Instructions for use

28.4.1 Safe Access to Memory (SASC)

28.4.1.1 Register access rights

SASC registers are accessed according to the following principles:

- CAR registers can only be read and written when the cpu is security-super.
- When the cpu is in un-security-super, it can access the register bits related to the region configured as un-security-user.
- When cpu is security-super, all registers can be read and written.
- In all other cases, the registers are inaccessible.

For example, when CAR is set to 16'he4e4, it means that region0 and region4 are set to un-security-user, and if cpu

is un-security-super, then the cpu can read and write registers REGION0 and REGION4, as well as CR[0], CR[4], APx[1:0], the

APx[9:8], CDx[1:0], CDx[9:8].

28.4.1.2 Protecting the inter-area address setting

Each SASC supports 8 configurable memory reserve regions. The base address in REGIONx[31:8] is 25 to 2 bits of the actual physical address of the memory reserve that you want to configure, and the size of the Size and the base address need to meet the following requirements:

SIZE

00101: 4B.

00110: 8B; Baddrx[0]=0
00111: 16B; Baddrx[1:0]=0
01000: 32B. Baddrx[2:0]=0
01001: 64B. Baddrx[3:0]=0
01010: 128B. Baddrx[4:0]=0
01011: 256B; Baddrx[5:0]=0
01100: 512B;. Baddrx[6:0]=0
01101:1KB; Baddrx[7:0]=0
01110: 2KB;. Baddrx[8:0]=0
01111:4KB; Baddrx[9:0]=0
10000: 8KB. Baddrx[10:0]=0
10001: 16KB; Baddrx[11:0]=0

For example, if 20000100 is used as the base address of region0, then region0 can be set to a maximum of 256B, and if you want to set region0 to 128B, REGION0 register should be filled with 0x0000400a. If 20040000 is used as the base address, the region can be set to a maximum of 64KB, and if you want to define a 64KB region, then this register should be filled with 0x01000013. If 20840000 Memory access rights be defined, the register should be 0x01000013.

— Priority of the four authority for mem:

request mem	Security super	Security user	Un-Security super	Un-Security user
Security super	✓ +	✓ +	✓ +	✓ +
Security user	X	✓	X	X
Un-Security super	X	X	✓	✓ +
Un-Security user	X	X	X	✓

Note: ✓ + all access, include read, write, data access, opcode fetch

✓ access based on the current attribute

X no access

Accordin

g to the
chart
above

- Bus access to security-super allows arbitrary access to the sram
- Bus access for un-security-super can access the region configured as un-security-user at will.
- Bus access for a security-user can only access the security-user's region by the current AP and CD attributes
- Bus access for un-security-super allows access to un-security-super's region by current AP and CD attributes
- Bus access for un-security-user can only access the un-security-user's region by the current AP and CD attributes

The APx and CDx registers are used to set the access attributes of each region corresponding to different privileges, where CDx is sent to the

The register is only valid for read operations during bus accesses, i.e., it sets whether read data and read instructions are allowed. The APx register is used to set whether read/write operations are allowed.

For example, if CAR[1:0] is set to 00 and CR[0] is 1, it means that REGION 0 is un-security-user and enabled for privilege protection, at this time, if the bus access is security-super or un-security-super, you can access REGION 0 at will; if the bus access is Security-user, any access will be denied; if the bus access is un-security-user and AP0[1:0] is 01 for read-only and CD0[1:0] is 01 for data access, REGION 0 allows the bus to read the data and all other operations are denied.

If the AHB bus accesses an inaccessible area or has incorrect privileges, the sasc module gives a read deny or write deny signal, which returns an incorrect HRESP response signal and generates a hardware exception interrupt if the CPU is the one initiating the bus access, or just denies access if it is another device.

28.4.2 Trusted access to peripherals

The TIPC module can only be written when the PROT[2] signal is 1, i.e., the trusted attribute configuration registers of each IP can only be written in the trusted world. ip_trust_vld register is 0 by default, i.e., all the peripherals are untrusted, and at this time, the peripherals can be accessed at will, and if the corresponding ip_trust_vld of a peripheral is set, i.e., setting the peripheral as a trusted device, then only commands from

the trusted world can access the peripheral. If the corresponding ip_trust_vld of the peripheral is set, that is, the peripheral is set as a trusted device, only commands from the trusted world can access the peripheral.

28.4.2 Trusted access to peripherals

29 Appendix 1. Chip Pin Definitions

29.1 Chip Pinout

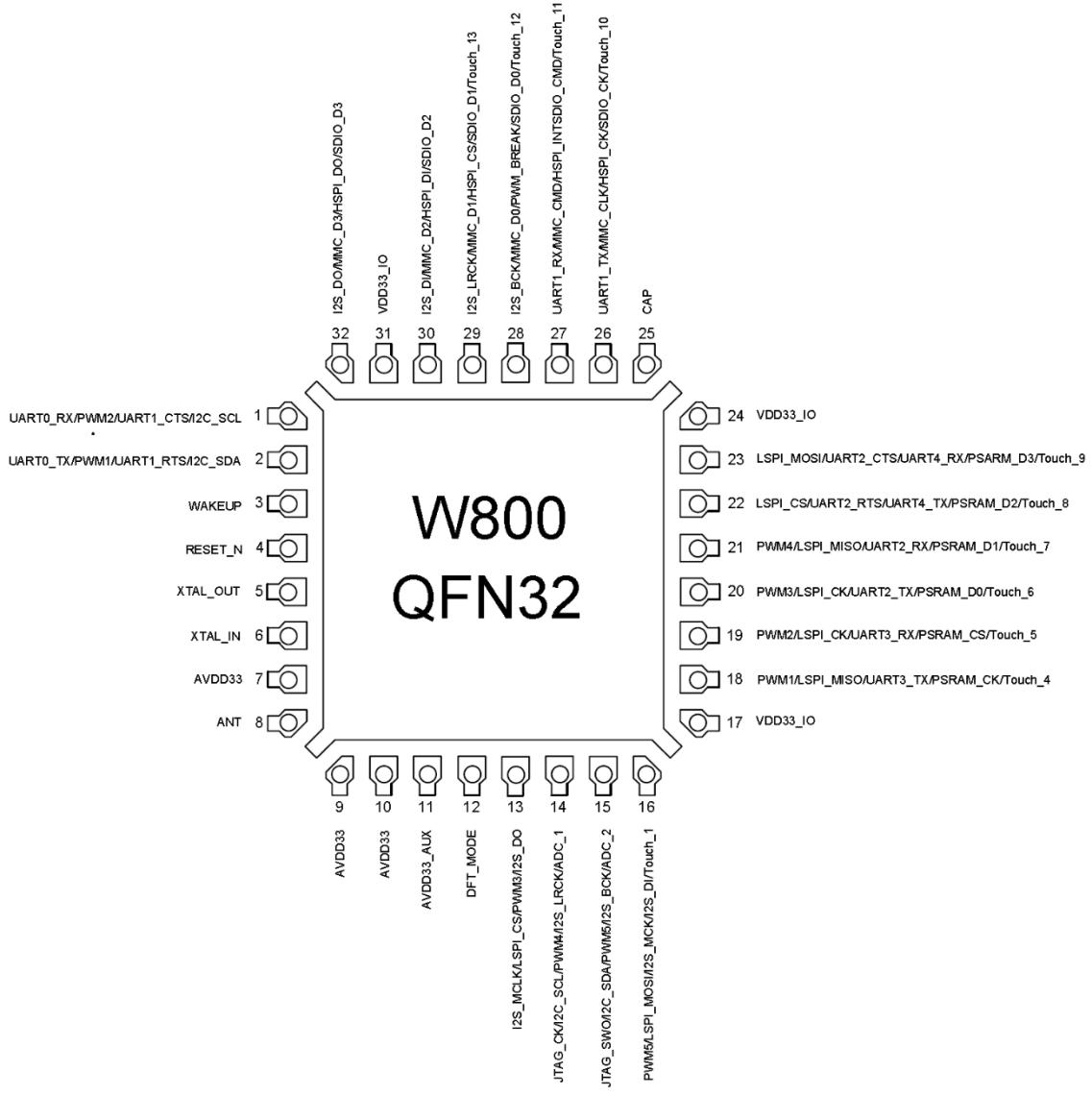


Figure 38 W800 Chip Pinout

Winner Micro

29.2 Chip pin multiplexing relationships

Table 220 Chip Pin-Multiplexing Relationships

serial numb er	name (of a thing)	typology	Pin function after reset	multiplexing function	maximum frequency	Ability to pull up and down	driving ability
1	PB_20	I/O	UART_RX	UART0_RX/PWM2/UART1_CTS/I2C_SCL	10MHz	UP/DOWN	12mA
2	PB_19	I/O	UART_TX	UART0_TX/PWM1/UART1_RTS/I2C_SDA	10MHz	UP/DOWN	12mA
3	WAKEUP	I	WAKEUP Wake-up function			DOWN	
4	RESET	I	RESET reset			UP	
5	XTAL_OUT	O	External Crystal Output				
6	XTAL_IN	I	External crystal input				
7	AVDD33	P	Chip Power Supply, 3.3V				
8	ANT	I/O	radio frequency antenna				
9	AVDD33	P	Chip Power Supply, 3.3V				
10	AVDD33	P	Chip Power Supply, 3.3V				
11	AVDD33_AUX	P	Chip Power Supply, 3.3V				
12	TEST	I	Test Function Configuration Pins				
13	BOOTMODE	I/O	BOOTMODE	I2S_MCLK/LSPI_CS/PWM3/I2S_DO	20MHz	UP/DOWN	12mA
14	PA_1	I/O	JTAG_CK	JTAG_CK/I2C_SCL/PWM4/I2S_LRCK/ADC_1	20MHz	UP/DOWN	12mA

15	PA_4	I/O	JTAG_SWO	jtag_swo/i2c_sda/pwm5/i2s_bck/adc_2	20MHz	UP/DOWN	12mA
16	PA_7	I/O	GPIO, Input, High Resistance	PWM5/LSPI_MOSI/I2S_MCK/I2S_DI/Touch_1	20MHz	UP/DOWN	12mA
17	VDD33IO	P	IO Power Supply, 3.3V				
18	PB_0	I/O	GPIO, Input, High Resistance	PWM1/LSPI_MISO/UART3_TX/PSRAM_CK/Touch_4	80MHz	UP/DOWN	12mA
19	PB_1	I/O	GPIO, Input, High Resistance	PWM2/LSPI_CK/UART3_RX/PSRAM_CS/Touch_5	80MHz	UP/DOWN	12mA
20	PB_2	I/O	GPIO, Input, High Resistance	PWM3/LSPI_CK/UART2_TX/PSRAM_D0/Touch_6	80MHz	UP/DOWN	12mA
21	PB_3	I/O	GPIO, Input, High Resistance	PWM4/LSPI_MISO/UART2_RX/PSRAM_D1/Touch_7	80MHz	UP/DOWN	12mA
22	PB_4	I/O	GPIO, Input, High Resistance	lspi_cs/uart2_rts/uart4_tx/psram_d2/touch_8	80MHz	UP/DOWN	12mA
23	PB_5	I/O	GPIO, Input, High Resistance	LSPI_MOSI/UART2_CTS/UART4_RX/PSARM_D3/Tou ch_9	80MHz	UP/DOWN	12mA
24	VDD33IO	P	IO Power Supply, 3.3V				
25	CAP	I	External Capacitance, 1μF	UART1_TX/MMC_CLK/HSPI_CK/SDIO_CK/Touch_10	50MHz	UP/DOWN	12mA
26	PB_6	I/O	GPIO, Input, High Resistance				
27	PB_7	I/O	GPIO, Input, High Resistance	UART1_RX/MMC_CMD/HSPI_INT/SDIO_CMD/Touch_11	50MHz	UP/DOWN	12mA
28	PB_8	I/O	GPIO, Input, High Resistance	2S_BCK/MMC_D0/PWM_BREAK/SDIO_D0/Touch_12	50MHz	UP/DOWN	12mA
29	PB_9	I/O	GPIO, Input, High Resistance	I2S_LRCK/MMC_D1/HSPI_CS/SDIO_D1/Touch_13	50MHz	UP/DOWN	12mA
30	PB_10	I/O	GPIO, Input, High Resistance	I2S_DI/MMC_D2/HSPI_DI/SDIO_D2	50MHz	UP/DOWN	12mA
31	VDD33IO	P	IO Power Supply, 3.3V				
32	PB_11	I/O	GPIO, Input, High Resistance	2S_DO/MMC_D3/HSPI_DO/SDIO_D3	50MHz	UP/DOWN	12mA

33	GND	P	grounding				
----	-----	---	-----------	--	--	--	--

herald

Ltd. reserves the right to modify or update the products of UniSolider Microelectronics or various documents and information at any time. All updates will be released through the official channels of Unimicron. Users must ensure that the correct information is obtained through official channels. Unimicron does not undertake to notify everyone of updates.