

UM2080F32 用户手册

版本：V1.2



广芯微电子（广州）股份有限公司

<http://www.unicmicro.com/>

条款协议

本文档的所有部分，其著作权归广芯微电子（广州）股份有限公司（以下简称广芯微电子）所有，未经广芯微电子授权许可，任何个人及组织不得复制、转载、仿制本文档的全部或部分组件。本文档没有任何形式的担保、立场表达或其他暗示，若有任何因本文档或其中提及的产品所有资讯所引起的直接或间接损失，广芯微电子及所属员工恕不为其担保任何责任。除此以外，本文档所提到的产品规格及资讯仅供参考，内容亦会随时更新，恕不另行通知。

1. 本文档中所记载的关于电路、软件和其他相关信息仅用于说明半导体产品的操作和应用实例。
用户如在设备设计中应用本文档中的电路、软件和相关信息，请自行负责。对于用户或第三方因使用上述电路、软件或信息而遭受的任何损失，广芯微电子不承担任何责任。
2. 在准备本文档所记载的信息的过程中，广芯微电子已尽量做到合理注意，但是，广芯微电子并不保证这些信息都是准确无误的。用户因本文档中所记载的信息的错误或遗漏而遭受的任何损失，广芯微电子不承担任何责任。
3. 对于因使用本文档中的广芯微电子产品或技术信息而造成的侵权行为或因此而侵犯第三方的专利、版权或其他知识产权的行为，广芯微电子不承担任何责任。本文档所记载的内容不应视为对广芯微电子或其他人所有的专利、版权或其他知识产权作出任何明示、默示或其它方式的许可及授权。
4. 使用本文档中记载的广芯微电子产品时，应在广芯微电子指定的范围内，特别是在最大额定值、电源工作电压范围、热辐射特性、安装条件以及其他产品特性的范围内使用。对于在上述指定范围之外使用广芯微电子产品而产生的故障或损失，广芯微电子不承担任何责任。
5. 虽然广芯微电子一直致力于提高广芯微电子产品的质量和可靠性，但是，半导体产品有其自身的具体特性，如一定的故障发生率以及在某些使用条件下会发生故障等。此外，广芯微电子产品均未进行防辐射设计。所以请采取安全保护措施，以避免当广芯微电子产品在发生故障而造成火灾时导致人身事故、伤害或损害的事故。例如进行软硬件安全设计（包括但不限于冗余设计、防火控制以及故障预防等）、适当的老化处理或其他适当的措施等。

目录

1	系统概述	1
1.1	功能框图	2
1.2	电源框图	3
2	引脚定义和描述	4
2.1	引脚分布	4
2.2	引脚复用	6
2.3	引脚描述	8
3	处理器	13
3.1	概述	13
3.2	主要特性	13
3.3	功能框图	14
3.4	内核寄存器组	14
4	系统配置 (SCU)	15
4.1	地址映射	15
4.2	时钟框图	16
4.3	时钟选择	17
4.4	复位源	17
4.4.1	内部 POR 上电复位	18
4.4.2	LVR 复位	18
4.4.3	RESETN 复位	18
4.4.4	WDT 复位	18
4.4.5	WWDT 复位	18
4.4.6	SOFT_RESETN 复位	18
4.4.7	模块软件复位	18
4.4.8	LOCKUP 复位	19
4.4.9	LVD 复位	19
4.5	低功耗模式	19
4.5.1	Sleep 模式	21
4.5.2	Deepsleep 模式	21
4.5.3	Stop 模式	22
4.6	系统寄存器	22
4.6.1	系统控制寄存器 0/ SYSCTRL0 (偏移: 000H)	24
4.6.2	系统控制保护寄存器/ SYSCTRL_PROTECT (偏移: 008H)	27
4.6.3	时钟控制寄存器/ OSC_CTRL (偏移: 00CH)	27
4.6.4	外围模块时钟寄存器/ PERI_CLKEN (偏移: 010H)	28

4.6.5	复位标识寄存器/ RESET_FLAG (偏移: 020H)	30
4.6.6	外围模块复位控制寄存器/ PERI_RESET (偏移: 024H)	31
4.6.7	外部复位滤波控制寄存器/ EXT_RESET_CTRL (偏移: 028H)	34
4.6.8	端口 PA 功能配置寄存器/ PA_SEL (偏移: 030H)	34
4.6.9	端口 PB 功能配置寄存器/ PB_SEL (偏移: 034H)	35
4.6.10	端口 PC 功能配置寄存器/ PC_SEL (偏移: 038H)	36
4.6.11	端口 PD 功能配置寄存器/ PD_SEL (偏移: 03CH)	38
4.6.12	端口 PE 功能配置寄存器/ PE_SEL (偏移: 040H)	39
4.6.13	端口数模配置寄存器/ PAD_ADS (偏移: 044H)	40
4.6.14	端口驱动能力配置寄存器 0/ PAD_DR0 (偏移: 048H)	41
4.6.15	端口驱动能力配置寄存器 1/ PAD_DR1 (偏移: 04CH)	43
4.6.16	端口上拉配置寄存器 0/ PAD_PU0 (偏移: 050H)	44
4.6.17	端口上拉配置寄存器 1/ PAD_PU1 (偏移: 054H)	46
4.6.18	端口下拉配置寄存器 0/ PAD_PD0 (偏移: 058H)	46
4.6.19	端口下拉配置寄存器 1/ PAD_PD1 (偏移: 05CH)	48
4.6.20	端口开漏输出配置寄存器 0/ PAD_OD0 (偏移: 060H)	48
4.6.21	端口开漏输出配置寄存器 1/ PAD_OD1 (偏移: 064H)	50
4.6.22	端口输入类型配置寄存器 0/ PAD_CS0 (偏移: 068H)	50
4.6.23	端口输入类型配置寄存器 1/ PAD_CS1 (偏移: 06CH)	52
4.6.24	端口输入配置寄存器 0/ PAD_IE0 (偏移: 070H)	52
4.6.25	端口输入配置寄存器 1/ PAD_IE1 (偏移: 074H)	54
4.6.26	端口速度配置寄存器 0/ PAD_SR0 (偏移: 078H)	55
4.6.27	端口速度配置寄存器 1/ PAD_SR1 (偏移: 07CH)	57
4.6.28	IO 控制保护寄存器/ IOCTRL_PROTECT (偏移: 080H)	57
4.6.29	LVD 配置寄存器/ LVD_CFG (偏移: 084H)	57
4.6.30	外部复位端口选择寄存器/ EXTRST_SEL (偏移: 090H)	59
4.6.31	停止模式选择寄存器/ STOPMODE_SEL (偏移: 094H)	59
4.6.32	REMAP 寄存器/ REMAP_ADDR (偏移: 098H)	60
4.6.33	中断向量地址重映射寄存器/ VECTOR_OFFSET (偏移: 09CH)	60
4.6.34	随机数控制寄存器/ HRNG_CR (偏移: 0A0H)	60
4.6.35	随机数种子寄存器/ HRNG_SEED (偏移: 0A4H)	61
4.6.36	随机数数据寄存器 / HRNG_DATA (偏移: 0A8H)	61
4.6.37	LVR 控制寄存器/ LVR_CFG (偏移: 0B0H)	61
4.6.38	VREF 控制寄存器/ VREF_CFG (偏移: 0B4H)	62
4.6.39	内部基准状态寄存器/ VREF_STATUS (偏移: 0CCH)	63
5	RF (Sub-1GHz 收发器)	64
5.1	概述	64
5.2	主要特性	64

5.3	寄存器描述.....	65
5.3.1	Reg00 Address: 0x00 Default: 0x1D	65
5.3.2	Reg01 Address: 0x01 Default: 0x78.....	65
5.3.3	Reg02 Address: 0x02 Default: 0x00.....	65
5.3.4	Reg03 Address: 0x03 Default: 0x00.....	65
5.3.5	Reg04 Address: 0x04 Default: 0x00.....	66
5.3.6	Reg05 Address: 0x05 Default: 0x03.....	66
5.3.7	Reg06 Address: 0x06 Default: 0x33.....	66
5.3.8	Reg07 Address: 0x07 Default: 0x33.....	66
5.3.9	Reg08 Address: 0x08 Default: 0x10.....	66
5.3.10	Reg09 Address: 0x09 Default: 0x00.....	67
5.3.11	Reg0A Address: 0x0A Default: 0x00	67
5.3.12	Reg0B Address: 0x0B Default: 0x00	67
5.3.13	Reg0C Address: 0x0C Default: 0x50	67
5.3.14	Reg0D Address: 0x0D Default: 0x30.....	67
5.3.15	Reg0E Address: 0x0E Default: 0x32.....	68
5.3.16	Reg10 Address: 0x10 Default: 0x40.....	69
5.3.17	Reg11 Address: 0x11 Default: 0x00.....	69
5.3.18	Reg1D Address: 0x1D Default: 0x80.....	70
5.3.19	Reg20 Address: 0x20 Default: 0x50.....	70
5.3.20	Reg21 Address: 0x21 Default: 0x60.....	71
5.3.21	Reg27 Address: 0x27 Default: 0x60.....	71
5.3.22	Reg28 Address: 0x28 Default: 0x19.....	71
5.3.23	Reg2B Address: 0x2B Default: 0xFF	72
5.3.24	Reg2C Address: 0x2C Default: 0xFF	72
5.3.25	Reg2D Address: 0x2D Default: 0x10.....	72
5.3.26	Reg30 Address: 0x30 Default: 0x80.....	73
5.3.27	Reg31 Address: 0x31 Default: 0x66.....	73
5.3.28	Reg37 Address: 0x37 Default: 0x06.....	73
5.3.29	Reg38 Address: 0x38 Default: 0x91.....	73
5.3.30	Reg4D Address: 0x4D Default: 0x10.....	74
5.3.31	Reg4E Address: 0x4E Default: 0x45.....	74
5.3.32	Reg4F Address: 0x4F Default: 0x67	75
5.3.33	Reg50 Address: 0x50 Default: 0x00.....	75
5.3.34	Reg51 Address: 0x51 Default: 0x00.....	75
5.3.35	Reg52 Address: 0x52 Default: 0x00.....	75
5.3.36	Reg53 Address: 0x53 Default: 0x00.....	75
5.3.37	Reg54 Address: 0x54 Default: 0x00.....	75

5.3.38	Reg55	Address: 0x55	Default: 0x00	75
5.3.39	Reg56	Address: 0x56	Default: 0xB7	76
5.3.40	Reg57	Address: 0x57	Default: 0x00	76
5.3.41	Reg58	Address: 0x58	Default: 0x00	76
5.3.42	Reg5D	Address: 0x5D	Default: 0x00	76
5.3.43	Reg60	Address: 0x60	Default: 0x80	77
5.3.44	Reg61	Address: 0x61	Default: 0x00	77
5.3.45	Reg64	Address: 0x64	Default: 0x00	77
5.3.46	Reg67	Address: 0x67	Default: 0x80	77
5.3.47	Reg68	Address: 0x68	Default: 0x00	78
5.3.48	Reg69	Address: 0x69	Default: 0x00	78
5.3.49	Reg6A	Address: 0x6A	Default: 0x00	78
5.3.50	Reg6B	Address: 0x6B	Default: 0x00	78
5.3.51	Reg6C	Address: 0x6C	Default: 0x00	78
5.3.52	Reg6D	Address: 0x6D	Default: 0x00	78
5.3.53	Reg6E	Address: 0x6E	Default: 0x00	79
5.3.54	Reg6F	Address: 0x6F	Default: 0x00	79
5.3.55	Reg70	Address: 0x70	Default: 0xFF	79
5.3.56	Reg71	Address: 0x71	Default: 0x09	79
5.3.57	Reg72	Address: 0x72	Default: 0x01	79
5.3.58	Reg73	Address: 0x73	Default: 0x55	80
5.3.59	Reg74	Address: 0x74	Default: 0x03	80
5.3.60	Reg75	Address: 0x75	Default: 0xA7	81
5.3.61	Reg76	Address: 0x76	Default: 0xA7	81
5.3.62	Reg77	Address: 0x77	Default: 0x98	81
5.3.63	Reg78	Address: 0x78	Default: 0xF3	81
5.3.64	Reg79	Address: 0x79	Default: 0x98	81
5.3.65	Reg7A	Address: 0x7A	Default: 0xF3	82
5.3.66	Reg7B	Address: 0x7B	Default: 0x98	82
5.3.67	Reg7C	Address: 0x7C	Default: 0xF3	82
5.3.68	Reg7D	Address: 0x7D	Default: 0x0E	82
5.3.69	Reg7E	Address: 0x7E	Default: 0x00	82
5.3.70	Reg7F	Address: 0x7F	Default: 0x00	82
5.3.71	Reg80	Address: 0x80	Default: 0x1D	83
5.3.72	Reg81	Address: 0x81	Default: 0x78	83
5.3.73	Reg82	Address: 0x82	Default: 0x00	84
5.3.74	Reg83	Address: 0x83	Default: 0x00	84
5.3.75	Reg84	Address: 0x84	Default: 0x00	84

5.3.76 Reg85 Address: 0x85 Default: 0x03.....84

5.3.77 Reg86 Address: 0x86 Default: 0x33.....84

5.3.78 Reg87 Address: 0x87 Default: 0x33.....85

5.3.79 Reg88 Address: 0x88 Default: 0x10.....85

5.3.80 Reg89 Address: 0x89 Default: 0x00.....85

5.3.81 Reg8A Address: 0x8A Default: 0x0085

5.3.82 Reg8B Address: 0x8B Default: 0x0086

5.3.83 Reg8C Address: 0x8C Default: 0x5086

5.3.84 Reg8D Address: 0x8D Default: 0x30.....86

5.3.85 Reg8E Address: 0x8E Default: 0x32.....86

5.3.86 Reg94 Address: 0x94 Default: 0x08.....86

5.3.87 Reg95 Address: 0x95 Default: 0x20.....87

5.3.88 Reg96 Address: 0x96 Default: 0x26.....87

5.3.89 Reg97 Address: 0x97 Default: 0x1F.....87

5.3.90 Reg98 Address: 0x98 Default: 0x68.....87

5.3.91 Reg99 Address: 0x99 Default: 0x04.....87

5.3.92 Reg9A Address: 0x9A Default: 0xAE.....88

5.3.93 Reg9B Address: 0x9B Default: 0x89.....88

5.3.94 Reg9C Address: 0x9C Default: 0x8488

5.3.95 Reg9D Address: 0x9D Default: 0x03.....88

5.3.96 Reg9E Address: 0x9E Default: 0x80.....88

5.3.97 Reg9F Address: 0x9F Default: 0x26.....89

5.3.98 RegA0 Address: 0xA0 Default: 0x5089

5.3.99 RegA3 Address: 0xA3 Default: 0x4289

5.3.100 RegE0 Address: 0xE0 Default: 0x8089

5.3.101 RegE1 Address: 0xE1 Default: 0x0089

5.3.102 RegE2 Address: 0xE2 Default: 0x0190

5.3.103 RegE3 Address: 0xE3 Default: 0x4090

5.4 功能描述90

5.4.1 接收机.....90

5.4.2 发射机.....90

5.4.3 系统复位91

5.4.4 频率综合器92

5.4.5 晶体振荡器94

5.4.6 AGC95

5.4.7 RSSI.....95

5.4.8 AFC.....98

5.5 运行状态99

5.5.1	SPI 接口	99
5.5.2	FIFO.....	100
5.5.3	GPIO 和中断.....	103
5.5.4	启动时序和状态机.....	105
5.6	数据包及包处理机制	107
5.6.1	直通模式 (Direct)	107
5.6.2	数据包 (Packet)	109
6	EFC	130
6.1	概述.....	130
6.2	主要特性	130
6.3	寄存器描述.....	130
6.3.1	控制寄存器 EFC_CTRL (偏移: 0000H).....	131
6.3.2	写擦安全寄存器 EFC_SEC (偏移: 0004H).....	131
6.3.3	状态寄存器 EFC_STATUS (偏移: 0008H).....	132
6.3.4	中断状态寄存器 EFC_INTSTATUS (偏移: 000CH).....	132
6.3.5	中断使能寄存器 EFC_INEN (偏移: 0010H).....	133
6.3.6	时间标尺寄存器 EFC_HALFUS (偏移: 0014H).....	134
6.3.7	RCH TRIM 寄存器 EFC_RCHTRIM (偏移: 0020H).....	134
6.3.8	RCL TRIM 寄存器 EFC_RCLTRIM (偏移: 0024H).....	134
6.4	功能描述	135
6.4.1	地址映射	135
6.4.2	自动锁总线	135
6.4.3	EFC 上电预读	135
6.4.4	eFlash 读效率.....	136
6.5	软件流程	136
6.5.1	Read 操作.....	136
6.5.2	Write 操作	136
6.5.3	Erase 操作	137
7	NVIC	138
7.1	概述.....	138
7.2	主要特性	138
7.3	中断源.....	138
8	UART0.....	140
8.1	概述.....	140
8.2	主要特性	140
8.3	寄存器描述.....	141
8.3.1	中断状态寄存器 UART_ISR (偏移: 00H).....	141

8.3.2	中断使能寄存器 UART_IER (偏移: 04H)	142
8.3.3	控制寄存器 UART_CR (偏移: 08H)	142
8.3.4	发送数据寄存器 UART_TDR (偏移: 0CH)	143
8.3.5	接收数据寄存器 UART_RDR (偏移: 0CH)	143
8.3.6	波特率参数低位寄存器 UART_BRPL (偏移: 10H)	143
8.3.7	波特率参数高位寄存器 UART_BRPH (偏移: 14H)	144
8.4	使用流程	144
8.4.1	串口的发送和接收	144
8.4.2	串口初始化	145
8.4.3	串口发送字节	145
8.4.4	串口接收字节	145
9	UART1	147
9.1	概述	147
9.2	主要特性	147
9.3	寄存器描述	147
9.3.1	接收缓冲寄存器 UART1_RBR (偏移: 00H)	148
9.3.2	发送缓冲寄存器 UART1_THR (偏移: 00H)	148
9.3.3	波特率分频低位寄存器 UART1_DLL (偏移: 00H)	148
9.3.4	波特率分频高位寄存器 UART1_DLH (偏移: 04H)	149
9.3.5	中断使能寄存器 UART1_IER (偏移: 04H)	149
9.3.6	中断状态寄存器 UART1_IIR (偏移: 08H)	150
9.3.7	FIFO 控制寄存器 UART1_FCR (偏移: 08H)	150
9.3.8	LINE 控制寄存器 UART1_LCR (偏移: 0CH)	151
9.3.9	流控制寄存器 UART1_MCR (偏移: 10H)	152
9.3.10	LINE 中断状态寄存器 UART1_LSR (偏移: 14H)	153
9.3.11	流状态寄存器 UART1_MSR (偏移: 18H)	154
9.3.12	状态寄存器 UART1_USR (偏移: 7CH)	154
9.3.13	发送 FIFO 数据个数寄存器 UART1_TFL (偏移: 80H)	154
9.3.14	接收 FIFO 数据个数寄存器 UART1_RFL (偏移: 84H)	155
9.3.15	小数分频寄存器 UART1_DLF (偏移: C0H)	155
9.3.16	接收地址匹配寄存器 UART1_RAR (偏移: C4H)	155
9.3.17	发送地址匹配寄存器 UART1_TAR (偏移: C8H)	155
9.4	使用流程	156
9.4.1	UART1 发送流程	156
9.4.2	UART1 接收流程	156
9.4.3	CTS 和 RTS 控制流功能设置流程	157
9.4.4	UART1 DMA 传输配置流程	157

10	I2C	159
10.1	概述	159
10.2	主要特性	159
10.3	寄存器描述	159
10.3.1	I2C 配置寄存器 I2C_CR (偏移: 00H)	160
10.3.2	I2C 配置清除寄存器 I2C_CLR (偏移: 04H)	161
10.3.3	I2C 状态寄存器 I2C_STAT (偏移: 08H)	162
10.3.4	I2C 数据寄存器 I2C_DATA (偏移: 0CH)	163
10.3.5	I2C 波特率配置寄存器 I2C_CCR (偏移: 10H)	163
10.3.6	I2C SLAVE 地址寄存器 0 I2C_SAD0 (偏移: 14H)	164
10.3.7	I2C SLAVE 地址屏蔽寄存器 0 I2C_SADM0 (偏移: 18H)	164
10.3.8	10 比特 I2C SLAVE 地址寄存器 I2C_XSAD (偏移: 1CH)	164
10.3.9	10 比特 I2C SLAVE 地址屏蔽寄存器 I2C_XSADM (偏移: 20H)	164
10.3.10	I2C 复位寄存器 I2C_SRST (偏移: 24H)	165
10.3.11	I2C SLAVE 地址寄存器 1 I2C_SAD1 (偏移: 28H)	165
10.3.12	I2C SLAVE 地址屏蔽寄存器 1 I2C_SADM1 (偏移: 2CH)	165
10.3.13	I2C SLAVE 地址寄存器 2 I2C_SAD2 (偏移: 30H)	166
10.3.14	I2C SLAVE 地址屏蔽寄存器 2 I2C_SADM2 (偏移: 34H)	166
10.3.15	I2C SLAVE 地址寄存器 2 I2C_SAD3 (偏移: 38H)	166
10.3.16	I2C SLAVE 地址屏蔽寄存器 3 I2C_SADM3 (偏移: 3CH)	166
10.4	协议描述	167
10.4.1	I2C 通信协议 (7 位寻址)	167
10.4.2	I2C 通信协议 (10 位寻址)	168
10.5	使用流程	170
10.5.1	初始化程序	170
10.5.2	主机发送功能	170
10.5.3	主机接收功能	171
10.5.4	从机接收功能	172
10.5.5	从机发送功能	173
11	SPI0	176
11.1	概述	176
11.2	主要特性	176
11.3	寄存器描述	176
11.3.1	SPI0 配置寄存器 SPI0_CR (偏移: 00H)	177
11.3.2	SPI0 主模式控制寄存器 0 SPI0_CS0 (偏移: 04H)	178
11.3.3	SPI0 主模式控制寄存器 1 SPI0_CS1 (偏移: 08H)	179
11.3.4	SPI0 过程控制寄存器 SPI0_OPCR (偏移: 14H)	180
11.3.5	SPI0 中断控制寄存器 SPI0_IE (偏移: 18H)	181

11.3.6	SPI0 中断标志寄存器 SPI0_IF (偏移: 1CH)	182
11.3.7	SPI0 发送缓存寄存器 SPI0_TXBUF (偏移: 20H)	182
11.3.8	SPI0 接收缓存寄存器 SPI0_RXBUF (偏移: 24H)	183
11.3.9	SPI0 DMA 接收设置寄存器 SPI0_DMARXLEV (偏移: 28H)	183
11.3.10	SPI0 DMA 发送设置寄存器 SPI0_DMATXLEV (偏移: 2CH)	183
11.4	接口时序	183
11.4.1	CPHA=0	184
11.4.2	CPHA=1	184
11.4.3	从器件 SSN	185
11.5	使用流程	186
11.5.1	初始化程序	186
11.5.2	发送流程	187
11.5.3	接收流程	187
11.5.4	SPI0 DMA 发送流程	187
11.5.5	SPI0 DMA 接收流程	188
12	SPI1	189
12.1	概述	189
12.2	主要特性	189
12.3	寄存器描述	189
12.3.1	SPI1 配置寄存器 SPI1_CR (偏移: 00H)	190
12.3.2	SPI1 主模式控制寄存器 0 SPI1_CS0 (偏移: 04H)	191
12.3.3	SPI1 过程控制寄存器 SPI1_OPCR (偏移: 14H)	192
12.3.4	SPI1 中断控制寄存器 SPI1_IE (偏移: 18H)	193
12.3.5	SPI1 中断标志寄存器 SPI1_IF (偏移: 1CH)	193
12.3.6	SPI1 发送缓存寄存器 SPI1_TXBUF (偏移: 20H)	194
12.3.7	SPI1 接收缓存寄存器 SPI1_RXBUF (偏移: 24H)	194
12.3.8	SPI1 DMA 接收设置寄存器 SPI1_DMARXLEV (偏移: 28H)	194
12.3.9	SPI1 DMA 发送设置寄存器 SPI1_DMATXLEV (偏移: 2CH)	194
12.4	接口时序	195
12.4.1	CPHA=0	195
12.4.2	CPHA=1	195
12.4.3	从器件 SSN	196
12.5	使用流程	197
12.5.1	初始化程序	197
12.5.2	发送流程	197
12.5.3	接收流程	198
12.5.4	SPI1 DMA 发送流程	198
12.5.5	SPI1 DMA 接收流程	199

13	QSPI.....	200
13.1	概述.....	200
13.2	主要特性.....	200
13.3	寄存器描述.....	200
13.3.1	配置寄存器 SPI_CTRL (偏移: 00H)	201
13.3.2	SPI 波特率配置寄存器 SPI_BAUD (偏移: 04H)	201
13.3.3	SPI 存储器访问配置寄存器 SPI_MEMO_ACC (偏移: 08H)	202
13.3.4	SPI 命令寄存器 SPI_CMD (偏移: 0Ch)	204
13.3.5	SPI 读参数寄存器 SPI_PARA_R (偏移: 10H)	204
13.3.6	SPI 写参数寄存器 SPI_PARA_W (偏移: 14H)	204
13.3.7	SPI 擦写时间设置寄存器 SPI_PGT_SET (偏移: 18H)	204
13.3.8	SPI 中断使能控制寄存器 SPI_INTEN (偏移: 1CH)	204
13.3.9	SPI 中断状态寄存器 SPI_INTUS (偏移: 20H)	205
13.3.10	SPI 状态寄存器 SPI_STATUS (偏移: 24H)	205
13.3.11	SPI 接收缓存寄存器 SPI_RXBUF (偏移: 28H)	205
13.4	使用流程.....	206
13.4.1	QSPI 读 FLASH.....	206
13.4.2	QSPI 写 FLASH.....	206
13.4.3	普通 SPI 读写.....	207
14	Cache.....	208
14.1	概述.....	208
14.2	主要特性.....	208
14.3	寄存器描述.....	208
14.3.1	控制寄存器 CACHE_CR (偏移: 00H).....	209
14.4	使用流程.....	209
15	CAN.....	210
15.1	概述.....	210
15.2	主要特性.....	210
15.3	寄存器描述.....	210
15.3.1	模式寄存器 CAN_MR (偏移: 00H)	211
15.3.2	指令寄存器 CAN_CMR (偏移: 04H)	212
15.3.3	状态寄存器 CAN_SR (偏移: 08H)	212
15.3.4	中断状态/应答寄存器 CAN_ISR (偏移: 0CH)	213
15.3.5	中断使能寄存器 CAN_IMR (偏移: 10H)	214
15.3.6	接收数据计数寄存器 CAN_RMC (偏移: 14H)	215
15.3.7	总线时序寄存器 CAN_BTR0 (偏移: 18H)	215
15.3.8	总线时序寄存器 CAN_BTR1 (偏移: 1CH)	216

15.3.9	发送缓存寄存器 CAN_TXBUF (偏移: 20H)	217
15.3.10	接收缓存寄存器 CAN_RXBUF (偏移: 24H)	217
15.3.11	接收过滤匹配寄存器 CAN_ACR (偏移: 28H)	218
15.3.12	接收过滤屏蔽寄存器 CAN_AMR (偏移: 2CH)	219
15.3.13	错误码捕捉寄存器 CAN_ECC (偏移: 30H)	221
15.3.14	接收错误计数寄存器 CAN_RXERR (偏移: 34H)	221
15.3.15	发送错误计数寄存器 CAN_TXERR (偏移: 38H)	221
15.3.16	仲裁丢失捕获寄存器 CAN_ALC (偏移: 3CH)	222
15.3.17	接收缓存基地址设置寄存器 CAN_RXADDR (偏移: 40H)	223
15.4	使用流程	223
15.4.1	发送 CAN 数据帧	223
15.4.2	接收 CAN 数据帧	223
15.4.3	CAN 速率计算	224
16	GTIMER	226
16.1	概述	226
16.2	主要特性	226
16.3	寄存器描述	227
16.3.1	GTIM 控制寄存器 GTIM_CR (偏移: 00H)	227
16.3.2	GTIM 中断使能寄存器 GTIM_IER (偏移: 04H)	231
16.3.3	GTIM 状态寄存器 GTIM_SR (偏移: 08H)	231
16.3.4	GTIM 事件产生寄存器 GTIM_EGR (偏移: 0CH)	232
16.3.5	GTIM 捕捉/比较模式寄存器 GTIM_CCMR (偏移: 10H)	232
16.3.6	GTIM 捕捉/比较使能寄存器 GTIM_CCER (偏移: 14H)	234
16.3.7	GTIM 计数寄存器 GTIM_CNT (偏移: 18H)	235
16.3.8	GTIM 预分频寄存器 GTIM_PSC (偏移: 1CH)	235
16.3.9	GTIM 自动重载寄存器 GTIM_ARR (偏移: 20H)	236
16.3.10	GTIM 捕捉/比较寄存器 GTIM_CCR (偏移: 24H)	236
16.3.11	GTIM 硬件触发寄存器 GTIM_CARS1 (偏移: 28H)	237
16.4	使用说明	237
16.4.1	计数器模式	237
16.4.2	输入捕获模式	238
16.4.3	PWM 模式	238
16.4.4	互补输出和死区插入	240
16.4.5	刹车功能	240
16.5	使用流程	241
16.5.1	普通定时器	241
16.5.2	PWM 输出	241
16.5.3	输入捕获	242

16.5.4	刹车功能	243
17	LPTIMER.....	244
17.1	概述.....	244
17.2	主要特性	244
17.3	寄存器描述.....	245
17.3.1	配置寄存器 LPTM_CFG (偏移: 00H)	245
17.3.2	计数值寄存器 LPTM_CNT (偏移: 04H)	247
17.3.3	比较寄存器 LPTM_COMP (偏移: 08H)	247
17.3.4	目标寄存器 LPTM_TARGET (偏移: 0CH)	247
17.3.5	中断使能寄存器 LPTM_IE (偏移: 10H)	248
17.3.6	中断标志寄存器 LPTM_IF (偏移: 14H)	248
17.3.7	控制寄存器 LPTM_CTRL (偏移: 18H)	249
17.4	使用流程	249
17.4.1	普通定时器	249
17.4.2	PWM 输出.....	249
17.4.3	Trigger 脉冲触发计数模式.....	250
17.4.4	外部异步脉冲计数模式	250
17.4.5	Timeout 模式.....	251
18	OPA252	
18.1	概述.....	252
18.2	主要特性	252
18.3	功能框图	252
18.4	寄存器描述.....	253
18.4.1	OPA 控制寄存器 OPA_CFG (偏移: 088H)	253
18.4.2	放大器状态寄存器 OPA_STATUS (偏移: 0C8H)	254
18.5	OPA 使用流程.....	254
18.6	OPA 作为 COMP 使用流程	255
19	COMP.....	256
19.1	概述.....	256
19.2	主要特性	256
19.3	功能框图	257
19.4	寄存器描述.....	258
19.4.1	COMP 控制寄存器 COMP_CFG (偏移: 08CH)	258
19.4.2	COMP2 寄存器 COMP2_STATUS (偏移: 0C4H)	259
19.5	使用流程	259
20	ADC.....	260
20.1	概述.....	260

20.2	主要特性	260
20.3	寄存器描述	261
20.3.1	ADC 通用控制寄存器 ADC_GCR (偏移: 000H)	261
20.3.2	A/D 通道 0 数据寄存器 ADC0_DR (偏移: 004H)	263
20.3.3	A/D 通道 3 数据寄存器 ADC3_DR (偏移: 010H)	263
20.3.4	A/D 通道 4 数据寄存器 ADC4_DR (偏移: 014H)	264
20.3.5	A/D 通道 5 数据寄存器 ADC5_DR (偏移: 018H)	264
20.3.6	ADC 时钟分频寄存器 ADC_CDR (偏移: 024H)	265
20.3.7	ADC 中断状态寄存器 ADC_ISR (偏移: 028H)	265
20.3.8	ADC 中断使能寄存器 ADC_IER (偏移: 02CH)	266
20.3.9	ADC 中断清除寄存器 ADC_ICR (偏移: 030H)	267
20.3.10	ADC 切换间隔计数寄存器 ADC_COUNT (偏移: 034H)	267
20.3.11	ADC 接收数据寄存器 ADC_RXREG (偏移: 038H)	268
20.3.12	ADC 状态寄存器 ADC_CSTAT (偏移: 03CH)	268
20.3.13	ADC 采样脉宽寄存器 ADC_SPW (偏移: 040H)	268
20.3.14	模拟 ADC 配置寄存器 ADC_TCRL (偏移: 044H)	269
20.3.15	ADC 硬件触发使能配置寄存器 ADC_HDT (偏移: 048H)	269
20.4	ADC 使用流程	271
20.4.1	单次扫描模式单通道 A/D 转换	271
20.4.2	单次扫描模式多通道 A/D 转换	272
20.4.3	连续扫描模式单通道 A/D 转换	273
20.4.4	连续扫描模式多通道 A/D 转换	273
20.4.5	注意事项	274
20.5	ADC 经 OPA 缓冲采样使用流程	275
20.5.1	ADC 经 OPA 缓冲采样图	275
20.5.2	ADC 经 OPA 缓冲采样流程图	275
20.5.3	ADC 经 OPA 缓冲后采样使用流程	276
21	GPIO	277
21.1	概述	277
21.2	主要特性	277
21.3	寄存器描述	277
21.3.1	数据方向寄存器 GPIO_DIR (偏移: 00H)	278
21.3.2	输出置位寄存器 GPIO_SET (偏移: 08H)	278
21.3.3	输出清零寄存器 GPIO_CLR (偏移: 0CH)	278
21.3.4	GPIO 输出引脚映射寄存器 GPIO_ODATA (偏移: 10H)	279
21.3.5	GPIO 输入引脚映射寄存器 GPIO_IDATA (偏移: 14H)	279
21.3.6	GPIO 中断使能寄存器 GPIO_IEN (偏移: 18H)	279
21.3.7	GPIO 中断触发模式寄存器 GPIO_IS (偏移: 1CH)	280

21.3.8	GPIO 中断边沿触发设置寄存器 GPIO_IBE (偏移: 20H)	280
21.3.9	GPIO 中断高低电平触发设置寄存器 GPIO_IHV (偏移: 24H)	280
21.3.10	GPIO 中断状态清除寄存器 GPIO_IC (偏移: 28H)	280
21.3.11	GPIO 原始中断状态寄存器 GPIO_RIS (偏移: 2CH)	281
21.3.12	GPIO 屏蔽后中断状态寄存器 GPIO_MIS (偏移: 30H)	281
21.4	使用流程	281
21.4.1	输入输出 IO	281
21.4.2	中断触发模式	282
21.4.3	清除中断	282
22	CRC16	283
22.1	概述	283
22.2	寄存器描述	283
22.2.1	数据寄存器 CRC16_DATA (偏移: 00H)	283
22.2.2	初始值寄存器 CRC16_INIT (偏移: 04H)	284
22.2.3	控制寄存器 CRC16_CTRL (偏移: 08H)	284
22.3	使用流程	284
23	WDT	285
23.1	概述	285
23.2	主要特性	285
23.3	寄存器描述	285
23.3.1	装载寄存器 WDT_LOAD (偏移: 00H)	286
23.3.2	计数寄存器 WDT_CNT (偏移: 04H)	286
23.3.3	控制寄存器 WDT_CTRL (偏移: 08H)	286
23.3.4	清除寄存器 WDT_CLR (偏移: 0CH)	286
23.3.5	RAW 中断状态寄存器 WDT_INTRAW (偏移: 10H)	287
23.3.6	MASK 中断状态寄存器 WDT_MINTS (偏移: 14H)	287
23.3.7	STALL 控制寄存器 WDT_STALL (偏移: 18H)	287
23.3.8	LOCK 寄存器 WDT_LOCK (偏移: 1CH)	288
23.4	使用流程	288
24	WWDT	289
24.1	概述	289
24.2	主要特性	289
24.3	寄存器描述	289
24.3.1	控制寄存器 WWDT_CON (偏移: 00H)	290
24.3.2	配置寄存器 WWDT_CFG (偏移: 04H)	290
24.3.3	计数寄存器 WWDT_CNT (偏移: 08H)	290
24.3.4	中断使能寄存器 WWDT_IE (偏移: 0CH)	290

24.3.5	中断标志寄存器 WWDT_IF (偏移: 10H).....	291
24.4	使用流程	291
25	RTC 292	
25.1	概述	292
25.2	主要特性	292
25.3	低功耗时基分频器 (LTBC)	292
25.3.1	LTBC 功能	292
25.3.2	LTBC 数字调校	293
25.4	时间戳功能	293
25.5	寄存器描述	294
25.5.1	写使能寄存器 (RTCWE) (偏移: 00H)	295
25.5.2	中断使能寄存器 (RTCIE) (偏移: 04H)	295
25.5.3	中断标志寄存器 (RTCIF) (偏移: 08H)	296
25.5.4	BCD 时间秒寄存器 (RTC_BCDSEC) (偏移: 0CH)	298
25.5.5	BCD 时间分钟寄存器 (RTC_BCDMIN) (偏移: 10H)	298
25.5.6	BCD 时间小时寄存器 (RTC_BCDHOUR) (偏移: 14H)	299
25.5.7	BCD 时间天寄存器 (RTC_BCDDATE) (偏移: 18H)	299
25.5.8	BCD 时间星期寄存器 (RTC_BCDWEEK) (偏移: 1CH)	299
25.5.9	BCD 时间月寄存器 (RTC_BCDMONTH) (偏移: 20H)	299
25.5.10	BCD 时间年寄存器 (RTC_BCDYEAR) (偏移: 24H)	299
25.5.11	闹钟寄存器 (RTC_ALARM) (偏移: 28H)	299
25.5.12	时钟信号输出控制寄存器 (RTC_FSEL) (偏移: 2CH)	300
25.5.13	LTBC 数值调整寄存器 (RTC_ADJUST) (偏移: 30H)	301
25.5.14	LTBC 数值调整方向寄存器 (RTC_ADSIGN) (偏移: 34H)	301
25.5.15	LTBC 虚拟调校使能寄存器 (RTC_PR1SEN) (偏移: 38H)	301
25.5.16	毫秒计数寄存器 (RTC_MSCNT) (偏移: 3CH)	302
25.5.17	RTC 时间戳使能寄存器 (RTC_STORREN) (偏移: 40H)	302
25.5.18	CLKSTAMP0R 寄存器 (RTC_CLKS0R) (偏移: 44H)	302
25.5.19	CALSTAMP0R 寄存器 (RTC_CALS0R) (偏移: 48H)	303
25.5.20	CLKSTAMP0F 寄存器 (RTC_CLKS0F) (偏移: 4CH)	303
25.5.21	CALSTAMP0F 寄存器 (RTC_CALS0F) (偏移: 50H)	303
25.5.22	CLKSTAMP1R 寄存器 (RTC_CLKS1R) (偏移: 54H)	304
25.5.23	CALSTAMP1R 寄存器 (RTC_CALS1R) (偏移: 58H)	304
25.5.24	CLKSTAMP1F 寄存器 (RTC_CLKS1F) (偏移: 5CH)	305
25.5.25	CALSTAMP1F 寄存器 (RTC_CALS1F) (偏移: 60H)	305
25.6	使用流程	305
25.6.1	RTC 时间设置	305
25.6.2	RTC 时间读取	306

25.6.3	时间戳使用	306
25.6.4	RTC 设置闹钟	307
26	DMA.....	308
26.1	概述	308
26.2	主要特性	308
26.3	寄存器描述.....	308
26.3.1	通道源传送地址寄存器 DMA_SRC_ADDR_Cx (偏移: 10xH) (x=0,1,2,3) ...	309
26.3.2	通道目的传送地址寄存器 DMA_DST_ADDR_Cx (偏移: 10x+04H) (x=0,1,2,3)	309
26.3.3	通道控制信息寄存器 DMA_CH_CTRL_Cx (偏移: 10x+08H) (x=0,1,2,3)	310
26.3.4	通道传送状态寄存器 DMA_CH_STS_Cx (偏移: 10x+0CH) (x=0,1,2,3)	311
26.3.5	DMA 控制器使能寄存器 DMAC_EN (偏移: 40H)	311
26.3.6	DMA 软复位寄存器 DMA_SOFT_RESET (偏移: 44H)	311
26.3.7	DMA 中断指示寄存器 DMA_INT_STATUS (偏移: 48H)	312
26.3.8	DMA 中断屏蔽寄存器 DMA_INT_MASK (偏移: 4CH)	312
26.3.9	DMA 外设请求寄存器 DMA_PER_REQ DMA (偏移: 54H)	313
26.4	使用流程	313
27	SysTick	314
27.1	概述	314
27.2	寄存器描述.....	314
27.2.1	控制和状态寄存器 SYS_CSR (偏移: 00H).....	314
27.2.2	重载值寄存器 SYS_RVR (偏移: 04H).....	315
27.2.3	当前值寄存器 SYS_CVR (SysTick->VAL) (偏移: 08H)	315
27.3	使用流程	315
28	典型应用	317
28.1	参考原理图.....	317
28.2	+13dBm 匹配参数	318
28.3	+18dBm 匹配参数	319
29	封装尺寸	321
29.1	QFN40 (5*5 mm)	321
30	版本维护	322

图目录

图 1-1: 功能框图	2
图 1-2: 电源框图	3
图 2-1: QFN40 引脚分布图	4
图 3-1: Cortex-M0+处理器功能框图	14
图 3-2: Cortex-M0+的寄存器组	14
图 4-1: 时钟模块框图	16
图 5-1: POR 复位时序图	91
图 5-2: RSSI 线性图	97
图 5-3: SPI 读写寄存器时序图	99
图 5-4: SPI 读写 FIFO 时序图	99
图 5-5: 状态机控制图	105
图 5-6: Mode 0 帧格式	110
图 5-7: Mode 1 帧格式	110
图 5-8: Mode 2 帧格式	110
图 5-9: Mode 3 帧格式	111
图 5-10: PAYLOAD_BIT_ORDER 操作	116
图 5-11: PN7 抽出模式移位顺序	117
图 5-12: PN9 插入模式移位顺序	117
图 5-13: m3_length_en=0 时 MODE 3 帧格式	119
图 5-14: 可变长度包格式, Address 不存在	120
图 5-15: Address 存在, Address 在 Length Byte 之前	121
图 5-16: Address 存在, Address 在 Length Byte 之后	122
图 5-17: ACK 的包格式	126
图 5-18: CRC 编码范围	127
图 5-19: CRC_INV	128
图 5-20: CRC_BIT_ORDER	128
图 5-21: CRC8 的配置示例	129
图 5-22: CRC16 的配置示例	129
图 6-1: eFlash Main 区启动地址映射	135
图 6-2: 写操作流程	136
图 6-3: 擦除操作流程	137
图 10-1: I2C 通信协议 (7 位寻址) 框图	167
图 10-2: I2C 通信协议 (10 位寻址) 框图	169
图 11-1: SPI0 数据/时钟时序图 (CPHA=0)	184
图 11-2: SPI0 数据/时钟时序图 (CPHA=1)	185
图 11-3: SPI0 SSN 时序图 (CPHA=0)	185

图 11-4: SPI0 SSN 时序图 (CPHA=1)	186
图 12-1: SPI1 数据/时钟时序图 (CPHA=0)	195
图 12-2: SPI1 数据/时钟时序图 (CPHA=1)	196
图 12-3: SPI1 SSN 时序图 (CPHA=0)	196
图 12-4: SPI1 SSN 时序图 (CPHA=1)	197
图 14-1: Cache 位置示意图	208
图 15-1: CAN 位周期结构	217
图 15-2: CAN RX FIFO 29 位 ID 数据格式	218
图 15-3: 单过滤器	219
图 16-1: PWM 中央对齐时序图	240
图 18-1: OPA 功能框图	252
图 19-1: COMP 功能框图	257
图 20-1: ADC 经 OPA 缓冲采样图	275
图 20-2: ADC 经 OPA 缓冲采样流程图	275
图 28-1: 参考原理图	317
图 29-1: QFN40 封装图	321

表目录

表 2-1: 引脚功能复用..... 6

表 2-2: 引脚功能描述..... 8

表 4-1: 模块地址划分..... 15

表 4-2: 系统时钟选择..... 17

表 4-3: 系统复位源..... 17

表 4-4: 低功耗模式..... 20

表 4-5: 系统寄存器列表..... 22

表 5-1: 外部复位相关寄存器..... 92

表 5-2: 软复位相关寄存器..... 92

表 5-3: 频率综合器相关寄存器..... 93

表 5-4: 晶振相关寄存器..... 94

表 5-5: AGC 相关寄存器..... 95

表 5-6: RSSI 相关寄存器列表..... 95

表 5-7: Reg2B 增益控制表..... 97

表 5-8: AFC 相关寄存器..... 98

表 5-9: SPI 时序参数..... 100

表 5-10: FIFO 相关寄存器列表..... 100

表 5-11: GPIO 配置相关寄存器列表..... 103

表 5-12: GPIO 输出功能描述..... 104

表 5-13: 工作模式表..... 106

表 5-14: Direct 相关寄存器列表..... 107

表 5-15: Mode 相关寄存器列表..... 109

表 5-16: Preamble 配置相关寄存器列表..... 112

表 5-17: Sync 配置相关寄存器列表..... 113

表 5-18: Payload 相关寄存器..... 114

表 5-19: Length 相关的寄存器列表..... 118

表 5-20: Address 配置相关寄存器列表..... 123

表 5-21: seqnum 相关的寄存器列表..... 124

表 5-22: FCS2 相关的寄存器列表..... 125

表 5-23: CRC 相关的寄存器列表..... 126

表 5-24: 常用 CRC 标准和多项式列表..... 128

表 6-1: EFC 寄存器列表..... 130

表 7-1: 中断源..... 138

表 8-1: UART0 寄存器列表..... 141

表 9-1: UART1 寄存器列表..... 147

表 10-1: I2C 寄存器列表..... 159

表 11-1: SPI0 寄存器列表..... 176

表 11-2: SPI0 两种引脚搭配方式..... 186

表 12-1: SPI1 寄存器列表..... 189

表 13-1: QSPI 寄存器列表..... 200

表 14-1: Cache 寄存器列表..... 208

表 15-1: CAN 寄存器列表..... 210

表 16-1: GTIMER 寄存器列表..... 227

表 17-1: LPTIMER 寄存器列表..... 245

表 18-1: OPA 寄存器列表..... 253

表 19-1: COMP 寄存器列表..... 258

表 20-1: ADC 寄存器列表..... 261

表 21-1: GPIO 寄存器列表..... 277

表 22-1: CRC 寄存器列表..... 283

表 23-1: WDT 寄存器列表..... 285

表 24-1: WWDT 寄存器列表..... 289

表 25-1: RTC 寄存器列表..... 294

表 26-1: DMA 寄存器列表..... 308

表 27-1: SysTick 寄存器列表..... 314

表 28-1: +13dBm 参数表..... 318

表 28-2: +18dBm 参数表..... 319

1 系统概述

UM2080F32 是广芯微电子（广州）股份有限公司研制的基于 ARM Cortex-M0+内核的高性能的、单片集成（G）FSK/OOK 无线收发器的 32 位 SoC 芯片。

UM2080F32 内部集成了完整的射频接收机、射频发射机、频率综合器及调制解调器，用户只需配备简单、低成本的外围器件即可获得良好的收发性能。芯片工作于 200 ~ 960MHz 范围内，支持灵活可设的数据包格式，支持自动应答与自动重发功能，支持跳频操作与 FEC 功能。

此外，UM2080F32 内部还集成了 CAN、12 位 SAR ADC、UART、SPI、QSPI、I2C 等通用外围通讯接口，ADC、OPA、COMP 等传感获取接口，以及 LPTIMER、WDT 等超低功耗模块接口，并内置 RC 高频和低频振荡器。

芯片系统采用了独特的低功耗设计技术，具有高集成度、高抗干扰性、高可靠性和超低功耗等技术特点。支持 Keil MDK 集成开发环境，支持 C 语言和汇编语言进行软件开发。

典型应用场景：

- 工业传感及工业控制
- 安防系统
- 自动抄表
- 无线标签、无线门禁
- 遥控装置、无线玩具
- 智能交通、智慧城市、智能家居
- 智能门锁、资产追踪、无线监控等智能传感器终端应用

1.1 功能框图

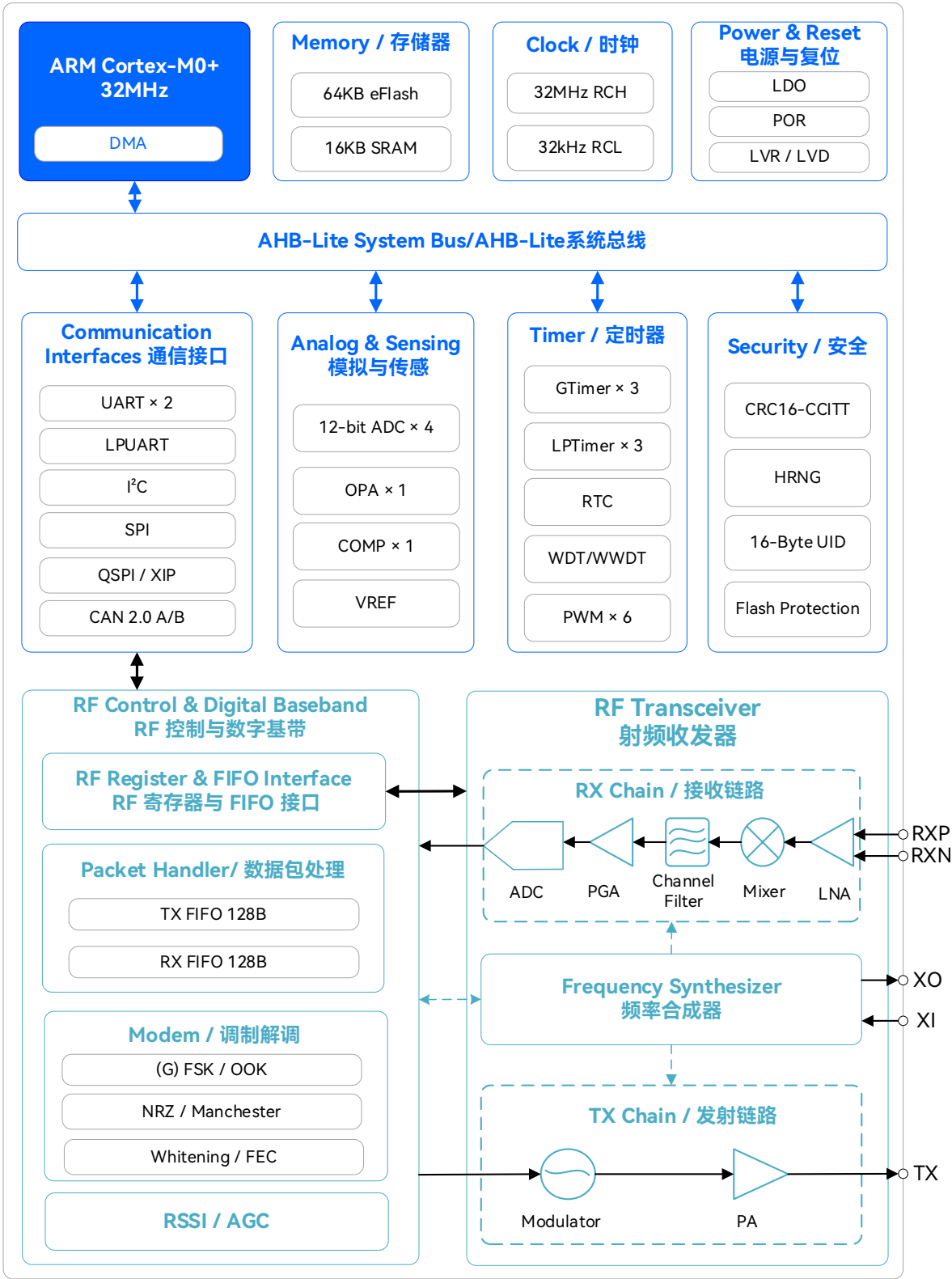


图 1-1: 功能框图

1.2 电源框图

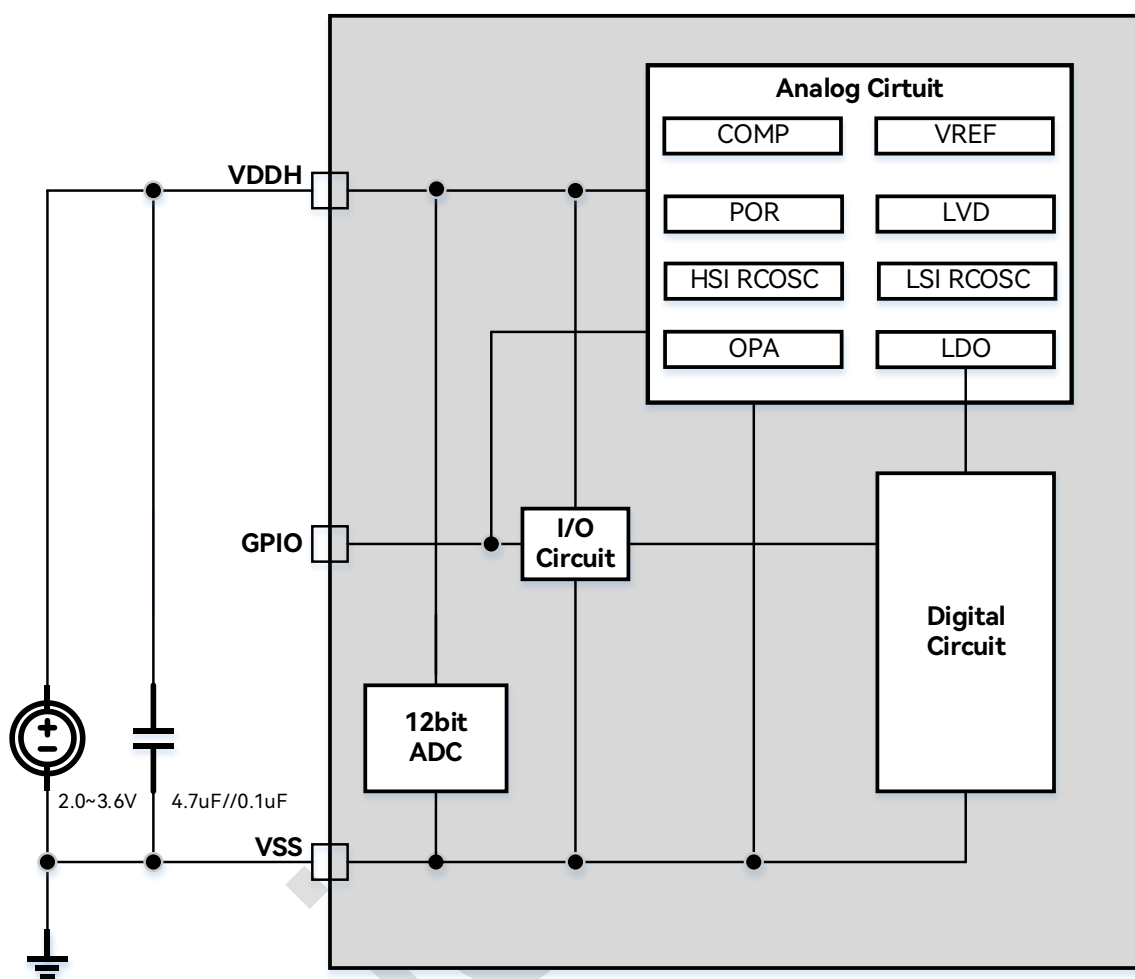


图 1-2: 电源框图

注：每组电源都需要一个去耦电容，去耦电容尽量靠近相应电源引脚。

2 引脚定义和描述

2.1 引脚分布

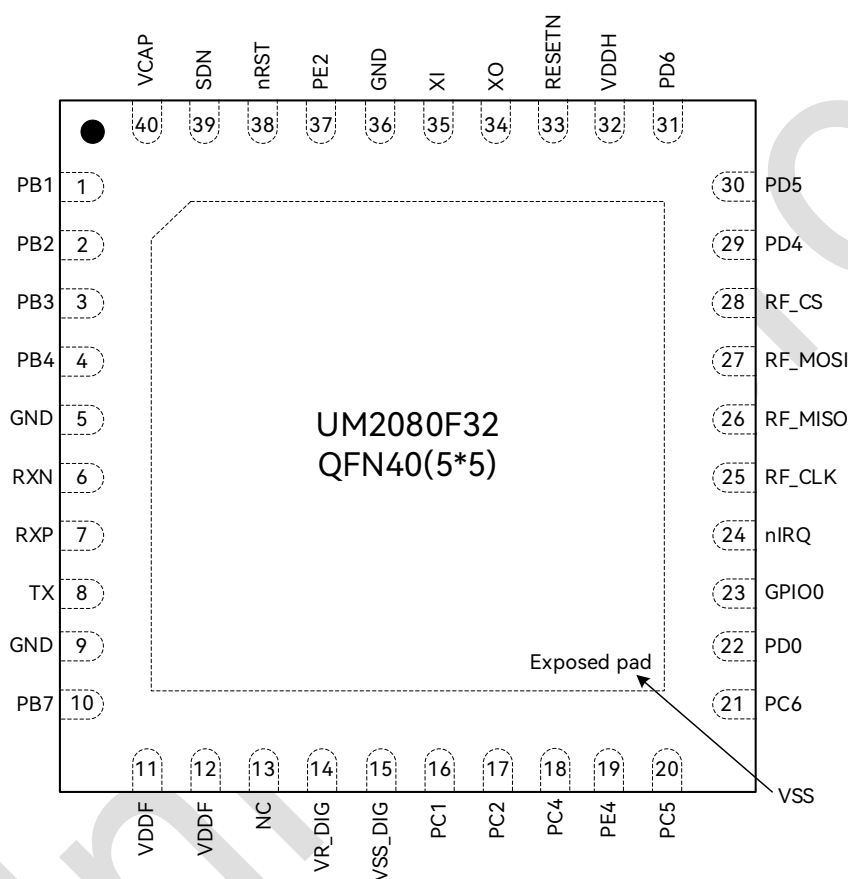


图 2-1: QFN40 引脚分布图

注：RF_CS、RF_CLK、RF_MOSI、RF_MISO、nRST、SDN、GPIO0 为 RF 与 MCU 协同控制相关引脚，其中：

- RF_CLK、RF_MOSI、RF_MISO 是 RF 与 MCU 通信互连保留引脚，外部不建议用作其他功能，应用中应保持 NC
- RF_CS:RF 的 SPI 片选，需板级连接至 MCU GPIO 引脚
- nIRQ: RF 多功能 IO，默认为中断输出；如应用需检测 RF 中断，需板级连接至 MCU GPIO
- SDN: RF 的关断引脚，如应用需控制 RF 关断/唤醒，需板级连接至 MCU GPIO；如应用不需要独立关断控制，可接固定低电平

- nRST: RF 多功能 IO, 默认为复位输入引脚, 如应用需要控制 RF 复位, 需板级连接 MCU GPIO;
如应用不需要独立控制复位, 可接固定高电平
- GPIO0: RF 多功能 IO, 可输出 RF 信号或直通输入数据; 如应用需要直通输入数据, 需连接至 MCU GPIO。如需要测试 RF 信号, 预留测试点

2.2 引脚复用

表 2-1: 引脚功能复用

引脚 编号	Config	Px_SEL[i+3;i]							
		0	1	2	3	4	5	6	7
1	AIN3/ OPA_N2	PB1	SPI1_CSN0	GTIM1_CHN	LPTIM0_EXT	LPTIM0_IN	-	I2C_SCL	-
2	AIN4/ OPA_O2P	PB2	-	SPI0_CSN0	GTIM0_CH	SPI0_MOSI	LPTIM1_IN	GTIM2_CHN	QSPI_HOLD
3	AIN5	PB3	-	-	LPTIM0_EXT	CAN_RX	RTC_TAMP1	LPTIM2_IN	GTIM0_BK
4	-	PB4	SPI0_MOSI	-	UART1_CTS	-	LPTIM0_OUT	CAN_TX	QSPI_MOSI
10	-	PB7	SPI0_SCK	LPTIM0_OUT	LPTIM2_EXT	RTC_TAMP0	GTIM2_CHN	QSPI_HOLD	GTIM2_BK
16	-	PC1	I2C_SCL	UART1_TX	-	SPI0_MISO	GTIM1_CH	LPTIM0_OUT	CAN_RX
17	-	PC2	I2C_SDA	UART1_RX	-	SPI0_CSN1	GTIM2_CH	LPTIM1_IN	CLKOUT
18	-	PC4	UART1_RTS	-	UART0_RX	SPI0_MI1	-	LPTIM2_EXT	QSPI_WP
19	-	PE4	QSPI_SCK	SPI0_MISO	I2C_SCL	-	GTIM2_BK	-	-
20	-	PC5	SWIO	-	LPTIM0_EXT	I2C_SDA	-	-	-
21	-	PC6	SWCLK	UART1_TX	-	-	-	LPTIM0_OUT	-
22	-	PD0	SPI1_CSN0	GTIM0_CH	UART1_RX	LPTIM1_IN	RTC_TAMP0	GTIM2_CHN	QSPI_WP
25	-	RF_CLK	SPI1_CLK	-	-	-	-	-	-
26	-	RF_MISO	SPI1_MISO	-	-	-	-	-	-
27	-	RF_MOSI	SPI1_MOSI	-	-	-	-	-	-

引脚 编号	Config	Px_SEL[i+3;i]							
		0	1	2	3	4	5	6	7
29	COMP2_INP	PD4	UART1_TX	I2C_SCL	-	-	SPI0_SCK	GTIM2_CH	LPTIM0_EXT
30	COMP2_INN	PD5	I2C_SDA	LPTIM1_IN	UART1_RX	-	GTIM0_CHN	CAN_RX	-
31	OPA_P2	PD6	UART0_TX	SPI0_MISO	LPTIM1_EXT	CAN_TX	QSPI_MISO	SPI0_CSN0	LPTIM2_OUT
33	RESETN	PA2	-	UART1_RX	UART0_RX	-	I2C_SCL	I2C_SDA	-
37	-	PE2	QSPI_CSN	SPI0_MI1	-	UART1_CTS	GTIM0_BK	-	-

2.3 引脚描述

表 2-2: 引脚功能描述

引脚 编号	引脚 名称	IO 类型	复位状态		引脚类型	功能描述
			DIR	PU PD		
0	VSS	G	-	-	GND	芯片地(LF PAD)公共地
1	PB1	I/O	DI	HZ	PB1（默认）	通用数字输入/输出引脚
					SPI1_CSN0	SPI1的CS0信号
					GTIM1_CHN	GTIMER1 的反向互补信号
					LPTIM0_EXT	LPTIMER0 的外部触发信号
					LPTIM0_IN	LPTIMER0 的输入信号
					I2C_SCL	I2C 的 SCL 信号
					AIN3	ADC 转换通道 3
					OPA_N2	OPA 的反向端输入信号 2
2	PB2	I/O	DI	HZ	PB2（默认）	通用数字输入/输出引脚
					SPI0_CSN0	SPI0 的 CS0 信号
					GTIM0_CH	GTIMER0 的 PWM 输出/输入捕获信号
					SPI0_MOSI	SPI0 的 MOSI 信号
					LPTIM1_IN	LPTIMER1 的输入信号
					GTIM2_CHN	GTIMER2 的反向互补信号
					QSPI_HOLD	QSPI 的 HOLD 信号
					AIN4	ADC 转换通道 4
					OPA_O2P	OPA 的输出信号
3	PB3	I/O	DI	HZ	PB3（默认）	通用数字输入/输出引脚
					LPTIM0_EXT	LPTIMER0 的外部触发信号
					CAN_RX	CAN 的 RX 信号
					RTC_TAMP1	RTC 的 TAMP1 输入信号
					LPTIM2_IN	LPTIMER2 的输入信号
					GTIM0_BK	GTIMER0 的刹车信号
					AIN5	ADC 转换通道 5
4	PB4	I/O	DI	HZ	PB4（默认）	通用数字输入/输出引脚
					SPI0_MOSI	SPI0 的 MOSI 信号
					UART1_CTS	UART1 的 CTS 信号

引脚 编号	引脚 名称	IO 类型	复位状态		引脚类型	功能描述
			DIR	PU PD		
					LPTIM0_OUT	LPTIMER0 的 PWM 输出信号
					CAN_TX	CAN 的 TX 信号
					QSPI_MOSI	QSPI 的 MOSI 信号
5	GND	G	-	-	GND	芯片地
6	RXN	RFI	-	-	RXN	射频负端输入
7	RXP	RFI	-	-	RXP	射频正端输入
8	TX	RFO	-	-	TX	射频 PA 输出
9	GND	G	-	-	GND	芯片地
10	PB7	I/O	DI	HZ	PB7 (默认)	通用数字输入/输出引脚
					SPI0_SCK	SPI0 的 SCK 信号
					LPTIM0_OUT	LPTIMER0 的 PWM 输出信号
					LPTIM2_EXT	LPTIMER2 的外部触发信号
					RTC_TAMP0	RTC 的 TAMP0 输入信号
					GTIM2_CHN	GTIMER2 的反向互补信号
					QSPI_HOLD	QSPI 的 HOLD 信号
					GTIM2_BK	GTIMER2 的刹车信号
11	VDDF	P	-	-	VDDF	2.0 ~ 3.6V 电源输入
12	VDDF	P	-	-	VDDF	2.0 ~ 3.6V 电源输入
13	NC	-	-	-	NC	空脚, 保持悬空
14	VR_DIG	DO	-	-	VR_DIG	数字 LDO 输出, 外接电容
15	VSS_DIG	G	-	-	VSS_DIG	芯片地
16	PC1	I/O	DO	-	PC1	通用数字输入/输出引脚
					I2C_SCL	I2C 的 SCL 信号
					UART1_TX (默认)	UART1 的 TX 信号 (UART BOOT 下载需用此口)
					SPI0_MISO	SPI0 的 MISO 信号
					GTIM1_CH	GTIMER1 的 PWM 输出/输入捕获信号
					LPTIM0_OUT	LPTIMER0 的 PWM 输出信号
					CAN_RX	CAN 的 RX 信号
17	PC2	I/O	DI	-	PC2	通用数字输入/输出引脚
					I2C_SDA	I2C 的 SDA 信号
					UART1_RX (默认)	UART1 的 RX 信号 (UART BOOT 下载需用此口)

引脚 编号	引脚 名称	IO 类型	复位状态		引脚类型	功能描述
			DIR	PU PD		
					SPI0_CSN1	SPI0 的 CS1 信号, 仅 master 模式 (只能与 SPI0_MI1 搭配使用)
					GTIM2_CH	GTIMER2 的 PWM 输出/输入捕获信号
					LPTIM1_IN	LPTIMER1 的输入信号
					CLKOUT	时钟信号输出
18	PC4	I/O	DI	HZ	PC4 (默认)	通用数字输入/输出引脚
					UART1_RTS	UART1 的 RTS 信号
					UART0_RX	UART0 的 RX 信号
					SPI0_MI1	SPI0 的 MISO 信号 1 (只能与 SPI0_CSN1 搭配使用)
					LPTIM2_EXT	LPTIMER2 的外部触发信号
					QSPI_WP	QSPI 的 WP 信号
19	PE4	I/O	DI	HZ	PE4 (默认)	通用数字输入/输出引脚
					QSPI_SCK	QSPI 的 SCK 信号
					SPI0_MISO	SPI0 的 MISO 信号
					I2C_SCL	I2C 的 SCL 信号
					GTIM2_BK	GTIMER2 的刹车功能
20	PC5	I/O	DI	PU	PC5	通用数字输入/输出引脚
					SWIO (默认)	JTAG SWD 的数据信号
					LPTIM0_EXT	LPTIMER0 的外部触发信号
					I2C_SDA	I2C 的 SDA 信号
21	PC6	I/O	DI	PU	PC6	通用数字输入/输出引脚
					SWCLK (默认)	JTAG SWD 的时钟信号
					UART1_TX	UART1 的 TX 信号
					LPTIM0_OUT	LPTIMER0 的 PWM 输出信号
22	PD0	I/O	DI	HZ	PD0 (默认)	通用数字输入/输出引脚
					SPI1_CSN0	SPI1 的 CS0 信号
					GTIM0_CH	GTIMER0 的 PWM 输出/输入捕获信号
					UART1_RX	UART1 的 RX 信号
					LPTIM1_IN	LPTIMER1 的输入信号
					RTC_TAMP0	RTC 的 TAMP0 输入信号
					GTIM2_CHN	GTIMER2 的反向互补信号

引脚 编号	引脚 名称	IO 类型	复位状态		引脚类型	功能描述
			DIR	PU PD		
					QSPI_WP	QSPI 的 WP 信号
23	GPIO0	I/O	DO	-	GPIO0	GPIO0 可配置功能引脚
24	nIRQ	I/O	DO	-	nIRQ	RF 中断输出信号
25	RF_CLK	I/O	DI	-	RF_CLK	SPI1 的 SCLK 信号 (RF 和 MCU 内部互联)
26	RF_MISO	I/O	DI	-	RF_MISO	SPI1 的 MISO 信号 (RF 和 MCU 内部互联)
27	RF_MOSI	I/O	DI	-	RF_MOSI	SPI1 的 MOSI 信号 (RF 和 MCU 内部互联)
28	RF_CS	I/O	DI	-	RF_CS	RF 片选信号
29	PD4	I/O	DI	HZ	PD4 (默认)	通用数字输入/输出引脚
					UART1_TX	UART1 的 TX 信号
					I2C_SCL	I2C 的 SCL 信号
					SPI0_SCK	SPI0 的 SCK 信号
					GTIM2_CH	GTIMER2 的 PWM 输出/输入捕获信号
					LPTIM0_EXT	LPTIMER0 的外部触发信号
					COMP2_INP	比较器 2 的 P 输入信号
30	PD5	I/O	DI	HZ	PD5 (默认)	通用数字输入/输出引脚
					I2C_SDA	I2C 的 SDA 信号
					LPTIM1_IN	LPTIMER1 的输入信号
					UART1_RX	UART1 的 RX 信号
					GTIM0_CHN	GTIMER0 的反向互补信号
					CAN_RX	CAN 的 RX 信号
					COMP2_INN	比较器 2 的 N 输入信号
31	PD6	I/O	DI	HZ	PD6 (默认)	通用数字输入/输出引脚
					UART0_TX	UART0 的 TX 信号
					SPI0_MISO	SPI0 的 MISO 信号
					LPTIM1_EXT	LPTIMER1 的外部触发信号
					CAN_TX	CAN 的 TX 信号
					QSPI_MISO	QSPI 的 MISO 信号
					SPI0_CSN0	SPI0 的 CS0 信号
					LPTIM2_OUT	LPTIMER2 的 PWM 输出信号
					OPA_P2	OPA 的正向端输入信号 2

引脚 编号	引脚 名称	IO 类型	复位状态		引脚类型	功能描述
			DIR	PU PD		
32	VDDH	P	-	-	VDDH	2.0V~3.6V 电源输入
33	RESETN (PA2)	I/O	DI	PU	PA2	通用数字输入/输出引脚
					RESETN (默认)	外部复位输入
					UART1_RX	UART1 的 RX 信号
					UART0_RX	UART0 的 RX 信号
					I2C_SCL	I2C 的 SCL 信号
					I2C_SDA	I2C 的 SDA 信号
34	XO	AO	-	-	XO	晶振输出
35	XI	AI	-	-	XI	晶振输入
36	GND	G	-	-	GND	芯片地
37	PE2	I/O	DI	HZ	PE2 (默认)	通用数字输入/输出引脚
					QSPI_CSN	QSPI 的 CS 信号
					SPI0_MI1	SPI0 的 MISO 信号 1 (只能与 SPI0_CS1 搭配使用)
					UART1_CTS	UART1 的 CTS 信号
					GTIMER0_BK	GTIMER0 的刹车信号
38	nRST	DIO	-	-	nRST	GPIO2 可配置功能引脚, 默认为 RF 外部复位引脚
39	SDN	DI	-	-	SDN	RF 关断使能, SDN 高电平时 RF 处于关断模式
40	VCAP	-	-	-	VCAP	外接电容 (4.7μF)

注:

RF-射频信号; A- 模拟信号; D- 数字信号; I- Input (输入); O- Output (输出); G- Ground (接地); P- Power (电源); PU- pull up (上拉); PD- pull down (下拉); HZ- 高阻状态。

3 处理器

3.1 概述

Cortex® M0+处理器是 32 位的两级流水线 RISC 处理器，内嵌 AMBA-Lite 接口和嵌套向量中断控制器 (NVIC)。具有硬件调试功能，可以执行 Thumb 指令，并与其它 Cortex-M 系列兼容。处理器运算能力达到 1.11Dhrystone MIPS/MHz。同时加入多项全新设计，改进调试和追踪能力、减少每条指令循环 (IPC) 数量和改进 Flash 访问的两级流水线等，更纳入了节能降耗技术。Cortex® M0+ 处理器全面支持已整合 Keil & IAR 调试器。

3.2 主要特性

- ARMv6 M Thumb
- Thumb/Thumb 2 技术
- ARMv6 M 兼容 24 位 SysTick 定时器
- 32 位硬件乘法器
- 系统接口支持小端 (little-endian) 数据访问
- 准确而及时的中断处理能力
- 加载、存储多个数据和多周期乘法指令可被终止后重新开始从而实现快速中断处理
- C 应用程序二进制接口的异常兼容模式 (C-ABI)。ARMv6 M 的模式允许用户使用纯 C 函数实现中断处理
- 使用中断唤醒 (WFI) 与事件唤醒 (WFE) 指令进入低功耗的休眠模式，或者从中断退出休眠模式

3.3 功能框图

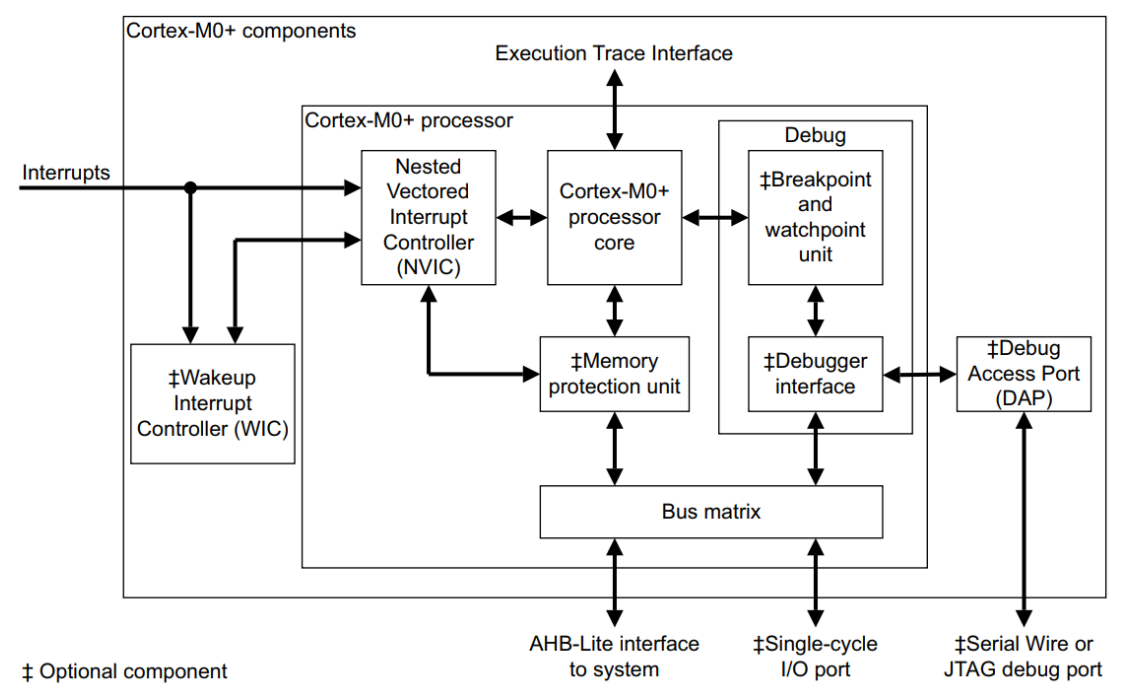


图 3-1: Cortex-M0+处理器功能框图

3.4 内核寄存器组

Cortex-M0+处理器寄存器组如下图:

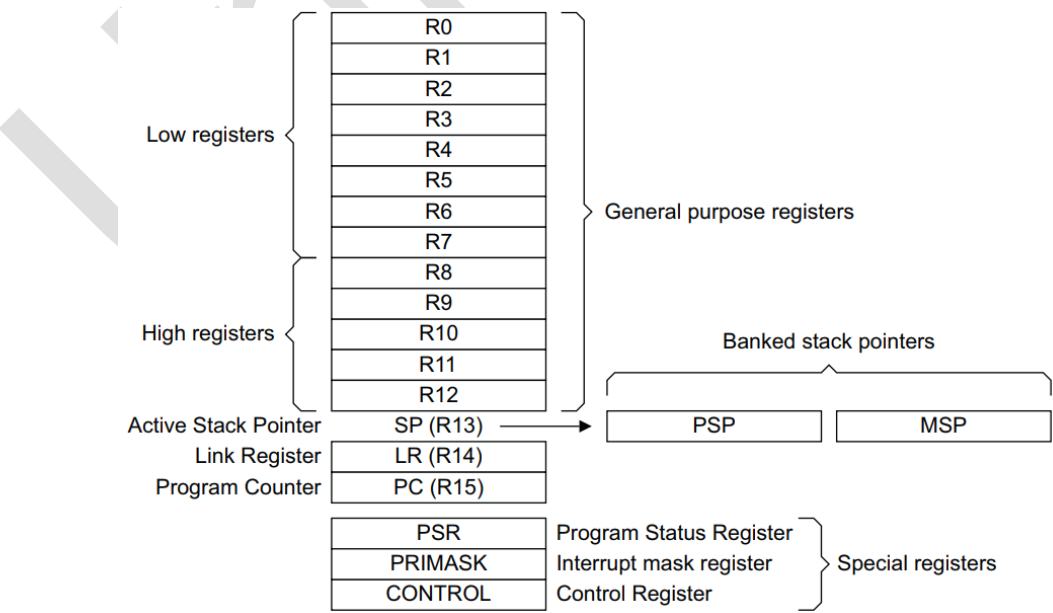


图 3-2: Cortex-M0+的寄存器组

4 系统配置 (SCU)

4.1 地址映射

表 4-1: 模块地址划分

模块名	模块地址	大小
FLASH	0x0000_0000——0x0001_0000	64kBytes
EFC	0x0110_0000——0x0110_0400	1kBytes
QSPI	0x0110_0400——0x0110_0800	1kBytes
SRAM1	0x2000_0000——0x2000_3000	12kBytes
SRAM2	0x2000_3000——0x2000_3800	2kBytes
ICACHE(WAP)	0x2000_3800——0x2000_4000	2kBytes
ICACHE(TAG)	0x2000_4000——0x2000_4400	1kBytes
ICACHE(REG)	0x2000_4400——0x2000_4800	1kBytes
UART0	0x4000_0000——0x4000_0400	1kBytes
SPI0	0x4000_0800——0x4000_0C00	1kBytes
GTIMER0	0x4000_0C00——0x4000_1000	1kBytes
LPTIMER0	0x4000_1000——0x4000_1400	1kBytes
RTC	0x4000_1400——0x4000_1800	1kBytes
CRC	0x4000_1800——0x4000_1C00	1kBytes
ADC	0x4000_1C00——0x4000_2000	1kBytes
SYSREG(SCU)	0x4000_2000——0x4000_2400	1kBytes
WDT	0x4000_2400——0x4000_2800	1kBytes
LPTIM1	0x4000_2800——0x4000_2C00	1kBytes
LPTIM2	0x4000_2C00——0x4000_3000	1kBytes
UART1	0x4000_3000——0x4000_3400	1kBytes
GTIMER1	0x4000_3400——0x4000_3800	1kBytes
GTIMER2	0x4000_3800——0x4000_3C00	1kBytes
WWDT	0x4000_3C00——0x4000_4000	1kBytes
GPIOA	0x4000_4000——0x4000_4400	1kBytes
GPIOB	0x4000_4400——0x4000_4800	1kBytes
GPIOC	0x4000_4800——0x4000_4C00	1kBytes
GPIOD	0x4000_4C00——0x4000_5000	1kBytes
GPIOE	0x4000_5000——0x4000_5400	1kBytes
I2C	0x4000_5400——0x4000_5800	1kBytes

模块名	模块地址	大小
CAN	0x4000_5C00——0x4000_6000	1Kbytes
DMA	0x4002_0000——0x4002_0400	1kBytes

4.2 时钟框图

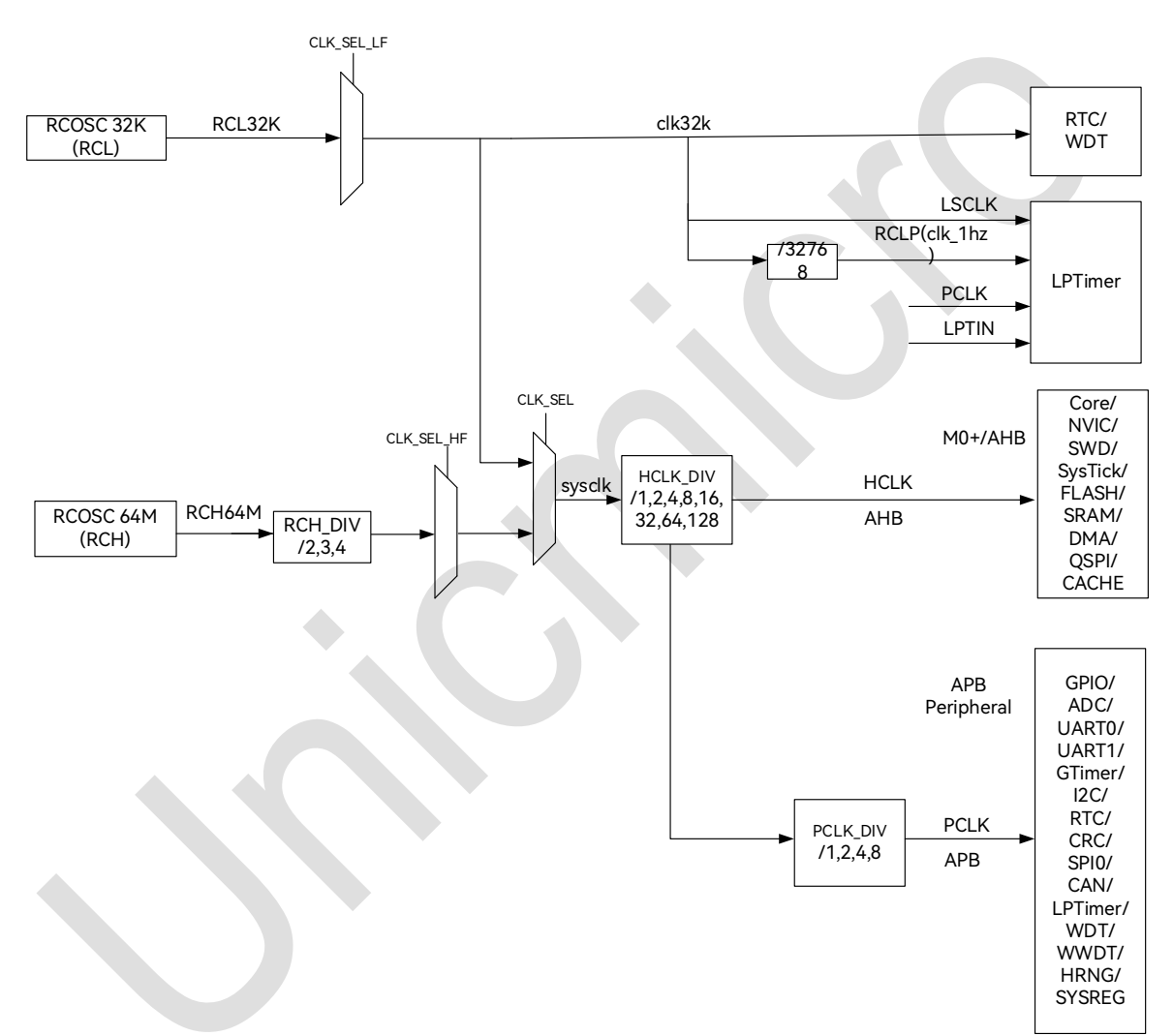


图 4-1：时钟模块框图

4.3 时钟选择

芯片共有 2 个系统时钟源：

- 1. 32MHz 高精度内部时钟 RCH，作为系统时钟源。
- 2. 32kHz 的内部时钟 RCL，作为低功耗时钟，可作为系统时钟源。

根据工作模式不同，采用不同时钟方案，通过配置系统控制寄存器0（SYSCTRL0）[14:12]位 CLK_SEL，CLK_SEL_HF和CLK_SEL_LF来选择系统时钟的来源。关系如表 4-2所示：

表 4-2：系统时钟选择

CLK_SEL	CLK_SEL_HF	系统时钟来源
0	0	RCH
CLK_SEL	CLK_SEL_LF	系统时钟来源
1	0	RCL

4.4 复位源

芯片有多个复位源，包括 POR 复位、RESETN 复位、WDT 复位、SOFT_RESETN 复位、模块软件复位、LVD 复位与 LOCKUP 复位。具体复位源如下表：

表 4-3：系统复位源

复位源	描述
内部模拟 POR 上电复位	复位所有
LVR 复位	
RESETN 复位	复位除 CPU DEBUG 逻辑外的所有
LOCKUP 复位	复位除 EFC 和 IO 相关以外的其它逻辑
LVD 复位	
WDT	
WWDT	
SOFT_RESETN	
各模块复位	复位对应 IP 模块

4.4.1 内部 POR 上电复位

内部上电复位 PORN：无条件复位整个芯片。

4.4.2 LVR 复位

LVR 复位：无条件复位整个芯片。

4.4.3 RESETN 复位

外部复位RESETN复位除CPU DEBUG逻辑外的整个芯片。在RESETN复位状态时，芯片可以连接SWD接口。RESETN引脚在上电完成后默认作为外部复位，但通过软件关闭外部复位功能。

4.4.4 WDT 复位

看门狗定时器复位除 EFC 和 IO 相关外的整个用户电路，缺省为释放状态。

当软件未能有效阻止超时事件时，看门狗定时器复位。该复位仅发生在软件无法正常执行，可能破坏数据时。在 CPU 处于 HALT 状态时，WDT 停止计数，不会产生复位信号。

4.4.5 WWDT 复位

复位除 EFC 和 IO 相关以外的其它逻辑。

4.4.6 SOFT_RESETN 复位

此复位由系统产生。系统可以通过写软复位重启，但不复位 EFC 控制器和 IO 相关设置。

4.4.7 模块软件复位

模块软件复位，通过软件复位各数字模块。

4.4.8 LOCKUP 复位

当系统连续发生两次 HardFault 时, CPU 会进入 LOCKUP 状态, 系统会产生 LOCKUP 复位。LOCKUP 复位除 EFC 外的其它逻辑。

4.4.9 LVD 复位

LVD 复位除 EFC 和 IO 相关以外的其它逻辑。

4.5 低功耗模式

芯片除正常工作模式外, 为了降低芯片的电流消耗, 提供三种低功耗模式: 休眠 (Sleep) 模式、深度休眠 (Deepsleep) 模式和停止 (Stop) 模式。

在休眠模式下, CM0+停止工作, 保留中断处理功能。其它外设等模块时钟和复位可由软件设置。休眠模式由 M0+特定指令 WFI/WFE 进入, 唤醒由中断触发。

深度休眠模式是休眠模式的升级, 在此模式下, CM0+停止运行, 高速时钟停止运行, 低功耗功能模块 (LPTIMER、RTC、WDT) 可以运行。深度休眠模式要先设置 CM0+内部的 Deepsleep 寄存器, 然后由 M0+特定指令 WFI/WFE 进入, 唤醒由中断触发。

停止模式下, 高速时钟和低速时钟均停止运行, 系统无任何运行的时钟, 一切外围模块均停止运行。上电复位信号有效, IO 状态保持, IO 中断有效, 所有寄存器, RAM 和 CPU 数据保存状态时的功耗; 停止模式要先设置 SYSREG 模块中 STOPMODE_SEL 寄存器和 CM0+内部的 Deepsleep 寄存器, 然后由 M0+特定指令 WFI/WFE 进入, 唤醒只能通过 GPIO 电平唤醒或者通过 LPTIMER 外部异步脉冲计数产生中断唤醒。

详细的描述如表 4-4 所示:

表 4-4: 低功耗模式

模式	模式描述	进入条件	退出条件
Sleep	CPU 大部分休眠 (包括 NVIC), WIC 不休眠; 软件可关闭各模块时钟。	1. 根据需要, 关闭各外设模块时钟, 仅留下需要监测中断事件的模块。 2. 执行 WFI/WFE 指令。	1. CM0+检测到中断或事件发生。 2. 进入中断服务程序清中断并返回。 3. 继续执行后续指令。
Deepsleep	CPU 大部分休眠 (包括 NVIC), WIC 不休眠; 高速时钟源关闭, 低速时钟 RCL 源运行	1. 根据需要, 关闭各外设模块时钟, 仅留下需要监测中断事件的模块。 2. 设置 CM0+内部的 DEEP SLEEP 寄存器。 3. 执行 WFI/WFE 指令。	1. CM0+检测到中断或事件发生。 2. 进入中断服务程序清中断并返回。 3. 继续执行后续指令。
Stop	关闭系统所有时钟	1. 根据需要, 设置 IO 唤醒或者 LPTIMER 唤醒的条件。 2. 设置 CM0+内部的 DEEP SLEEP 寄存器。 3. 设置 SYSREG 中的 STOPMODE_SEL寄存器。 4. 执行 WFI/WFE 指令。	1. 外部 IO 唤醒事件到来或者 LPTIMER 计数溢出。 2. CM0+检测到 IO 唤醒事件中断或者 LPTIMER 溢出中断发生。 3. 进入中断服务程序清中断并返回。 4. 继续执行后续指令。

低功耗模式的进入和唤醒条件阐述如下:

- Sleep, Deepsleep, Stop 模式进入条件都需要设置 SCB->SCR 寄存器, 调用 WFI/WFE; 三者模式的唤醒本质是都是需要产生中断或者事件发生。
- Sleep 模式下, 内部高速时钟 RCH (32MHz) 和内部低速时钟 RCL (32K) 没有关闭, 只要系统产生中断就可以唤醒退出。
- Deepsleep 模式下, RCH 时钟关闭了, RCL 时钟在工作, 所以只有工作在 RCL (32K) 时钟源的模块如 WDT、RTC、LPTIMER 可以产生中断唤醒退出, 以及 GPIO 电平/边沿模式, 可以在无时钟情况下产生中断唤醒退出。
- Stop 模式下, 所有时钟源关闭, 只能通过 GPIO 电平/边沿模式, 在无时钟情况下产生中断唤醒退出或者通过 LPTIMER 外部异步脉冲计数产生中断唤醒退出。

注：进入 Deepsleep 或 Stop 模式前，需禁止非唤醒源中断（即除了 wdt、gpio、lpuart、lptimer、rtc、lvd 外的中断），再配置芯片进入 Deepsleep 或 Stop 模式。芯片唤醒后，再使能非唤醒源中断。

4.5.1 Sleep 模式

进入 sleep 模式设置：

- SCB->SCR 的 bit2 配置为 0
- 调用 WFI/WFE 进入 sleep 模式

唤醒条件：中断唤醒。

4.5.2 Deepsleep 模式

进入 Deepsleep 模式设置，下面以 PB3 引脚低电平唤醒为例阐述配置流程：

1. 配置外围模块时钟控制寄存器 PERI_CLKEN，打开 GPIOB 时钟。
2. 配置外围模块复位控制寄存器 PERI_RESET，GPIOB 设置正常工作。
3. 配置 GPIO_DIR 寄存器，PB3 为输入。
4. 配置 REG_GPIO_IEN 寄存器，禁止 PB3 中断。
5. 配置 GPIO_IS 寄存器，触发模式为电平触发。
6. 配置 GPIO_IIEV 寄存器，低电平触发。
7. 配置 GPIO_IC 寄存器，清除 PB3 中断状态。
8. 配置 GPIO_IEN 寄存器，使能 PB3 中断。
9. 清除 GPIOB 中断，使能外部 GPIOB 中断。
10. SCB->SCR 的 bit2 配置为 1。
11. 调用 WFI/WFE 进入 Deepsleep 模式。

唤醒条件：WDT、RTC、LPTIMER 中断唤醒，GPIO PB3 中断唤醒。

4.5.3 Stop 模式

进入 Stop 模式设置，下面以 PB3 引脚低电平唤醒为例阐述配置流程：

1. 配置 STOPMODE_SEL 寄存器，使能 Stop mode 停止模式有效。
2. 配置外围模块时钟控制寄存器 PERI_CLKEN，打开 GPIOB 时钟。
3. 配置外围模块复位控制寄存器 PERI_RESET，GPIOB 设置正常工作。
4. 配置 GPIO_DIR 寄存器，PB3 为输入。
5. 配置 REG_GPIO_IEN 寄存器，禁止 PB3 中断。
6. 配置 GPIO_IS 寄存器，触发模式为电平触发。
7. 配置 GPIO_IEV 寄存器，低电平触发。
8. 配置 GPIO_IC 寄存器，清除 PB3 中断状态。
9. 配置 GPIO_IEN 寄存器，使能 PB3 中断。
10. 清除 GPIOB 中断，使能外部 GPIOB 中断。
11. SCB->SCR 的 bit2 配置为 1。
12. 调用 WFI/WFE 进入 Stop 模式。

唤醒条件：GPIO PB3 中断唤醒。

4.6 系统寄存器

SYSREG (SCU) 寄存器基地址：0x40002000

表 4-5：系统寄存器列表

偏置	名称	描述
0x000	SYSCTRL0	系统控制寄存器 0
0x008	SYSCTRL_PROTECT	系统控制保护寄存器
0x00C	OSC_CTRL	时钟控制寄存器
0x010	PERI_CLKEN	外设时钟门控寄存器
0x020	RESET_FLAG	复位标识寄存器
0x024	PERI_RESET	外围模块复位控制寄存器
0x028	EXT_RESET_CTRL	外部复位滤波控制寄存器

偏置	名称	描述
0x030	PA_SEL	端口 PA 功能配置寄存器, 只能外部复位和 POR 复位此寄存器
0x034	PB_SEL	端口 PB 功能配置寄存器, 只能外部复位和 POR 复位此寄存器
0x038	PC_SEL	端口 PC 功能配置寄存器, 只能外部复位和 POR 复位此寄存器
0x03C	PD_SEL	端口 PD 功能配置寄存器, 只能外部复位和 POR 复位此寄存器
0x040	PE_SEL	端口 PE 功能配置寄存器, 只能外部复位和 POR 复位此寄存器
0x044	PAD_ADS	端口数模配置寄存器, 只能外部复位和 POR 复位此寄存器
0x048	PAD_DR0	端口驱动能力配置寄存器 0, 只能外部复位和 POR 复位此寄存器
0x04C	PAD_DR1	端口驱动能力配置寄存器 1, 只能外部复位和 POR 复位此寄存器
0x050	PAD_PU0	端口上拉配置寄存器 0, 只能外部复位和 POR 复位此寄存器
0x054	PAD_PU1	端口上拉配置寄存器 1, 只能外部复位和 POR 复位此寄存器
0x058	PAD_PD0	端口下拉配置寄存器 0, 只能外部复位和 POR 复位此寄存器
0x05C	PAD_PD1	端口下拉配置寄存器 1, 只能外部复位和 POR 复位此寄存器
0x060	PAD_OD0	端口开漏输出配置寄存器 0, 只能外部复位和 POR 复位此寄存器
0x064	PAD_OD1	端口开漏输出配置寄存器 1, 只能外部复位和 POR 复位此寄存器
0x068	PAD_CS0	端口输入类型配置寄存器 0, 只能外部复位和 POR 复位此寄存器
0x06C	PAD_CS1	端口输入类型配置寄存器 1, 只能外部复位和 POR 复位此寄存器
0x070	PAD_IE0	端口输入配置寄存器 0, 只能外部复位和 POR 复位此寄存器
0x074	PAD_IE1	端口输入配置寄存器 1, 只能外部复位和 POR 复位此寄存器

偏置	名称	描述
0x078	PAD_SR0	端口速率配置寄存器 0, 只能外部复位和 POR 复位此寄存器
0x07C	PAD_SR1	端口速率配置寄存器 1, 只能外部复位和 POR 复位此寄存器
0x080	IOCTRL_PROTECT	IO 控制保护寄存器
0x084	LVD_CFG	LVD 控制寄存器
0x090	EXTRST_SEL	外部复位端口选择寄存器
0x094	STOPMODE_SEL	停止模式选择寄存器
0x098	REMAP_ADDR	REMAP 寄存器
0x09C	VECTOR_OFFSET	中断向量地址重映射寄存器
0x0A0	HRNG_CR	随机数控制寄存器
0x0A4	HRNG_SEED	随机数种子寄存器
0x0A8	HRNG_DATA	随机数数据寄存器
0x0B0	LVR_CFG	LVR 控制寄存器
0x0B4	VREF_CFG	VREF 控制寄存器
0x0CC	VREF_STATUS	VREF 状态寄存器

注：寄存器描述表中使用的缩写列表如下

- R: 只读
- W: 只写
- R/W: 可读可写
- R/W1C: 可读, 写 1 清零
- R/W0C: 可读, 写 0 清除
- RSV: Reserved 的缩写, 此寄存器位没有功能, 是保留位

4.6.1 系统控制寄存器 0/ SYSCTRL0 (偏移: 000H)

比特	名称	属性	复位值	描述
31:30	RSV	-	-	保留

比特	名称	属性	复位值	描述
29:27	CLKOUT_SEL	R/W	0	PC2 作为时钟输出时, 输出时钟选择: 000: HCLK_OUT (系统时钟) 010: RCL 时钟 100: PCLK_OUT 101: RTC_1Hz 110: RCH_DIV
26:24	CLK16M_DIV	R/W	3	RTC 16.384M 时钟分频: 01: RCH 经过选择后的时钟二分频 10: RCH 经过选择后的时钟三分频 11: RCH 经过选择后的时钟四分频 注: 此位不能写入 00
23	RSV	-	-	保留
22	LPTIMER_EXTIG_SEL	R/W	0	LPTIMER0 的 EXTIG (外部触发) 信号来源选择信号: 0: EXTIG 来自于外部引脚 1: EXTIG 来自于 RTC 的 1HZ 时钟输出
21	LPTIMER_LPTIN_SEL	R/W	0	LPTIMER0 的 LPTIN (外部时钟输入) 信号来源选择信号: 0: LPTIN 来自于外部引脚 1: LPTIN 来自于 RTC 的 1HZ 时钟输出
20	DMA_C7HS_G0GT_SEL	R/W	0	DMA 第七个握手信号选择: 0: 选择 GTIMER 的 TIMER0 的中断信号作为启动 DMA 搬移的信号。 1: 选择 GPIOA 的中断信号作为启动 DMA 搬移的信号
19	DMA_C7HS_EN	R/W	0	DMA 第七个握手信号使能。当使能此位后, GPIOA 或 GTIMER 的中断信号可以启动 DMA 搬移。 0: 不使能 1: 使能
18:17	RCH_DIV	R/W	1	RCH 时钟源分频设置位: 01: 二分频 10: 三分频 11: 四分频 注: 此位不能写入 00, 写入 00 硬件仍为二分频

比特	名称	属性	复位值	描述
16	SWD_WACK_EN	R/W	1	连接 ULINK 或者 JLINK 后, 在 Deepsleep 或者 Stop 模式下, 唤醒系统的使能位: 0: 在上述条件下, 不唤醒系统 1: 在上述条件下, 使用 NMI 中断唤醒系统
15	Wakeup_byRCH	R/W	0	从 Deep Sleep 唤醒后, system clock 来源选择: 0: 不改变 system clock 来源 1: system clock 来源为 RCH, 原时钟继续使用
14	CLK_SEL	R/W	0	系统时钟来源选择: 0: 高速时钟 CLK_SEL_HF 1: 低速时钟 CLK_SEL_LF
13	CLK_SEL_HF	R/W	0	0: 高频时钟选择内部高速时钟 RCH 1: 保留
12	CLK_SEL_LF	R/W	0	0: 低频时钟选择内部低速时钟 RCL 1: 保留
11	RSV	-	-	保留
10:9	PCLK_DIV	R/W	0	PCLK 分频选择: 00: HCLK 01: HCLK/2 10: HCLK/4 11: HCLK/8
8:6	HCLK_DIV	R/W	0	HCLK 分频选择: 000: SystemClk 001: SystemClk/2 010: SystemClk/4 011: SystemClk/8 100: SystemClk/16 101: SystemClk/32 110: SystemClk/64 111: SystemClk/128
5:3	RSV	-	-	保留
2	RCL_EN	R/W	1	内部低速时钟 RCL 使能控制: 0: 关闭 1: 使能
1	RSV	-	-	保留

比特	名称	属性	复位值	描述
0	RCH_EN	R/W	1	内部高速时钟 RCH 使能信号： 0：关闭 1：使能 注：当系统进入 Deepsleep，此高速时钟会自动关闭

4.6.2 系统控制保护寄存器/ SYSCTRL_PROTECT (偏移：008H)

比特	名称	属性	复位值	描述
31:0	SYSCTRL_PROTECT	R/W	0	寄存器 SYSCTRL0 写保护的寄存器。 给此寄存器写 0xA5A5_5A5A，启动寄存器 SYSCTRL0 的写使能。给此寄存器写其他值，关闭它们的写使能。SYSCTRL0 寄存器配置完后，它们的写使能会自动关闭 读取此寄存器返回 SYSCTRL0 寄存器的写使能状态。 0：写未使能 1：写已经使能

4.6.3 时钟控制寄存器/ OSC_CTRL (偏移：00CH)

比特	名称	属性	复位值	描述
31:29	RSV	-	-	保留
28:16	WAKP_delay	R/W	13'h138	系统从 Deepsleep / Stop 模式唤醒，CLK 给出的时间延时。此寄存器的时间为系统时钟计数的时钟周期
15:11	RSV	-	-	保留
10	RCH_stable	R	1	内部高速时钟 RCH 稳定标志位： 0：RCH 未稳定，不可以被内部电路使用 1：RCH 已经稳定，可以被内部电路使用
9:8	RCH_startup	R/W	00	内部高速时钟 RCH 稳定时间选择： 00：4 个周期 01：16 个周期 10：64 个周期 11：256 个周期
7:3	RSV	-	-	保留

比特	名称	属性	复位值	描述
2	RCL_stable	R	1	内部低速时钟 RCL 稳定标志位: 0: RCL 未稳定, 不可以被内部电路使用 1: RCL 已经稳定, 可以被内部电路使用
1:0	RCL_startup	R/W	00	内部低速时钟 RCL 稳定时间选择: 00: 4 个周期 01: 16 个周期 10: 64 个周期 11: 256 个周期

4.6.4 外围模块时钟寄存器/ PERI_CLKEN (偏移: 010H)

比特	名称	属性	复位值	描述
31:29	RSV	-	-	保留
28	GPIOE_CLKEN	RW	0	GPIOE 模块时钟使能: 0: 关闭 1: 使能
27	RSV	-	-	保留
26	CACHE_CLKEN	RW	0	CACHE 模块时钟使能: 0: 关闭 1: 使能
25	WWDT_CLKEN	RW	0	WWDT 模块时钟使能: 0: 关闭 1: 使能
24	CAN_CLKEN	RW	0	CAN 模块时钟使能: 0: 关闭 1: 使能
23	QSPI_CLKEN	RW	0	QSPI 模块时钟使能: 0: 关闭 1: 使能
22	DMA_CLKEN	RW	0	DMA 模块时钟使能: 0: 关闭 1: 使能
21	SPI1_CLKEN	RW	0	SPI1 模块时钟使能: 0: 关闭 1: 使能

比特	名称	属性	复位值	描述
20	EFC_CLKEN	RW	1	EFC 模块时钟使能: 0: 关闭 1: 使能
19	GPIOD_CLKEN	RW	0	GPIOD 模块时钟使能: 0: 关闭 1: 使能
18	GPIOC_CLKEN	RW	0	GPIOC 模块时钟使能: 0: 关闭 1: 使能
17	GPIOB_CLKEN	RW	0	GPIOB 模块时钟使能: 0: 关闭 1: 使能
16	GPIOA_CLKEN	RW	0	GPIOA 模块时钟使能: 0: 关闭 1: 使能
15	I2C_CLKEN	RW	0	I2C 模块时钟使能: 0: 关闭 1: 使能
14	ADC_CLKEN	RW	0	ADC 控制器模块时钟使能: 0: 关闭 1: 使能
13	RTC_CLKEN	RW	1	RTC 模块时钟使能: 0: 关闭 1: 使能
12	WDT_CLKEN	RW	0	WDT 模块时钟使能: 0: 关闭 1: 使能
11	CRC_CLKEN	RW	0	CRC 模块时钟使能: 0: 关闭 1: 使能
10	UART1_CLKEN	RW	0	UART1 模块时钟使能: 0: 关闭 1: 使能
9	GTIM0_CLKEN	RW	0	GTime0 模块时钟使能: 0: 关闭 1: 使能

比特	名称	属性	复位值	描述
8	LPTIM0_CLKEN	RW	0	LPTimer0 模块时钟使能: 0: 关闭 1: 使能
7	RSV	-	-	保留
6	LPTIM2_CLKEN	RW	0	LPTimer2 模块时钟使能: 0: 关闭 1: 使能
5	LPTIM1_CLKEN	RW	0	LPTimer1 模块时钟使能: 0: 关闭 1: 使能
4	SPI0_CLKEN	RW	0	SPI0 模块时钟使能: 0: 关闭 1: 使能
3	GTIM2_CLKEN	RW	0	GTime 2 模块时钟使能: 0: 关闭 1: 使能
2	GTIM1_CLKEN	RW	0	GTime 1 模块时钟使能: 0: 关闭 1: 使能
1	RSV	RW	0	保留
0	UART0_CLKEN	RW	0	UART0 模块时钟使能: 0: 关闭 1: 使能

4.6.5 复位标识寄存器/ RESET_FLAG (偏移: 020H)

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7	SYS_RESET_REQ_FLAG	R/W1C	0	CPU 复位状态。需要软件初始化和清除: 0: 无复位发生 1: Cortex M0+ 系统复位发生 此位只能被 PORN 和外部 RESETN 复位, 写 1 清 0

比特	名称	属性	复位值	描述
6	LOCKUP_RSTN_FLAG	R/W1C	0	CPU 死锁复位状态。需要软件初始化和清除： 0: 无复位发生 1: Lockup 复位发生 此位只能被 PORN 和外部 RESETN 复位，写 1 清 0
5	LVD_RSTN_FLAG	R/W1C	0	低电压复位状态。需要软件初始化和清除： 0: 无复位发生 1: LVD 复位发生 此位只能被 PORN 和外部 RESETN 复位，写 1 清 0
4	SOFT_RSTN_FLAG	R/W1C	0	软件复位状态。需要软件初始化和清除： 0: 无复位发生 1: 写 REMAP_ADDR 寄存器的 SOFT_RESETN 位复位发生 此位只能被 PORN 和外部 RESETN 复位，写 1 清 0
3	WDT_FLAG	R/W1C	0	看门狗复位状态。需要软件初始化和清除： 0: 无复位发生 1: WDT 复位发生 此位只能被 PORN 和外部 RESETN 复位，写 1 清 0
2	RESETN_FLAG	R/W1C	0	外部复位状态。需要软件初始化和清除： 0: 无复位发生 1: 外部复位发生 此位只能被 PORN 复位，写 1 清 0
1:0	RSV	-	-	保留

注：这些寄存器只能被 PORN 复位

4.6.6 外围模块复位控制寄存器/ PERI_RESET (偏移: 024H)

比特	名称	属性	复位值	描述
31:29	RSV	-	-	保留

比特	名称	属性	复位值	描述
28	GPIOE_RESET	R/W	0	GPIOE 模块复位使能: 0: 模块处于复位状态 1: 正常工作
27	RSV	-	-	保留
26	CACHE_RESET	R/W	1	CACHE 模块复位使能: 0: 模块处于复位状态 1: 正常工作
25	WWDT_RESET	R/W	0	WWDT 模块复位使能: 0: 模块处于复位状态 1: 正常工作
24	CAN_RESET	R/W	0	CAN 控制器模块复位使能: 0: 模块处于复位状态 1: 正常工作
23	QSPI_RESET	R/W	0	QSPI 控制器模块复位使能: 0: 模块处于复位状态 1: 正常工作
22	DMA_RESET	R/W	0	DMA 控制器模块复位使能: 0: 模块处于复位状态 1: 正常工作
21	SPI1_RESET	R/W	0	SPI1 控制器模块复位使能: 0: 模块处于复位状态 1: 正常工作
20	RSV	-	-	保留
19	GPIOD_RESET	R/W	0	GPIOD 模块复位使能: 0: 模块处于复位状态 1: 正常工作
18	GPIOC_RESET	R/W	0	GPIOC 模块复位使能: 0: 模块处于复位状态 1: 正常工作
17	GPIOB_RESET	R/W	0	GPIOB 模块复位使能: 0: 模块处于复位状态 1: 正常工作
16	GPIOA_RESET	R/W	0	GPIOA 模块复位使能: 0: 模块处于复位状态 1: 正常工作

比特	名称	属性	复位值	描述
15	I2C_RESET	R/W	0	I2C 模块复位使能: 0: 模块处于复位状态 1: 正常工作
14	ADC_RESET	R/W	0	ADC 控制器模块复位使能: 0: 模块处于复位状态 1: 正常工作
13	RTC_RESET	R/W	1	RTC 模块复位使能: 0: 模块处于复位状态 1: 正常工作
12	WDT_RESET	R/W	0	WDT 模块复位使能: 0: 模块处于复位状态 1: 正常工作
11	CRC_RESET	R/W	0	CRC 模块复位使能: 0: 模块处于复位状态 1: 正常工作
10	UART1_RESET	R/W	0	UART1 模块复位使能: 0: 模块处于复位状态 1: 正常工作
9	GTIM0_RESET	R/W	0	GTime0 模块复位使能: 0: 模块处于复位状态 1: 正常工作
8	LPTIM0_RESET	R/W	0	LPTimer0 模块复位使能: 0: 模块处于复位状态 1: 正常工作
7	RSV	-	-	保留
6	LPTIM2_RESET	R/W	0	LPTimer2 模块复位使能: 0: 模块处于复位状态 1: 正常工作
5	LPTIM1_RESET	R/W	0	LPTimer1 模块复位使能: 0: 模块处于复位状态 1: 正常工作
4	SPIO_RESET	R/W	0	SPIO 控制器模块复位使能: 0: 模块处于复位状态 1: 正常工作
3	GTIM2_RESET	R/W	0	GTime 2 模块复位使能: 0: 模块处于复位状态 1: 正常工作

比特	名称	属性	复位值	描述
2	GTIM1_RESET	R/W	0	GTIM1 模块复位使能: 0: 模块处于复位状态 1: 正常工作
1	RSV	R/W	0	保留
0	UART0_RESET	R/W	0	UART0 模块复位使能: 0: 模块处于复位状态 1: 正常工作

4.6.7 外部复位滤波控制寄存器/ EXT_RESET_CTRL (偏移: 028H)

比特	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	EXT_FILTER_EN	R/W	0	外部复位滤波使能位: 0: 外部复位滤波禁止 1: 外部复位滤波使能

4.6.8 端口 PA 功能配置寄存器/ PA_SEL (偏移: 030H)

比特	名称	属性	复位值	描述
31:11	RSV	-	-	保留
10:8	PA2_SEL	R/W	3'b000	端口 PA2 功能选择: 3'b000: GPIO PA2 3'b001: NC 3'b010: UART1_RX 3'b011: UART0_RX 3'b100: NC 3'b101: I2C_SCL 3'b110: I2C_SDA 3'b111: NC
7:0	RSV	-	-	保留

4.6.9 端口 PB 功能配置寄存器/ PB_SEL (偏移: 034H)

比特	名称	属性	复位值	描述
31	RSV	-	-	保留
30:28	PB7_SEL	R/W	3'b000	端口 PB7 功能选择: 3'b000: GPIO PB7 3'b001: SPI0_SCK 3'b010: LPTIM0_OUT 3'b011: LPTIM2_EXT 3'b100: RTC_TAMP0 3'b101: GTIM2_CHN 3'b110: QSPI_HOLD 3'b111: GTIM2_BK
27:19	RSV	-	-	保留
18:16	PB4_SEL	R/W	3'b000	端口 PB4 功能选择: 3'b000: GPIO PB4 3'b001: SPI0_MOSI 3'b010: NC 3'b011: UART1_CTS 3'b100: NC 3'b101: LPTIM0_OUT 3'b110: CAN_TX 3'b111: QSPI_MOSI
15	RSV	-	-	保留
14:12	PB3_SEL	R/W	3'b000	端口 PB3 功能选择: 3'b000: GPIO PB3 3'b001: NC 3'b010: NC 3'b011: LPTIM0_EXT 3'b100: CAN_RX 3'b101: RTC_TAMP1 3'b110: LPTIM2_IN 3'b111: GTIM0_BK
11	RSV	-	-	保留

比特	名称	属性	复位值	描述
10:8	PB2_SEL	R/W	3'b000	端口 PB2 功能选择: 3'b000: GPIO PB2 3'b001: NC 3'b010: SPI0_CSN0 3'b011: GTIM0_CH 3'b100: SPI0_MOSI 3'b101: LPTIM1_IN 3'b110: GTIM2_CHN 3'b111: QSPI_HOLD
7	RSV	-	-	保留
6:4	PB1_SEL	R/W	3'b000	端口 PB1 功能选择: 3'b000: GPIO PB1 3'b001: SPI1_CSN0 3'b010: GTIM1_CHN 3'b011: LPTIM0_EXT 3'b100: LPTIM0_IN 3'b101: NC 3'b110: I2C_SCL 3'b111: NC
3:0	RSV	-	-	保留

4.6.10 端口 PC 功能配置寄存器/ PC_SEL (偏移: 038H)

比特	名称	属性	复位值	描述
31:27	RSV	-	-	保留
26:24	PC6_SEL	R/W	3'b001	端口 PC6 功能选择: 3'b000: GPIO PC6 3'b001: SWCLK 3'b010: UART1_TX 3'b011: NC 3'b100: NC 3'b101: NC 3'b110: LPTIM0_OUT 3'b111: NC
23	RSV	-	-	保留

比特	名称	属性	复位值	描述
22:20	PC5_SEL	R/W	3'b001	端口 PC5 功能选择: 3'b000: GPIO PC5 3'b001: SWIO 3'b010: NC 3'b011: LPTIM0_EXT 3'b100: I2C_SDA 3'b101: NC 3'b110: NC 3'b111: NC
19	RSV	-	-	保留
18:16	PC4_SEL	R/W	3'b000	端口 PC4 功能选择: 3'b000: GPIO PC4 3'b001: UART1_RTS 3'b010: NC 3'b011: UART0_RX 3'b100: SPI0_MI1 3'b101: NC 3'b110: LPTIM2_EXT 3'b111: QSPI_WP
15:11	RSV	-	-	保留
10:8	PC2_SEL	R/W	3'b010	端口 PC2 功能选择: 3'b000: GPIO PC2 3'b001: I2C_SDA 3'b010: UART1_RX 3'b011: NC 3'b100: SPI0_CSN1 3'b101: GTIM2_CH 3'b110: LPTIM1_IN 3'b111: CLKOUT
7	RSV	-	-	保留

比特	名称	属性	复位值	描述
6:4	PC1_SEL	R/W	3'b010	端口 PC1 功能选择: 3'b000: GPIO PC1 3'b001: I2C_SCL 3'b010: UART1_TX 3'b011: NC 3'b100: SPI0_MISO/SPI0_TRI_MO 3'b101: GTIM1_CH 3'b110: LPTIM0_OUT 3'b111: CAN_RX
3:0	RSV	-	-	保留

4.6.11 端口 PD 功能配置寄存器/ PD_SEL (偏移: 03CH)

比特	名称	属性	复位值	描述
31:27	RSV	-	-	保留
26:24	PD6_SEL	R/W	3'b000	端口 PD6 功能选择: 3'b000: GPIO PD6 3'b001: UART0_TX 3'b010: SPI0_MISO/SPI0_TRI_MO 3'b011: LPTIM1_EXT 3'b100: CAN_TX 3'b101: QSPI_MISO 3'b110: SPI0_CSN0 3'b111: LPTIM2_OUT
23	RSV	-	-	保留
22:20	PD5_SEL	R/W	3'b000	端口 PD5 功能选择: 3'b000: GPIO PD5 3'b001: I2C_SDA 3'b010: LPTIM1_IN 3'b011: UART1_RX 3'b100: NC 3'b101: GTIM0_CHN 3'b110: CAN_RX 3'b111: NC
19	RSV	-	-	保留

比特	名称	属性	复位值	描述
18:16	PD4_SEL	R/W	3'b000	端口 PD4 功能选择: 3'b000: GPIO PD4 3'b001: UART1_TX 3'b010: I2C_SCL 3'b011: NC 3'b100: NC 3'b101: SPI0_SCK 3'b110: GTIM2_CH 3'b111: LPTIM0_EXT
15	RSV	-	-	保留
14:12	RF_MOSI_SEL	R/W	0	端口 RF_MOSI 功能选择: 3'b001: SPI1_MOSI
11	RSV	-	-	保留
10:8	RF_MISO_SEL	R/W	0	端口 RF_MISO 功能选择: 3'b001: SPI1_MISO
7	RSV	-	-	保留
6:4	RF_CLK_SEL	R/W	0	端口 RF_CLK 功能选择: 3'b001: SPI1_SCK
3	RSV	-	-	保留
2:0	PD0_SEL	R/W	3'b000	端口 PD0 功能选择: 3'b000: GPIO PD0 3'b001: SPI1_CSN0 3'b010: GTIM0_CH 3'b011: UART1_RX 3'b100: LPTIM1_IN 3'b101: RTC_TAMP0 3'b110: GTIM2_CHN 3'b111: QSPI_WP

4.6.12 端口 PE 功能配置寄存器/ PE_SEL (偏移: 040H)

比特	名称	属性	复位值	描述
31:19	RSV	-	-	保留

比特	名称	属性	复位值	描述
18:16	PE4_SEL	R/W	3'b000	端口 PE4 功能选择: 3'b000: GPIO PE4 3'b001: QSPI_SCK 3'b010: SPI0_MISO 3'b011: I2C_SCL 3'b100: NC 3'b101: GTIM2_BK 3'b110: NC 3'b111: NC
15:11	RSV	-	-	保留
10:8	PE2_SEL	R/W	3'b000	端口 PE2 功能选择: 3'b000: GPIO PE2 3'b001: QSPI_CSN 3'b010: SPI0_MI1 3'b011: NC 3'b100: UART1_CTS 3'b101: GTIM0_BK 3'b110: NC 3'b111: NC
7:0	RSV	-	-	保留

4.6.13 端口数模配置寄存器/ PAD_ADS (偏移: 044H)

比特	名称	属性	复位值	描述
31	RSV	-	-	保留
30	PD6_ADS	R/W	0	端口 PD6 数模配置寄存器: 0: 配置为数字接口 1: 配置为模拟接口
29	PD5_ADS	R/W	0	端口 PD5 数模配置寄存器: 0: 配置为数字接口 1: 配置为模拟接口
28	PD4_ADS	R/W	0	端口 PD4 数模配置寄存器: 0: 配置为数字接口 1: 配置为模拟接口

比特	名称	属性	复位值	描述
27	RF_MOSI_ADS	R/W	0	端口 RF_MOSI 数模配置寄存器: 0: 配置为数字接口 1: 配置为模拟接口
26	RF_MISO_ADS	R/W	0	端口 RF_MISO 数模配置寄存器: 0: 配置为数字接口 1: 配置为模拟接口
25	RF_CLK_ADS	R/W	0	端口 RF_CLK 数模配置寄存器: 0: 配置为数字接口 1: 配置为模拟接口
24	PD0_ADS	R/W	0	端口 PD0 数模配置寄存器: 0: 配置为数字接口 1: 配置为模拟接口
23:21	RSV	-	-	保留
20	PC4_ADS	R/W	0	端口 PC4 数模配置寄存器: 0: 配置为数字接口 1: 配置为模拟接口
19:13	RSV	R/W	0	保留
12	PB4_ADS	R/W	0	端口 PB4 数模配置寄存器: 0: 配置为数字接口 1: 配置为模拟接口
11	PB3_ADS	R/W	0	端口 PB3 数模配置寄存器: 0: 配置为数字接口 1: 配置为模拟接口
10	PB2_ADS	R/W	0	端口 PB2 数模配置寄存器: 0: 配置为数字接口 1: 配置为模拟接口
9	PB1_ADS	R/W	0	端口 PB1 数模配置寄存器: 0: 配置为数字接口 1: 配置为模拟接口
8:0	RSV	-	-	保留

4.6.14 端口驱动能力配置寄存器 0/ PAD_DR0 (偏移: 048H)

比特	名称	属性	复位值	描述
31	RSV	-	-	保留

比特	名称	属性	复位值	描述
30	PD6_DR	R/W	0	端口 PD6 驱动能力配置寄存器: 0: 高驱动能力 1: 低驱动能力
29	PD5_DR	R/W	0	端口 PD5 驱动能力配置寄存器: 0: 高驱动能力 1: 低驱动能力
28	PD4_DR	R/W	0	端口 PD4 驱动能力配置寄存器: 0: 高驱动能力 1: 低驱动能力
27	RF_MOSI_DR	R/W	0	端口 RF_MOSI 驱动能力配置寄存器: 0: 高驱动能力 1: 低驱动能力
26	RF_MISO_DR	R/W	0	端口 RF_MISO 驱动能力配置寄存器: 0: 高驱动能力 1: 低驱动能力
25	RF_CLK_DR	R/W	0	端口 RF_CLK 驱动能力配置寄存器: 0: 高驱动能力 1: 低驱动能力
24	PD0_DR	R/W	0	端口 PD0 驱动能力配置寄存器: 0: 高驱动能力 1: 低驱动能力
23	RSV	-	-	保留
22	PC6_DR	R/W	0	端口 PC6 驱动能力配置寄存器: 0: 高驱动能力 1: 低驱动能力
21	PC5_DR	R/W	0	端口 PC5 驱动能力配置寄存器: 0: 高驱动能力 1: 低驱动能力
20	PC4_DR	R/W	0	端口 PC4 驱动能力配置寄存器: 0: 高驱动能力 1: 低驱动能力
19	RSV	-	-	保留
18	PC2_DR	R/W	0	端口 PC2 驱动能力配置寄存器: 0: 高驱动能力 1: 低驱动能力

比特	名称	属性	复位值	描述
17	PC1_DR	R/W	0	端口 PC1 驱动能力配置寄存器: 0: 高驱动能力 1: 低驱动能力
16	RSV	-	-	保留
15	PB7_DR	R/W	0	端口 PB7 驱动能力配置寄存器: 0: 高驱动能力 1: 低驱动能力
14:13	RSV	-	-	保留
12	PB4_DR	R/W	0	端口 PB4 驱动能力配置寄存器: 0: 高驱动能力 1: 低驱动能力
11	PB3_DR	R/W	0	端口 PB3 驱动能力配置寄存器: 0: 高驱动能力 1: 低驱动能力
10	PB2_DR	R/W	0	端口 PB2 驱动能力配置寄存器: 0: 高驱动能力 1: 低驱动能力
9	PB1_DR	R/W	0	端口 PB1 驱动能力配置寄存器: 0: 高驱动能力 1: 低驱动能力
8:3	RSV	-	-	保留
2	PA2_DR	R/W	0	端口 PA2 驱动能力配置寄存器: 0: 高驱动能力 1: 低驱动能力
1:0	RSV	-	-	保留

4.6.15 端口驱动能力配置寄存器 1/ PAD_DR1 (偏移: 04CH)

比特	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4	PE4_DR	R/W	0	端口 PE4 驱动能力配置寄存器: 0: 高驱动能力 1: 低驱动能力
3	RSV	-	-	保留

比特	名称	属性	复位值	描述
2	PE2_DR	R/W	0	端口 PE2 驱动能力配置寄存器: 0: 高驱动能力 1: 低驱动能力
1:0	RSV	-	-	保留

4.6.16 端口上拉配置寄存器 0/ PAD_PU0 (偏移: 050H)

比特	名称	属性	复位值	描述
31	RSV	-	-	保留
30	PD6_PU	R/W	0	端口 PD6 上拉配置寄存器: 0: 禁止 1: 使能
29	PD5_PU	R/W	0	端口 PD5 上拉配置寄存器: 0: 禁止 1: 使能
28	PD4_PU	R/W	0	端口 PD4 上拉配置寄存器: 0: 禁止 1: 使能
27	RF_MOSI_PU	R/W	0	端口 RF_MOSI 上拉配置寄存器: 0: 禁止 1: 使能
26	RF_MISO_PU	R/W	0	端口 RF_MISO 上拉配置寄存器: 0: 禁止 1: 使能
25	RF_CLK_PU	R/W	0	端口 RF_CLK 上拉配置寄存器: 0: 禁止 1: 使能
24	PD0_PU	R/W	0	端口 PD0 上拉配置寄存器: 0: 禁止 1: 使能
23	RSV	-	-	保留
22	PC6_PU	R/W	1	端口 PC6 上拉配置寄存器: 0: 禁止 1: 使能

比特	名称	属性	复位值	描述
21	PC5_PU	R/W	1	端口 PC5 上拉配置寄存器: 0: 禁止 1: 使能
20	PC4_PU	R/W	0	端口 PC4 上拉配置寄存器: 0: 禁止 1: 使能
19	RSV	-	-	保留
18	PC2_PU	R/W	0	端口 PC2 上拉配置寄存器: 0: 禁止 1: 使能
17	PC1_PU	R/W	0	端口 PC1 上拉配置寄存器: 0: 禁止 1: 使能
16	RSV	-	-	保留
15	PB7_PU	R/W	0	端口 PB7 上拉配置寄存器: 0: 禁止 1: 使能
14:13	RSV	-	-	保留
12	PB4_PU	R/W	0	端口 PB4 上拉配置寄存器: 0: 禁止 1: 使能
11	PB3_PU	R/W	0	端口 PB3 上拉配置寄存器: 0: 禁止 1: 使能
10	PB2_PU	R/W	0	端口 PB2 上拉配置寄存器: 0: 禁止 1: 使能
9	PB1_PU	R/W	0	端口 PB1 上拉配置寄存器: 0: 禁止 1: 使能
8:3	RSV	-	-	保留
2	PA2_PU	R/W	1	端口 PA2 上拉配置寄存器: 0: 禁止 1: 使能
1:0	RSV	-	-	保留

4.6.17 端口上拉配置寄存器 1/ PAD_PU1 (偏移: 054H)

比特	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4	PE4_PU	R/W	0	端口 PE4 上拉配置寄存器: 0: 禁止 1: 使能
3	RSV	-	-	保留
2	PE2_PU	R/W	0	端口 PE2 上拉配置寄存器: 0: 禁止 1: 使能
1:0	RSV	-	-	保留

4.6.18 端口下拉配置寄存器 0/ PAD_PD0 (偏移: 058H)

比特	名称	属性	复位值	描述
31	RSV	-	-	保留
30	PD6_PD	R/W	0	端口 PD6 下拉配置寄存器: 0: 禁止 1: 使能
29	PD5_PD	R/W	0	端口 PD5 下拉配置寄存器: 0: 禁止 1: 使能
28	PD4_PD	R/W	0	端口 PD4 下拉配置寄存器: 0: 禁止 1: 使能
27	RF_MOSI_PD	R/W	0	端口 RF_MOSI 下拉配置寄存器: 0: 禁止 1: 使能
26	RF_MISO_PD	R/W	0	端口 RF_MISO 下拉配置寄存器: 0: 禁止 1: 使能
25	RF_CLK_PD	R/W	0	端口 RF_CLK 下拉配置寄存器: 0: 禁止 1: 使能

比特	名称	属性	复位值	描述
24	PD0_PD	R/W	0	端口 PD0 下拉配置寄存器: 0: 禁止 1: 使能
23	RSV	-	-	保留
22	PC6_PD	R/W	0	端口 PC6 下拉配置寄存器: 0: 禁止 1: 使能
21	PC5_PD	R/W	0	端口 PC5 下拉配置寄存器: 0: 禁止 1: 使能
20	PC4_PD	R/W	0	端口 PC4 下拉配置寄存器: 0: 禁止 1: 使能
19	RSV	-	-	保留
18	PC2_PD	R/W	0	端口 PC2 下拉配置寄存器: 0: 禁止 1: 使能
17	PC1_PD	R/W	0	端口 PC1 下拉配置寄存器: 0: 禁止 1: 使能
16	RSV	-	-	保留
15	PB7_PD	R/W	0	端口 PB7 下拉配置寄存器: 0: 禁止 1: 使能
14:13	RSV	-	-	保留
12	PB4_PD	R/W	0	端口 PB4 下拉配置寄存器: 0: 禁止 1: 使能
11	PB3_PD	R/W	0	端口 PB3 下拉配置寄存器: 0: 禁止 1: 使能
10	PB2_PD	R/W	0	端口 PB2 下拉配置寄存器: 0: 禁止 1: 使能
9	PB1_PD	R/W	0	端口 PB1 下拉配置寄存器: 0: 禁止 1: 使能

比特	名称	属性	复位值	描述
8:3	RSV	-	-	保留
2	PA2_PD	R/W	0	端口 PA2 下拉配置寄存器: 0: 禁止 1: 使能
1:0	RSV	-	-	保留

4.6.19 端口下拉配置寄存器 1/ PAD_PD1 (偏移: 05CH)

比特	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4	PE4_PD	R/W	0	端口 PE4 下拉配置寄存器: 0: 禁止 1: 使能
3	RSV	-	-	保留
2	PE2_PD	R/W	0	端口 PE2 下拉配置寄存器: 0: 禁止 1: 使能
1:0	RSV	-	-	保留

4.6.20 端口开漏输出配置寄存器 0/ PAD_OD0 (偏移: 060H)

比特	名称	属性	复位值	描述
31	RSV	-	-	保留
30	PD6_OD	R/W	0	端口 PD6 开漏输出配置寄存器: 0: 禁止 1: 使能
29	PD5_OD	R/W	0	端口 PD5 开漏输出配置寄存器: 0: 禁止 1: 使能
28	PD4_OD	R/W	0	端口 PD4 开漏输出配置寄存器: 0: 禁止 1: 使能
27	RF_MOSI_OD	R/W	0	端口 RF_MOSI 开漏输出配置寄存器: 0: 禁止 1: 使能

比特	名称	属性	复位值	描述
26	RF_MISO_OD	R/W	0	端口 RF_MISO 开漏输出配置寄存器: 0: 禁止 1: 使能
25	RF_CLK_OD	R/W	0	端口 RF_CLK 开漏输出配置寄存器: 0: 禁止 1: 使能
24	PD0_OD	R/W	0	端口 PD0 开漏输出配置寄存器: 0: 禁止 1: 使能
23	RSV	-	-	保留
22	PC6_OD	R/W	0	端口 PC6 开漏输出配置寄存器: 0: 禁止 1: 使能
21	PC5_OD	R/W	0	端口 PC5 开漏输出配置寄存器: 0: 禁止 1: 使能
20	PC4_OD	R/W	0	端口 PC4 开漏输出配置寄存器: 0: 禁止 1: 使能
19	RSV	-	-	保留
18	PC2_OD	R/W	0	端口 PC2 开漏输出配置寄存器: 0: 禁止 1: 使能
17	PC1_OD	R/W	0	端口 PC1 开漏输出配置寄存器: 0: 禁止 1: 使能
16	RSV	-	-	保留
15	PB7_OD	R/W	0	端口 PB7 开漏输出配置寄存器: 0: 禁止 1: 使能
14:13	RSV	-	-	保留
12	PB4_OD	R/W	0	端口 PB4 开漏输出配置寄存器: 0: 禁止 1: 使能
11	PB3_OD	R/W	0	端口 PB3 开漏输出配置寄存器: 0: 禁止 1: 使能

比特	名称	属性	复位值	描述
10	PB2_OD	R/W	0	端口 PB2 开漏输出配置寄存器: 0: 禁止 1: 使能
9	PB1_OD	R/W	0	端口 PB1 开漏输出配置寄存器: 0: 禁止 1: 使能
8:3	RSV	-	-	保留
2	PA2_OD	R/W	0	端口 PA2 开漏输出配置寄存器: 0: 禁止 1: 使能
1:0	RSV	-	-	保留

4.6.21 端口开漏输出配置寄存器 1/ PAD_OD1 (偏移: 064H)

比特	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4	PE4_OD	R/W	0	端口 PE4 开漏输出配置寄存器: 0: 禁止 1: 使能
3	RSV	-	-	保留
2	PE2_OD	R/W	0	端口 PE2 开漏输出配置寄存器: 0: 禁止 1: 使能
1:0	RSV	-	-	保留

4.6.22 端口输入类型配置寄存器 0/ PAD_CS0 (偏移: 068H)

比特	名称	属性	复位值	描述
31	RSV	R/W	1	保留
30	PD6_CS	R/W	1	端口 PD6 输入类型配置寄存器: 0: Schmitt input buffer 1: CMOS input buffer
29	PD5_CS	R/W	1	端口 PD5 输入类型配置寄存器: 0: Schmitt input buffer 1: CMOS input buffer

比特	名称	属性	复位值	描述
28	PD4_CS	R/W	1	端口 PD4 输入类型配置寄存器: 0: Schmitt input buffer 1: CMOS input buffer
27	RF_MOSI_CS	R/W	1	端口 RF_MOSI 输入类型配置寄存器: 0: Schmitt input buffer 1: CMOS input buffer
26	RF_MISO_CS	R/W	1	端口 RF_MISO 输入类型配置寄存器: 0: Schmitt input buffer 1: CMOS input buffer
25	RF_CLK_CS	R/W	1	端口 RF_CLK 输入类型配置寄存器: 0: Schmitt input buffer 1: CMOS input buffer
24	PD0_CS	R/W	1	端口 PD0 输入类型配置寄存器: 0: Schmitt input buffer 1: CMOS input buffer
23	RSV	-	-	保留
22	PC6_CS	R/W	1	端口 PC6 输入类型配置寄存器: 0: Schmitt input buffer 1: CMOS input buffer
21	PC5_CS	R/W	1	端口 PC5 输入类型配置寄存器: 0: Schmitt input buffer 1: CMOS input buffer
20	PC4_CS	R/W	1	端口 PC4 输入类型配置寄存器: 0: Schmitt input buffer 1: CMOS input buffer
19	RSV	-	-	保留
18	PC2_CS	R/W	1	端口 PC2 输入类型配置寄存器: 0: Schmitt input buffer 1: CMOS input buffer
17	PC1_CS	R/W	1	端口 PC1 输入类型配置寄存器: 0: Schmitt input buffer 1: CMOS input buffer
16	RSV	-	-	保留
15	PB7_CS	R/W	1	端口 PB7 输入类型配置寄存器: 0: Schmitt input buffer 1: CMOS input buffer
14:13	RSV	-	-	保留

比特	名称	属性	复位值	描述
12	PB4_CS	R/W	1	端口 PB4 输入类型配置寄存器: 0: Schmitt input buffer 1: CMOS input buffer
11	PB3_CS	R/W	1	端口 PB3 输入类型配置寄存器: 0: Schmitt input buffer 1: CMOS input buffer
10	PB2_CS	R/W	1	端口 PB2 输入类型配置寄存器: 0: Schmitt input buffer 1: CMOS input buffer
9	PB1_CS	R/W	1	端口 PB1 输入类型配置寄存器: 0: Schmitt input buffer 1: CMOS input buffer
8:3	RSV	-	-	保留
2	PA2_CS	R/W	1	端口 PA2 输入类型配置寄存器: 0: Schmitt input buffer 1: CMOS input buffer
1:0	RSV	-	-	保留

4.6.23 端口输入类型配置寄存器 1/ PAD_CS1 (偏移: 06CH)

比特	名称	属性	复位值	描述
6:5	RSV	-	-	保留
4	PE4_CS	R/W	1	端口 PE4 输入类型配置寄存器: 0: Schmitt input buffer 1: CMOS input buffer
3	RSV	-	-	保留
2	PE2_CS	R/W	1	端口 PE2 输入类型配置寄存器: 0: Schmitt input buffer 1: CMOS input buffer
1:0	RSV	-	-	保留

4.6.24 端口输入配置寄存器 0/ PAD_IE0 (偏移: 070H)

比特	名称	属性	复位值	描述
31	RSV	-	-	保留

比特	名称	属性	复位值	描述
30	PD6_IE	R/W	0	端口 PD6 输入配置寄存器: 0: 输入禁止 1: 输入使能
29	PD5_IE	R/W	0	端口 PD5 输入配置寄存器: 0: 输入禁止 1: 输入使能
28	PD4_IE	R/W	0	端口 PD4 输入配置寄存器: 0: 输入禁止 1: 输入使能
27	RF_MOSI_IE	R/W	0	端口 RF_MOSI 输入配置寄存器: 0: 输入禁止 1: 输入使能
26	RF_MISO_IE	R/W	0	端口 RF_MISO 输入配置寄存器: 0: 输入禁止 1: 输入使能
25	RF_CLK_IE	R/W	0	端口 RF_CLK 输入配置寄存器: 0: 输入禁止 1: 输入使能
24	PD0_IE	R/W	0	端口 PD0 输入配置寄存器: 0: 输入禁止 1: 输入使能
23	RSV	-	-	保留
22	PC6_IE	R/W	1	端口 PC6 输入配置寄存器: 0: 输入禁止 1: 输入使能
21	PC5_IE	R/W	1	端口 PC5 输入配置寄存器: 0: 输入禁止 1: 输入使能
20	PC4_IE	R/W	0	端口 PC4 输入配置寄存器: 0: 输入禁止 1: 输入使能
19	RSV	-	-	保留
18	PC2_IE	R/W	0	端口 PC2 输入配置寄存器: 0: 输入禁止 1: 输入使能

比特	名称	属性	复位值	描述
17	PC1_IE	R/W	0	端口 PC1 输入配置寄存器: 0: 输入禁止 1: 输入使能
16	RSV	-	-	保留
15	PB7_IE	R/W	0	端口 PB7 输入配置寄存器: 0: 输入禁止 1: 输入使能
14:13	RSV	-	-	保留
12	PB4_IE	R/W	0	端口 PB4 输入配置寄存器: 0: 输入禁止 1: 输入使能
11	PB3_IE	R/W	0	端口 PB3 输入配置寄存器: 0: 输入禁止 1: 输入使能
10	PB2_IE	R/W	0	端口 PB2 输入配置寄存器: 0: 输入禁止 1: 输入使能
9	PB1_IE	R/W	0	端口 PB1 输入配置寄存器: 0: 输入禁止 1: 输入使能
8:3	RSV	-	-	保留
2	PA2_IE	R/W	1	端口 PA2 输入配置寄存器: 0: 输入禁止 1: 输入使能
1:0	RSV	-	-	保留

4.6.25 端口输入配置寄存器 1/ PAD_IE1 (偏移: 074H)

比特	名称	属性	复位值	描述
6:5	RSV	-	-	保留
4	PE4_IE	R/W	0	端口 PE4 输入配置寄存器: 0: 输入禁止 1: 输入使能
3	RSV	-	-	保留

比特	名称	属性	复位值	描述
2	PE2_IE	R/W	0	端口 PE2 输入配置寄存器: 0: 输入禁止 1: 输入使能
1:0	RSV	-	-	保留

4.6.26 端口速度配置寄存器 0/ PAD_SR0 (偏移: 078H)

比特	名称	属性	复位值	描述
31	RSV	-	-	保留
30	PD6_SR	R/W	1	端口 PD6 速度配置寄存器: 0: 高速 1: 低速
29	PD5_SR	R/W	1	端口 PD5 速度配置寄存器: 0: 高速 1: 低速
28	PD4_SR	R/W	1	端口 PD4 速度配置寄存器: 0: 高速 1: 低速
27	RF_MOSI_SR	R/W	1	端口 RF_MOSI 速度配置寄存器: 0: 高速 1: 低速
26	RF_MISO_SR	R/W	1	端口 RF_MISO 速度配置寄存器: 0: 高速 1: 低速
25	RF_CLK_SR	R/W	1	端口 RF_CLK 速度配置寄存器: 0: 高速 1: 低速
24	PD0_SR	R/W	1	端口 PD0 速度配置寄存器: 0: 高速 1: 低速
23	RSV	-	-	保留
22	PC6_SR	R/W	1	端口 PC6 速度配置寄存器: 0: 高速 1: 低速

比特	名称	属性	复位值	描述
21	PC5_SR	R/W	1	端口 PC5 速度配置寄存器: 0: 高速 1: 低速
20	PC4_SR	R/W	1	端口 PC4 速度配置寄存器: 0: 高速 1: 低速
19	RSV	-	-	保留
18	PC2_SR	R/W	1	端口 PC2 速度配置寄存器: 0: 高速 1: 低速
17	PC1_SR	R/W	1	端口 PC1 速度配置寄存器: 0: 高速 1: 低速
16	RSV	-	-	保留
15	PB7_SR	R/W	1	端口 PB7 速度配置寄存器: 0: 高速 1: 低速
14:13	RSV	-	-	保留
12	PB4_SR	R/W	1	端口 PB4 速度配置寄存器: 0: 高速 1: 低速
11	PB3_SR	R/W	1	端口 PB3 速度配置寄存器: 0: 高速 1: 低速
10	PB2_SR	R/W	1	端口 PB2 速度配置寄存器: 0: 高速 1: 低速
9	PB1_SR	R/W	1	端口 PB1 速度配置寄存器: 0: 高速 1: 低速
8:3	RSV	-	-	保留
2	PA2_SR	R/W	1	端口 PA2 速度配置寄存器: 0: 高速 1: 低速
1:0	RSV	-	-	保留

4.6.27 端口速度配置寄存器 1/ PAD_SR1 (偏移: 07CH)

比特	名称	属性	复位值	描述
6:5	RSV	-	-	保留
4	PE4_SR	R/W	1	端口 PE4 速度配置寄存器: 0: 高速 1: 低速
3	RSV	-	-	保留
2	PE2_SR	R/W	1	端口 PE2 速度配置寄存器: 0: 高速 1: 低速
1:0	RSV	-	-	保留

4.6.28 IO 控制保护寄存器/ IOCTRL_PROTECT (偏移: 080H)

比特	名称	属性	复位值	描述
31:0	IOCTRL_PROTECT	R/W	0	IO 寄存器 PA_SEL/PB_SEL/PC_SEL/PD_SEL/PE_SEL/PAD_ADS/PAD_DR0/PAD_DR1/PAD_PU0/PAD_PU1/PAD_PD0/PAD_PD1/PAD_OD0/PAD_OD1/PAD_CS0/PAD_CS1/PAD_IE0/PAD_IE1/PAD_SR0/PAD_SR1 保护的寄存器。给此寄存器写 0xA5A5_5A5A, 启动这些 IO 寄存器的写使能。配置完 IO 寄存器后, 它们的写使能不会自动关闭。可以给此寄存器写其它值, 来关闭 IO 寄存器的写使能 读取此寄存器返回 IO 寄存器的写使能状态: 0: 写未使能 1: 写已经使能

4.6.29 LVD 配置寄存器/ LVD_CFG (偏移: 084H)

比特	名称	属性	复位值	描述
31:25	RSV	-	-	保留
24	LVD_LVEN	R/W	1	LVD 滤波使能位

比特	名称	属性	复位值	描述
23:16	LVD_FILTER	R/W	8'h20	LVD 滤波配置位: 0: 对 LVD 滤除 1 个 32K 系统低速时钟的毛刺 1: 对 LVD 滤除 2 个 32K 系统低速时钟的毛刺 65535: 对 LVD 滤除 65536 个 32K 系统低速时钟毛刺
15:10	RSV	-	-	保留
9	LVD_INTR_EN	R/W	0	LVD 中断使能控制位: 0: 不使能 LVD 中断 1: 使能 LVD 中断
8	LVD_RESET_EN	R/W	0	LVD 复位使能控制位: 0: 不使能 LVD 复位 1: 使能 LVD 复位
7:4	LVDS	R/W	0	LVD 检测点电压设置: 0000: 1.65V 0001: 1.75V 0010: 1.85V 0011: 1.95V 0100: 2.05V 0101: 2.15V 0110: 2.25V 0111: 2.35V 1000: 2.45V 1001: 2.55V 1010: 2.65V 1011: 2.75V 1100: 2.85V 1101: 2.95V 1110: 3.05V 1111: 3.15V
3	RSV	-	-	保留
2	LVD_FLAG	R	0	LVD 当前状态: 0: 当前 LVD 未检测到电压过低的事件 1: 当前 LVD 检测到电压过低的事件

比特	名称	属性	复位值	描述
1	LVD_INTR	R/W	0	LVD 中断状态标志: 0: 未发生过 LVD 中断 1: 发生过 LVD 中断 写 1 清 0
0	LVD_EN	R/W	0	LVD 模块使能寄存器: 0: 禁止 1: 使能

4.6.30 外部复位端口选择寄存器/ EXTRST_SEL (偏移: 090H)

比特	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	RESETN_SEL	R/W	0	外部复位端口选择寄存器 0: 外部复位信号有效。 1: 外部复位信号无效。该引脚可以作为 GPIO 使用 只有该寄存器 RESETN_SEL [31:16]的高 16 位写 0xA5A5 时才能写这个 bit.

4.6.31 停止模式选择寄存器/ STOPMODE_SEL (偏移: 094H)

比特	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	STOPMODE_SEL	R/W	0	停止模式选择寄存器: 0: Stop mode 停止模式无效 1: Stop mode 停止模式有效 只有该寄存器 STOPMODE_SEL[31:16]的高 16 位写 0xA5A5 时才能写这个 bit

4.6.32 REMAP 寄存器/ REMAP_ADDR (偏移: 098H)

比特	名称	属性	复位值	描述
31:3	RSV	-	-	保留
2	REMAP	R	0	eFlash 地址 remap 标志位: 0: eFlash 地址没有重映射, Bootloader 启动 1: eFlash 地址进行重映射, Main 区启动
1	REMAP_IM	W	1	立即进行 REMAP 操作, 但是系统不发生复位 0: 立即进行 eFlash 地址重映射 1: eFlash 地址不进行重映射
0	SOFT_RESETN	W	1	软复位, 当此位写 0 时, 会产生一次软件复位, 复位 CPU 及 AHB/APB 总线上的所有 IP。 并且, eFlash 地址重新映射 (remap 为 1) 0: 系统进行软复位 1: 系统不进行软复位

注: 寄存器 REMAP 位复位信号为 SYSTEM_RESETN, 不会随 PRESETn 复位而复位掉。

4.6.33 中断向量地址重映射寄存器/ VECTOR_OFFSET (偏移: 09CH)

比特	名称	属性	复位值	描述
31:10	VECTROFFSET	R/W	0	中断向量重映射功能使能后, 中断向量的地址是本寄存器中的值
9:1	RSV	-	-	保留
0	VECTROFFSET_EN	R/W	0	中断向量重映射功能使能: 0: 不使能中断向量重映射功能 1: 使能中断向量重映射功能

4.6.34 随机数控制寄存器/ HRNG_CR (偏移: 0A0H)

比特	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	HRNG_EN	R/W	0	1: 随机数使能 0: 随机数禁止

4.6.35 随机数种子寄存器/ HRNG_SEED (偏移: 0A4H)

比特	名称	属性	复位值	描述
31:0	HRNG_SEED	R/W	0	随机数种子寄存器

4.6.36 随机数数据寄存器 / HRNG_DATA (偏移: 0A8H)

比特	名称	属性	复位值	描述
31:0	HRNG_DATA	R	32'hFF00FF	随机数寄存器。读取此寄存器，读出随机数值。

4.6.37 LVR 控制寄存器/ LVR_CFG (偏移: 0B0H)

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:4	LVR_S	R/W	0	LVR 检测点电压设置: 0000: 1.65V 0001: 1.75V 0010: 1.85V 0011: 1.95V 0100: 2.05V 0101: 2.15V 0110: 2.25V 0111: 2.35V 1000: 2.45V 1001: 2.55V 1010: 2.65V 1011: 2.75V 1100: 2.85V 1101: 2.95V 1110: 3.05V 1111: 3.15V
3:1	RSV	-	-	保留
0	LVR_EN	R/W	1	LVR 使能信号

4.6.38 VREF 控制寄存器/VREF_CFG (偏移: 0B4H)

比特	名称	属性	复位值	描述
31:22	RSV	-	-	保留
21	VREF_LVEN	R/W	0	VREF 滤波使能: 0: 禁止 VREF 滤波 1: 使能 VREF 滤波
20:19	VREF_LVSET	R/W	0	VREF 滤波时间设置: 00: VREF 过温电路输出滤波 2 个 32K 时钟 01: VREF 过温电路输出滤波 4 个 32K 时钟 10: VREF 过温电路输出滤波 8 个 32K 时钟 11: VREF 过温电路输出滤波 16 个 32K 时钟
18	VREF_DIV_EN	R/W	0	VREF 时钟使能位: 0: 不使能 VREF 时钟 1: 使能 VREF 时钟
17:13	VREF_DIV_VAL	R/W	0	VREF 时钟分频值, VREF 工作需要 3M 左右的时钟, 从系统时钟分频得到
12:11	VREF_VREFOUT_SEL	R/W	0	选择 VREFOUT 的输出电压: 00: VREFOUT = 1.25V (VDDH > 1.8V) 01: VREFOUT = 2V (VDDH > 2.3V) 10: VREFOUT = 2.5V (VDDH > 2.8V)
10	VREF_EN_TEST	R/W	0	控制内部 VREF 的测试模式: 0: 禁用 VREF 测试模式 1: 使能 VREF 测试模式 输出 buffer 设置为单位缓冲区 VREFOUT = VREF
9	VREF_EN_TS	R/W	0	控制温度传感器: 0: 关闭温度传感器 1: 使能温度传感器
8	VREF_EN_OP_CHOP	R/W	0	控制内部 OPA 的斩波: 0: 禁止 OPA 的斩波功能 1: 使能 OPA 的斩波功能
7	VREF_EN_LOAD	R/W	0	控制 VREFOUT 的虚拟负载: 0: 禁用虚拟加载 1: 使能虚拟加载

比特	名称	属性	复位值	描述
6	VREF_EN_DEM	R/W	0	控制动态元素匹配 (DEM): 0: 禁用 DEM 1: 启用 DEM
5	VREF_EN_BJT_CHOP	R/W	0	控制 BJT 的斩波功能: 0: 禁止 BJT 斩波 1: 使能 BJT 斩波
4	VREF_EN	R/W	0	控制 VREFOUT 输出: 0: 禁止 VREFOUT 输出 1: 使能 VREFOUT 输出
3:1	VREF_CHOP_OP_SEL	R/W	0	设置内部 OPA 斩波时钟的频率: 000: 8 分频 001: 16 分频 010: 32 分频 011: 64 分频 10x: 128 分频 11x: 256 分频
0	VREF_CHOP_BJT_SEL	R/W	0	设置 BJT 的频率 建议保持为默认值 0

4.6.39 内部基准状态寄存器/VREF_STATUS (偏移: 0CCH)

比特	名称	属性	复位值	描述
31:3	RSV	-	-	保留
2	VREF_INTEN	R/W	0	VREF 过温保护中断使能: 0: 当前温度低于设定温度 1: 当前温度超过设定温度
1	VREF_CST	R	0	VREF 实时状态寄存器: 0: 当前 VREF 没有过温保护事件发生 1: 当前 VREF 有过温保护事件发生
0	VREF_INTR	R/W	0	VREF 过温保护中断状态: 0: VREF 未发生过温保护中断 1: VREF 发生过温保护中断 写 1 清 0

5 RF (Sub-1GHz 收发器)

5.1 概述

RF (Sub-1GHz 收发器) 工作于 200 ~ 960MHz 范围内的低功耗、高性能、单片集成的 (G)FSK/OOK 收发器。内部集成完整的射频接收机、射频发射机、频率综合器及调制解调器，只需配备简单、低成本的外围器件即可获得良好的收发性能。

RF 支持灵活可设的数据包格式，支持自动应答与自动重发功能，支持跳频操作，同时集成了 FEC 功能，可通过 SPI 对 RF 进行控制，并访问内部收发各 128 bytes 的 TX/RX FIFO。

5.2 主要特性

- 频率范围：200 ~ 960MHz
- 调制方式：(G)FSK / OOK
- 数据率：0.1 ~ 300 kbps@(G)FSK、0.5~40kbps@OOK
- 支持 NRZ、曼彻斯特、数据白化
- 自动应答/自动重传
- 支持 RSSI, 0.5dB 检测精度
- 可配置包处理机及 128-Byte TX/RX FIFO
- AGC / AFC
- 支持 FEC
- 支持外部复位、关断

5.3 寄存器描述

5.3.1 Reg00 Address: 0x00 Default: 0x1D

比特	名称	属性	复位值	描述
7	RSV	R/W	1'b0	-
6:0	rf_f0[30:24]	R/W	7'h1D	信道频率 f0[30:24]。 信道频率 rf_freq 设置公式： rf_freq=rf_f0+ch_num*f_step rf_f0 由{Reg00,Reg01,Reg02,Reg03}配置，单位为 MHz，低 20 位为小数； ch_num 由 Reg04 配置； 步进频率 f_step 由{Reg05,Reg06,Reg07}配置，单位为 MHz，低 20 位为小数

5.3.2 Reg01 Address: 0x01 Default: 0x78

比特	名称	属性	复位值	描述
7:0	rf_f0[23:16]	R/W	8'h78	信道频率 f0[23:16]

5.3.3 Reg02 Address: 0x02 Default: 0x00

比特	名称	属性	复位值	描述
7:0	rf_f0[15:8]	R/W	8'h00	信道频率 f0[15:8]

5.3.4 Reg03 Address: 0x03 Default: 0x00

比特	名称	属性	复位值	描述
7:0	rf_f0[7:0]	R/W	8'h00	信道频率 f0[7:0]

5.3.5 Reg04 Address: 0x04 Default: 0x00

比特	名称	属性	复位值	描述
7:0	ch_num	R/W	8'h00	信道号 ch_num, 参考 Reg00 描述

5.3.6 Reg05 Address: 0x05 Default: 0x03

比特	名称	属性	复位值	描述
7:0	f_step[23:16]	R/W	8'h03	信道步进频率 f_step[23:16], 单位为 MHz, 低 20 位为小数, 参考 Reg00 描述

5.3.7 Reg06 Address: 0x06 Default: 0x33

比特	名称	属性	复位值	描述
7:0	f_step[15:8]	R/W	8'h33	信道步进频率 f_step[15:8], 参考 Reg00 描述

5.3.8 Reg07 Address: 0x07 Default: 0x33

比特	名称	属性	复位值	描述
7:0	f_step[7:0]	R/W	8'h33	信道步进频率 f_step[7:0], 参考 Reg00 描述

5.3.9 Reg08 Address: 0x08 Default: 0x10

比特	名称	属性	复位值	描述
7	RSV	R/W	1'b0	-
6:0	ref_freq[30:24]	R/W	7'h10	参考频率 (晶振频率) 设置, ref_freq 由 {Reg08[6:0],Reg09,Reg0A,Reg0B}配置, 单位为 MHz, 低 24bits 为小数部分。

5.3.10 Reg09 Address: 0x09 Default: 0x00

比特	名称	属性	复位值	描述
7:0	ref_freq[23:16]	R/W	8'h00	参考 Reg08 描述

5.3.11 Reg0A Address: 0x0A Default: 0x00

比特	名称	属性	复位值	描述
7:0	ref_freq[15:8]	R/W	8'h00	参考 Reg08 描述

5.3.12 Reg0B Address: 0x0B Default: 0x00

比特	名称	属性	复位值	描述
7:0	ref_freq[7:0]	R/W	8'h00	参考 Reg08 描述

5.3.13 Reg0C Address: 0x0C Default: 0x50

比特	名称	属性	复位值	描述
7:0	tx_preamble_len	R/W	8'h50	发射 Preamble 长度, 单位字节

5.3.14 Reg0D Address: 0x0D Default: 0x30

比特	名称	属性	复位值	描述
7	payload_bit_order	R/W	1'b0	Payload bit 顺序: 0: LSB 低位在前 1: MSB 高位在前
6	manchester_inv	R/W	1'b0	曼彻斯特编码: 0: 上升沿为编码 1, 下降沿为编码 0 1: 下降沿为编码 1, 上升沿为编码 0

比特	名称	属性	复位值	描述
5	syncword_en	R/W	1'b1	Syncword 同步字使能: 0: 禁止 1: 使能
4	preamble_en	R/W	1'b1	设置 Preamble 前导码使能: 0: 禁止 1: 使能
3:2	pkt_enc_type	R/W	2'b00	数据包编码: 00: NRZ 01: 曼彻斯特编码 10: 无效 11: 交织编码
1:0	fec_type	R/W	2'b00	数据包 FEC 类型: 00: NO FEC 01: 1/3 FEC 10: 2/3 FEC 11: 1/2 FEC

5.3.15 Reg0E Address: 0x0E Default: 0x32

比特	名称	属性	复位值	描述
7	length_byte_swap	R/W	1'b0	包长度控制域为两个字节时, 高低字节顺序: 0: 低字节在前 1: 高字节在前
6	length_sel	R/W	1'b0	包长度控制域字节配置: 0: 1 字节 1: 2 字节
5	crc_en	R/W	1'b1	CRC 使能: 0: 禁止 1: 使能
4	scramble_en	R/W	1'b1	数据白化(扰码)使能: 0: 禁止 1: 使能
3	fifo_share_en	R/W	1'b0	FIFO 共享设置: 0: RX 和 TX 各 128 字节 FIFO 1: RX 和 TX 共用 256 字节 FIFO

比特	名称	属性	复位值	描述
2	direct_mode	R/W	1'b0	直通模式使能: 0: 禁止 1: 使能
1:0	packet_mode	R/W	2'b10	数据包控制模式: 00: 模式 0, 循环发射 TX FIFO 中的数据, 需要 MCU 写命令退出接收或发射状态 01: 模式 1, 由寄存器控制数据包长度 10: 模式 2, TX FIFO 中的第 1 个或 2 个字节作为包长度 11: 模式 3, 含有 length, address, seqnum 等多种数据域的模式

5.3.16 Reg10 Address: 0x10 Default: 0x40

比特	名称	属性	复位值	描述
7:4	retx_times	R/W	4'h4	Auto ACK 使能时, 发射端未收到 ACK 包时自动重发的次数
3	rss_val_sel	R/W	1'b0	读取 RSSI 时, 选择读取的数据内容: 0: 经过计算的 RSSI 值 1: 原始信号幅度值
2	preamble_int_en	R/W	1'b0	接收 Preamble (前导码) 匹配中断使能: 0: 禁止 1: 使能
1	syncword_int_en	R/W	1'b0	接收 Syncword (同步字) 匹配中断使能: 0: 禁止 1: 使能
0	RSV	R/W	1'b0	-

5.3.17 Reg11 Address: 0x11 Default: 0x00

比特	名称	属性	复位值	描述
7	RSV	R/W	1'b0	-
6	int_flag_clr	R/W	1'b0	写“1”清除中断标志
5	miso_tri_opt	R/W	1'b0	SPI 在非工作状态时, spi_miso 状态: 0: 高阻 1: 输出

比特	名称	属性	复位值	描述
4:3	RSV	R/W	2'b00	-
2:0	brclk_sel	R/W	3'b000	GPIO brclk 输出时钟选择: 000: rx_clk 接收 bit rate 时钟 001: xtal_clk 晶振时钟 010: xtal_clk/8 011: xtal_clk/16 100: 频综时钟 101: tx_clk 发射 bit rate 时钟 111: ADC_clk

5.3.18 Reg1D Address: 0x1D Default: 0x80

比特	名称	属性	复位值	描述
7:6	vdddig	R/W	2'b10	数字 LDO 输出电压设置: 00: 0.8V 01: 1.0V 10: 1.2V 11: 1.4V
5:0	RSV	R/W	6'h00	-

5.3.19 Reg20 Address: 0x20 Default: 0x50

比特	名称	属性	复位值	描述
7:6	frequency_band	R/W	2'b01	工作频段设置: 00: 800MHz ~ 1GHz 频段 01: 400MHz ~ 500MHz 频段 10: 267MHz ~ 350MHz 频段 11: 200MHz ~ 250MHz 频段
5:0	RSV	R/W	6'h10	-

5.3.20 Reg21 Address: 0x21 Default: 0x60

比特	名称	属性	复位值	描述
7	RSV	R/W	1'b0	-
6	pa_ramp_en	R/W	1'b1	PA RAMP 使能: 0: 禁止 1: 使能
5:0	pa_bias	R/W	6'h20	PA_bias 控制: 000000: 最小 111111: 最大

5.3.21 Reg27 Address: 0x27 Default: 0x60

比特	名称	属性	复位值	描述
7	RSV	R/W	1'b0	-
6	pa_drvh	R/W	1'b1	PA 前级驱动设置: 0: Low 1: High
5:0	pa_gain	R/W	6'h20	PA 增益控制

5.3.22 Reg28 Address: 0x28 Default: 0x19

比特	名称	属性	复位值	描述
7	rss_i_Reg_clr	R/W	1'b0	写“1”清除被锁定的 RSSI
6	rss_i_sel	R/W	1'b0	收到同步字后锁定 RSSI 值: 0: 禁止 1: 使能
5	rx_rssi_thr_en	R/W	1'b0	接收时判断 RSSI, 如果收到前导码的 RSSI 低于设置的 RSSI 门限值 (Reg2C), 会复位解调器重新接收 0: 禁止 1: 使能

比特	名称	属性	复位值	描述
4	agc_en	R/W	1'b1	AGC 使能: 0: 禁止 1: 使能
3:0	RSV	R/W	4'h9	-

5.3.23 Reg2B Address: 0x2B Default: 0xFF

比特	名称	属性	复位值	描述
7:6	RSV	R/W	2'b11	-
5:4	rxfe_gn	R/W	2'b11	rxfe 增益控制
3:2	filter_gn	R/W	2'b11	Filter 增益控制
1:0	pga_gn	R/W	2'b11	PGA 增益控制

5.3.24 Reg2C Address: 0x2C Default: 0xFF

比特	名称	属性	复位值	描述
7:0	rx_rssi_thr	R/W	8'hFF	接收 RSSI 门限值, 参考 Reg28[5]

5.3.25 Reg2D Address: 0x2D Default: 0x10

比特	名称	属性	复位值	描述
7	cw_mode	R/W	1'b0	发射载波模式: 0: 禁止 1: 使能
6:4	RSV	R/W	3'b001	-
3	ook_en	R/W	1'b0	调制解调模式选择: 0: FSK 1: OOK
2:0	RSV	R/W	3'b000	-

5.3.26 Reg30 Address: 0x30 Default: 0x80

比特	名称	属性	复位值	描述
7	sel_gau_o	R/W	1'b1	发射高斯滤波器使能: 0: 禁止 1: 使能
6:4	RSV	R/W	3'b000	-
3:0	dev_set[11:8]	R/W	4'h0	发射调制频偏 dev_set 的高 4 位

5.3.27 Reg31 Address: 0x31 Default: 0x66

比特	名称	属性	复位值	描述
7:0	dev_set[7:0]	R/W	8'h66	发射调制频偏 dev_set[11:0]={Reg30[3:0],Reg31[7:0]} 例: 频偏为正负 25kHz 即 0.025MHz, 则 dev_set = 0.025*4096

5.3.28 Reg37 Address: 0x37 Default: 0x06

比特	名称	属性	复位值	描述
7:0	csrst_len	R/W	8'h06	在补偿功能使能时, 找到信号的频谱后连续收到 Reg37*4 个前导后, 则认为找到有效的 Preamble

5.3.29 Reg38 Address: 0x38 Default: 0x91

比特	名称	属性	复位值	描述
7:5	RSV	R/W	3'b100	-
4	demod_comp_en	R/W	1'b1	解调频偏补偿: 0: 禁止 1: 使能
3:0	RSV	R/W	4'h1	-

5.3.30 Reg4D Address: 0x4D Default: 0x10

比特	名称	属性	复位值	描述
7	nirq_dir	R/W	1'b0	nIRQ_flag IO 口方向: 0: Output 1: Input
6	gpio0_dir	R/W	1'b0	gpio0 方向: 0: Output 1: Input
5	RSV	R/W	1'b0	-
4	gpio2_dir	R/W	1'b1	gpio2 方向, 默认复位输入功能: 0: Output 1: Input
3:0	nirq_sel	R/W	4'h0	nIRQ 引脚输出功能选择: 0x00: pkt_flag (包括数据包完成中断 pkt_int, 接收前导码中断 preamble_int, 接收 同步字中断 syncword_int) 0x01: pkt_int 0x02: preamble_int 0x03: syncword_int 0x04: fifo_flag 0x05: brclk (参考 Reg11[2:0]描述) 0x06: rxdata 0x07: txdata 0x08: tr_switch 0x09: tr_switch 反向 0x0A: rxdata 0x0B: txdata 0x0D: 高电平 Others: 低电平

5.3.31 Reg4E Address: 0x4E Default: 0x45

比特	名称	属性	复位值	描述
7:4	gpio0_sel	R/W	4'h4	gpio0 输出功能选择, 参考 Reg4D[3:0]描述
3:0	RSV	R/W	4'h5	-

5.3.32 Reg4F Address: 0x4F Default: 0x67

比特	名称	属性	复位值	描述
7:4	gpio2_sel	R/W	4'h6	gpio2 输出功能选择, 参考 Reg4D[3:0]描述
3:0	RSV	R/W	4'h7	-

5.3.33 Reg50 Address: 0x50 Default: 0x00

比特	名称	属性	复位值	描述
7:0	rx_fifo_wr_ptr	R/W	8'h00	RX FIFO 写指针, 写 0x80 清除指针

5.3.34 Reg51 Address: 0x51 Default: 0x00

比特	名称	属性	复位值	描述
7:0	rx_fifo_rd_ptr	R/W	8'h00	RX FIFO 读指针, 写 0x80 清除指针

5.3.35 Reg52 Address: 0x52 Default: 0x00

比特	名称	属性	复位值	描述
7:0	rx_fifo_data	R/W	8'h00	RX FIFO

5.3.36 Reg53 Address: 0x53 Default: 0x00

比特	名称	属性	复位值	描述
7:0	tx_fifo_wr_ptr	R/W	8'h00	TX FIFO 写指针, 写 0x80 清除指针

5.3.37 Reg54 Address: 0x54 Default: 0x00

比特	名称	属性	复位值	描述
7:0	tx_fifo_rd_ptr	R/W	8'h00	TX FIFO 读指针, 写 0x80 清除指针

5.3.38 Reg55 Address: 0x55 Default: 0x00

比特	名称	属性	复位值	描述
7:0	tx_fifo_data	R/W	8'h00	TX FIFO

5.3.39 Reg56 Address: 0x56 Default: 0xB7

比特	名称	属性	复位值	描述
7:0	delta_rssi	R/W	8'hB7	计算 RSSI 时的预偏移量

5.3.40 Reg57 Address: 0x57 Default: 0x00

比特	名称	属性	复位值	描述
7:2	RSV	R/W	6'h00	-
1	crc_err_st	R/W	1'b0	接收 CRC 错误时, 数据包的处理方法: 0: 正常完成该包的接收, 退出接收状态发出数据包中断和错误 CRC 标志 1: 丢弃该包数据并重新进入接收状态
0	RSV	R/W	1'b0	-

5.3.41 Reg58 Address: 0x58 Default: 0x00

比特	名称	属性	复位值	描述
7:5	RSV	R/W	3'b000	-
4:3	direct_tx_data_sel	R/W	2'b00	Direct mode 时, TX 数据源选择: 00: nIRQ 引脚 01: gpio0
2:0	RSV	R/W	3'b000	-

5.3.42 Reg5D Address: 0x5D Default: 0x00

比特	名称	属性	复位值	描述
7	nrst_sel	R/W	1'b0	nRST 引脚功能设置: 0: nRST 1: GPIO2
6	pull_up_en	R/W	1'b0	nRST 引脚 Pull-UP 使能: 0: 使能 1: 禁止
5:0	reseved	R/W	5'h00	-

5.3.43 Reg60 Address: 0x60 Default: 0x80

比特	名称	属性	复位值	描述
7:0	command	R/W	8'h80	0x80: IDLE command 0x40: TX Command 0x20: RX command 0x05: StandBy command 0x06: Reset command, 需要先使能 RegA3[7] 0x10: FS command 打开频综

5.3.44 Reg61 Address: 0x61 Default: 0x00

比特	名称	属性	复位值	描述
7	syncword_rec	R	1'b0	接收到正确的 Syncword
6	preamble_rec	R	1'b0	接收到有效的 Preamble
5	crc_error	R	1'b0	高电平时表示 CRC 校验错误
4	pkt_flag	R	1'b0	pkt_flag
3	fifo_flag	R	1'b0	fifo_flag
2:0	RSV	R	3'b000	-

5.3.45 Reg64 Address: 0x64 Default: 0x00

比特	名称	属性	复位值	描述
7:0	rssi	R	8'h00	读出的值为实际 RSSI 的绝对值

5.3.46 Reg67 Address: 0x67 Default: 0x80

比特	名称	属性	复位值	描述
7	st_idle	R	1'b1	IDLE 状态
6	st_tx	R	1'b0	发射状态
5	st_rx	R	1'b0	接收状态
4	st_fson	R	1'b0	频综开启状态
3:0	reseved	R	4'h0	-

5.3.47 Reg68 Address: 0x68 Default: 0x00

比特	名称	属性	复位值	描述
7:2	reseved	R	6'h0	-
1	m3_seqnum_ok	R	1'b0	数据包模式 3 时, 接收 seqnum 匹配正确
0	m3_addr_comp_ok	R	1'b0	数据包模式 3 时, 接收 address 匹配正确

5.3.48 Reg69 Address: 0x69 Default: 0x00

比特	名称	属性	复位值	描述
7:0	m3_rx_fcs2_value	R	8'h00	数据包模式 3 时, 接收 FCS2 数据

5.3.49 Reg6A Address: 0x6A Default: 0x00

比特	名称	属性	复位值	描述
7:0	m3_rx_payload_len[7:0]	R	8'h00	数据包模式 3 时, 接收 length 数据

5.3.50 Reg6B Address: 0x6B Default: 0x00

比特	名称	属性	复位值	描述
7:0	m3_rx_payload_len[15:8]	R	8'h00	数据包模式 3 时, 接收 length 数据

5.3.51 Reg6C Address: 0x6C Default: 0x00

比特	名称	属性	复位值	描述
7:0	m3_rx_addr[7:0]	R	8'h00	数据包模式 3 时, 接收 address

5.3.52 Reg6D Address: 0x6D Default: 0x00

比特	名称	属性	复位值	描述
7:0	m3_rx_addr[15:8]	R	8'h00	数据包模式 3 时, 接收 address

5.3.53 Reg6E Address: 0x6E Default: 0x00

比特	名称	属性	复位值	描述
7:0	m3_rx_addr[23:16]	R	8'h00	数据包模式 3 时, 接收 address

5.3.54 Reg6F Address: 0x6F Default: 0x00

比特	名称	属性	复位值	描述
7:0	m3_rx_addr[31:24]	R	8'h00	数据包模式 3 时, 接收地址域数据

5.3.55 Reg70 Address: 0x70 Default: 0xFF

比特	名称	属性	复位值	描述
7:0	scramble_init_val[7:0]	R/W	8'hFF	scramble 初始化值

5.3.56 Reg71 Address: 0x71 Default: 0x09

比特	名称	属性	复位值	描述
7	RSV	R/W	1'b0	-
6	srcamble_len	R/W	1'b0	伪随机数长度选择: 0: PN9 1: PN7
5	scramble_msb	R/W	1'b0	选择首先输出的移位寄存器: 0: LSB 1: MSB
4	scramble_type	R/W	1'b0	伪随机数产生器类型: 0: 抽出模式 1: 插入模式
3:1	scramble_poly	R/W	3'b100	反馈抽头选择
0	scramble_init_val[8]	R/W	1'b1	scramble 初始化值最高位

5.3.57 Reg72 Address: 0x72 Default: 0x01

比特	名称	属性	复位值	描述
7:5	RSV	R/W	3'b000	-
4:0	sync_thres	R/W	5'h01	同步字检测允许的误差位数

5.3.58 Reg73 Address: 0x73 Default: 0x55

比特	名称	属性	复位值	描述
7:0	preamble_val	R/W	8'h55	发射前导码时使用的的数据

5.3.59 Reg74 Address: 0x74 Default: 0x03

比特	名称	属性	复位值	描述
7	auto_ack	R/W	1'b0	auto_ack 使能: 0: 禁止 1: 使能
6	ack_payload_en	R/W	1'b0	发射 ACK 时带 payload 数据返回: 0: 禁止 1: 使能
5	cont_rx_en	R/W	1'b0	控制接收状态机: 0: 接收完成数据包后退出接收状态进入 IDLE 状态 1: 接收完成数据包后重新再次进入接收状态
4	sync_man_en	R/W	1'b0	同步字曼彻斯特编码使能: 0: 禁止 1: 使能
3	sync_bit_order	R/W	1'b0	同步字 bit 顺序: 0: 字节低位 1: 字节高位 MSB
2:0	sync_len	R/W	3'b011	同步字长度: 000: {sync_id_1} 001: {sync_id_1, sync_id_2} 010: {sync_id_1, sync_id_2, sync_id_3} 011: {sync_id_1, sync_id_2, sync_id_3, sync_id_4} 100: {sync_id_1, sync_id_2, sync_id_3, sync_id_4, sync_id_5} 101: {sync_id_1, sync_id_2, sync_id_3, sync_id_4,

比特	名称	属性	复位值	描述
				sync_id_5, sync_id_6} 110: {sync_id_1, sync_id_2, sync_id_3, sync_id_4, sync_id_5, sync_id_6, sync_id_7} 111: {sync_id_1, sync_id_2, sync_id_3, sync_id_4, sync_id_5, sync_id_6, sync_id_7, sync_id_8}

5.3.60 Reg75 Address: 0x75 Default: 0xA7

比特	名称	属性	复位值	描述
7:0	sync_id_1	R/W	8'hA7	同步字字节 1

5.3.61 Reg76 Address: 0x76 Default: 0xA7

比特	名称	属性	复位值	描述
7:0	sync_id_2	R/W	8'hA7	同步字字节 2

5.3.62 Reg77 Address: 0x77 Default: 0x98

比特	名称	属性	复位值	描述
7:0	sync_id_3	R/W	8'h98	同步字字节 3

5.3.63 Reg78 Address: 0x78 Default: 0xF3

比特	名称	属性	复位值	描述
7:0	sync_id_4	R/W	8'hF3	同步字字节 4

5.3.64 Reg79 Address: 0x79 Default: 0x98

比特	名称	属性	复位值	描述
7:0	sync_id_5	R/W	8'h98	同步字字节 5

5.3.65 Reg7A Address: 0x7A Default: 0xF3

比特	名称	属性	复位值	描述
7:0	sync_id_6	R/W	8'hF3	同步字字节 6

5.3.66 Reg7B Address: 0x7B Default: 0x98

比特	名称	属性	复位值	描述
7:0	sync_id_7	R/W	8'h98	同步字字节 7

5.3.67 Reg7C Address: 0x7C Default: 0xF3

比特	名称	属性	复位值	描述
7:0	sync_id_8	R/W	8'hF3	同步字字节 8

5.3.68 Reg7D Address: 0x7D Default: 0x0E

比特	名称	属性	复位值	描述
7:0	payload_len[7:0]	R/W	8'h0E	payload 长度，在数据包模式 1 和数据包模式 3 有效

5.3.69 Reg7E Address: 0x7E Default: 0x00

比特	名称	属性	复位值	描述
7:0	payload_len[15:8]	R/W	8'h00	payload 长度，在数据包模式 1 和数据包模式 3 有效

5.3.70 Reg7F Address: 0x7F Default: 0x00

比特	名称	属性	复位值	描述
7:1	RSV	R/W	7'h0	-
0	page_sel	R/W	1'b0	寄存器页地址： 0: Reg00 ~ Reg7E 1: Reg80 ~ RegFE

5.3.71 Reg80 Address: 0x80 Default: 0x1D

比特	名称	属性	复位值	描述
7:6	RSV	R/W	2'b00	-
5	m3_length_en	R/W	1'b0	数据包模式 3 时，数据长度域控制： 0：禁止 1：使能
4	m3_crc_sel	R/W	1'b1	数据包模式 3 时，CRC 的校验范围： 0：All payload 1：仅 FIFO 中的数据
3:0	m3_ack_node_ctrl	R/W	4'hD	数据包模式 3 且 AUTO_ACK 功能开启，返回的 ACK 包控制： [3]：为“1”时使能返回接收到的数据长度 [2]：为“1”时使能返回接收到的地址 [1]：为“1”时使能返回接收到的 seqnum [0]：为“1”时使能返回接收到的 FCS2 数据

5.3.72 Reg81 Address: 0x81 Default: 0x78

比特	名称	属性	复位值	描述
7:6	RSV	R/W	2'b01	-
5:4	m3_addr_size	R/W	2'b11	数据包模式 3 时，地址长度： 00：1 字节 01：2 字节 10：3 字节 11：4 字节
3	RSV	R/W	1'b1	-
2	m3_addr_split_mode	R/W	1'b0	数据包模式 3 时，将地址分为源地址和目的地址，在返回 ACK 时，将收到的源地址和目的地址位置互换： 0：不互换 1：互换

比特	名称	属性	复位值	描述
1	m3_addr_pos_sel	R/W	1'b0	数据包模式 3 时, 地址位置选择: 0: address 放在 length 之后 1: address 放在 length 之前
0	m3_addr_en	R/W	1'b0	数据包模式 3 时, 地址域控制: 0: 禁止 1: 使能

5.3.73 Reg82 Address: 0x82 Default: 0x00

比特	名称	属性	复位值	描述
7:0	m3_addr_val[7:0]	R/W	8'h00	数据包模式 3 时, 地址值

5.3.74 Reg83 Address: 0x83 Default: 0x00

比特	名称	属性	复位值	描述
7:0	m3_addr_val[15:8]	R/W	8'h00	数据包模式 3 时, 地址值

5.3.75 Reg84 Address: 0x84 Default: 0x00

比特	名称	属性	复位值	描述
7:0	m3_addr_val[23:16]	R/W	8'h00	数据包模式 3 时, 地址值

5.3.76 Reg85 Address: 0x85 Default: 0x03

比特	名称	属性	复位值	描述
7:0	m3_addr_val[31:24]	R/W	8'h03	数据包模式 3 时, 地址值

5.3.77 Reg86 Address: 0x86 Default: 0x33

比特	名称	属性	复位值	描述
7:0	m3_addr_bit_mask[7:0]	R/W	8'h33	数据包模式 3 时, 地址过滤, mask 为“1”时对应地址位不作比较

5.3.78 Reg87 Address: 0x87 Default: 0x33

比特	名称	属性	复位值	描述
7:0	m3_addr_bit_mask[15:8]	R/W	8'h33	数据包模式 3 时, 地址过滤, mask 为“1”时对应地址位不作比较

5.3.79 Reg88 Address: 0x88 Default: 0x10

比特	名称	属性	复位值	描述
7:0	m3_addr_bit_mask[23:16]	R/W	8'h10	数据包模式 3 时, 地址过滤, mask 为“1”时对应地址位不作比较

5.3.80 Reg89 Address: 0x89 Default: 0x00

比特	名称	属性	复位值	描述
7:0	m3_addr_bit_mask[31:24]	R/W	8'h00	数据包模式 3 时, 地址过滤, mask 为“1”时对应地址位不作比较

5.3.81 Reg8A Address: 0x8A Default: 0x00

比特	名称	属性	复位值	描述
7:6	RSV	R/W	2'b00	-
5	m3_seqnum_auto_inc	R/W	1'b0	数据包模式 3 时, seqnum 配置方式: 0: seqnum 数据由{Reg8c,Reg8b}配置 1: 由内部 seqnum 计数器自动加 1
4	m3_seqnum_en	R/W	1'b0	数据包模式 3 时, seqnum 数据使能: 0: 禁止 1: 使能
3	m3_seqnum_bit_size	R/W	1'b0	数据包模式 3 时, seqnum 数据长度: 0: 8bits 1: 16bits
2:0	RSV	R/W	3'b000	-

5.3.82 Reg8B Address: 0x8B Default: 0x00

比特	名称	属性	复位值	描述
7:0	seqnum_Reg[7:0]	R/W	8'h00	数据包模式 3 时, seqnum 值

5.3.83 Reg8C Address: 0x8C Default: 0x50

比特	名称	属性	复位值	描述
7:0	seqnum_Reg[15:8]	R/W	8'h50	数据包模式 3 时, seqnum 值

5.3.84 Reg8D Address: 0x8D Default: 0x30

比特	名称	属性	复位值	描述
7:1	RSV	R/W	7'h30	-
0	m3_fcs2_en	R/W	1'b0	数据包模式 3 时, FCS2 数据域使能: 0: 禁止 1: 使能

5.3.85 Reg8E Address: 0x8E Default: 0x32

比特	名称	属性	复位值	描述
7	m3_fcs2_val	R/W	1'b0	数据包模式 3 时, ACK request
6:0	RSV	R/W	7'h32	-

5.3.86 Reg94 Address: 0x94 Default: 0x08

比特	名称	属性	复位值	描述
7:5	RSV	R/W	3'b000	-
4	crc_bit_order	R/W	1'b0	CRC bit 顺序: 0: LSB 1: MSB
3	crc_inv	R/W	1'b1	CRC bit 取反: 0: 禁止 1: 使能

比特	名称	属性	复位值	描述
2	crc_man_en	R/W	1'b0	CRC 曼彻斯特编码使能: 0: 禁止 1: 使能
1:0	crc_len	R/W	2'b00	CRC 长度: 00: 8 位 01: 16 位 10: 24 位 11: 32 位

5.3.87 Reg95 Address: 0x95 Default: 0x20

比特	名称	属性	复位值	描述
7:0	crc_poly[7:0]	R/W	8'h20	CRC 多项式配置

5.3.88 Reg96 Address: 0x96 Default: 0x26

比特	名称	属性	复位值	描述
7:0	crc_poly[15:8]	R/W	8'h26	CRC 多项式配置

5.3.89 Reg97 Address: 0x97 Default: 0x1F

比特	名称	属性	复位值	描述
7:0	crc_poly[23:16]	R/W	8'h1F	CRC 多项式配置

5.3.90 Reg98 Address: 0x98 Default: 0x68

比特	名称	属性	复位值	描述
7:0	crc_poly[31:24]	R/W	8'h68	CRC 多项式配置

5.3.91 Reg99 Address: 0x99 Default: 0x04

比特	名称	属性	复位值	描述
7:0	crc_init_val[7:0]	R/W	8'h04	CRC 移位寄存器初始值

5.3.92 Reg9A Address: 0x9A Default: 0xAE

比特	名称	属性	复位值	描述
7:0	crc_init_val[15:8]	R/W	8'hAE	CRC 移位寄存器初始化值

5.3.93 Reg9B Address: 0x9B Default: 0x89

比特	名称	属性	复位值	描述
7:0	crc_init_val[23:16]	R/W	8'h89	CRC 移位寄存器初始化值

5.3.94 Reg9C Address: 0x9C Default: 0x84

比特	名称	属性	复位值	描述
7:0	crc_init_val[31:24]	R/W	8'h84	CRC 移位寄存器初始化值

5.3.95 Reg9D Address: 0x9D Default: 0x03

比特	名称	属性	复位值	描述
7	tx_data_inv	R/W	1'b0	发射 bit 数据取反
6	rx_data_inv	R/W	1'b0	解调器输出的 bit 数据取反
5	auto_det_tx_ch1	R/W	1'b0	发射前检测信道 RSSI: 0: 禁止 1: 使能
4	auto_det_tx_mode	R/W	1'b0	发射检测信道模式: 0: 检测次数满且信道仍然忙, 退出发射模式, 发射数据失败 1: 持续检测, 直到发射数据成功
3:0	auto_det_times	R/W	4'h3	发射前最大检测信道次数

5.3.96 Reg9E Address: 0x9E Default: 0x80

比特	名称	属性	复位值	描述
7:0	chl_busy_thr	R/W	8'h80	发射前检测信道门限值

5.3.97 Reg9F Address: 0x9F Default: 0x26

比特	名称	属性	复位值	描述
7:6	RSV	R/W	2'b00	-
5:0	fifo_empty_thr	R/W	6'h26	TX FIFO 空门限, FIFO 数据低于门限值时 fifo_flag 置“1”

5.3.98 RegA0 Address: 0xA0 Default: 0x50

比特	名称	属性	复位值	描述
7:6	RSV	R/W	2'b00	-
5:0	fifo_full_thr	R/W	6'h50	RX FIFO 满门限, RX FIFO 数据高于门限值时 fifo_flag 置“1”

5.3.99 RegA3 Address: 0xA3 Default: 0x42

比特	名称	属性	复位值	描述
7	srst_cmd_enable	R/W	1'b0	Reset Command 使能: 0: 禁止 1: 使能
6	RSV	R/W	1'b1	-
5:0	auto_ack_wait_time	R/W	6'h02	auto ack 功能开启时, 发射端发完数据包后等待 ACK 数据的时间, 超时会重新发射数据

5.3.100 RegE0 Address: 0xE0 Default: 0x80

比特	名称	属性	复位值	描述
7:0	rx_seqnum_val[7:0]	R	8'h80	数据包模式 3 时, 接收 seqnum 数据

5.3.101 RegE1 Address: 0xE1 Default: 0x00

比特	名称	属性	复位值	描述
7:0	rx_seqnum_val[15:8]	R	8'h00	数据包模式 3 时, 接收 seqnum 数据

5.3.102 RegE2 Address: 0xE2 Default: 0x01

比特	名称	属性	复位值	描述
7:0	seqnum_cnt[7:0]	R	8'h01	seqnum_cnt

5.3.103 RegE3 Address: 0xE3 Default: 0x40

比特	名称	属性	复位值	描述
7:0	seqnum_cnt[15:8]	R	8'h40	seqnum_cnt

5.4 功能描述

RF 高集成度的 Sub-1GHz 无线收发器。支持 OOK，2-(G)FSK 调制解调方式，支持 Direct 和 Packet 数据处理模式，支持通过 SPI 对 RF 进行控制。

5.4.1 接收机

RF 内部集成了低功耗、高性能的低中频架构接收机。天线接收到的射频信号经过低噪声放大器放大之后，由正交混频器下变频至中频。I/Q 两路中频信号进一步通过低通滤波器（LPF）、可编程放大器（PGA）放大到合适的幅度，然后由 ADC 转换到数字域。

接收到的信号在数字域完成镜像抑制、中频滤波、基带滤波和 OOK/(G)FSK 解调。解调后的数据在直通模式下可以通过 GPIO 引脚直接输出，也可以在包模式下通过 SPI 从 RF 内部 FIFO 中读取。

5.4.2 发射机

RF 发射机采用基于频率综合器的单点调制结构。内部的高效率功率放大器可以输出最大 +18dBm 的功率，输出功率可以在 -20 ~ +18dBm 范围内调节，调节精度为 1dB。在 GFSK 模式下，发射数据经过高斯滤波之后再送入频率综合器进行调制，使得发射频谱更为集中。为了降低 PA 开关过程中引起的频谱杂散和毛刺，并削弱对 VCO 的牵引，PA 的输出功率引入了缓慢升降机制（PA Ramp）。

发射机支持包模式和直通模式。包模式下，数据先填入 RF 内部的 FIFO，芯片会自动按照配置完成相应的包处理然后由发射机发射出去。直通模式下，发射数据可以通过 GPIO 直接输入并发射出去。

5.4.3 系统复位

RF 内部集成了上电复位 (POR)、外部复位和软件复位。上电默认状态下，GPIO2 默认为外部复位信号(nRST)输入，只要 nRST 拉低就可以完成对 RF 的复位。此外，还可以通过 SPI 写命令的方式对 RF 进行复位操作。RF 一旦复位，内部所有配置都恢复到上电默认状态，需要对 RF 重新进入初始化操作。

5.4.3.1 POR 复位

芯片上电或者 RF 退出全局关断模式 (SDN 拉低) 后，RF 均会触发上电复位 (POR)，复位之后 RF 默认处于 IDLE 状态。上电稳定或 SDN 拉低后，至少等待 4.5ms 的时间等待晶体启动稳定后，再进行 SPI 操作 RF。

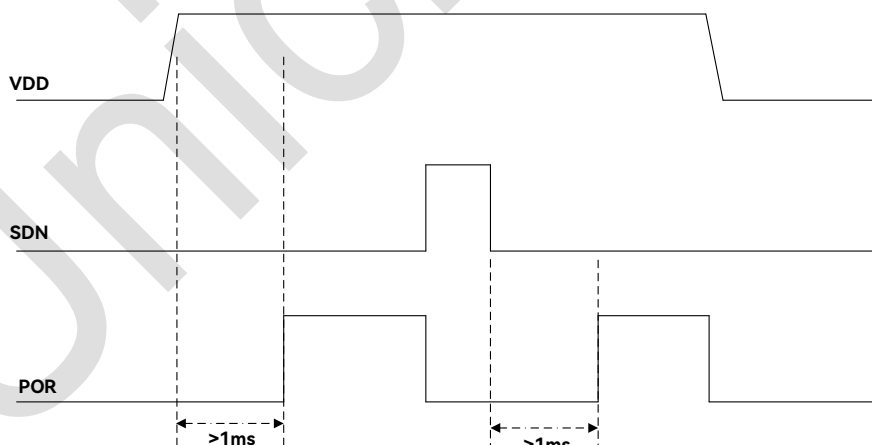


图 5-1: POR 复位时序图

5.4.3.2 外部复位

芯片上电 GPIO2 默认配置为外部 nRST 复位功能，通过拉低 nRST 引脚电平使 RF 处于复位状态，拉高 nRST 引脚电平使 RF 处于工作状态。如果 GPIO2 配置为其他功能，需要先关闭 GPIO2 的

复位功能，否则可能存在配置过程中使 RF 复位。RF 外部复位后，内部所有配置都恢复到默认状态，需要重新进行配置。

表 5-1：外部复位相关寄存器

寄存器地址	位数	R/W	比特名	功能说明
0x5D	7	R/W	nrst_sel	nRST 引脚功能设置： 0：nRST 1：GPIO2
	6	R/W	pull_up_en	nRST 引脚 Pull-UP 使能： 0：使能 1：禁止

5.4.3.3 软件复位

软件复位是通过 SPI 写命令的方式对 RF 进行软复位操作，RF 除了 Shutdown 和 Sleep 状态外的所有状态，都可以通过 SPI 写命令进行软复位操作，写复位命令前需要先使能软件复位。软复位是复位 RF 整个数字系统，包括 SPI 接口、寄存器、FIFO、数字内核。RF 复位后回到 IDLE 状态，所有寄存器丢失，需要重新初始化系统。

软复位流程如下：

- 1. SPI 写指令，RegA3=0xC2，使能复位命令。
- 2. SPI 写指令，Reg60=0x06，写复位命令进行复位。

表 5-2：软复位相关寄存器

寄存器地址	位数	R/W	比特名	功能说明
0x60	7:0	R/W	command	0x06：Reset command
0xA3	7	R/W	srst_cmd_enable	Reset Command 使能： 0：禁止 1：使能

5.4.4 频率综合器

RF 内部集成了高精度的 Sigma-Delta 小数分频频率综合器，在 200 ~ 960MHz 范围内产生精准的载波频率。载波频率可以通过两种方式进行设置：

- 1、寄存器直接设置：在这种方式下，把所需的载波频率值直接写入相应的寄存器即可。
- 2、查表方式。在这种方式下，可以按照以下公式进行频率设置：

$$F_c=F_0+N*F_{step}$$

其中，F₀、N 和 F_{step} 都可以通过寄存器进行设置。

- F₀ 由{Reg00[6:0],Reg01,Reg02,Reg03}共 31bit 设置，高 11 位表示整数，低 20 位表示小数，单位为 MHz。
- N 为整数由 Reg04 设置。
- F_{step} 由{Reg05,Reg06,Reg07}设置，低 20bit 为小数，单位为 MHz。

如设置 F_c=433.92MHz，可以设置 F₀ = 433.92，N = 0，F_{step} = 0。即写入以下寄存器值：

Reg00=0x1B
Reg01=0x1E
Reg02=0xB8
Reg03=0x51
Reg04=0x00
Reg05=0x00
Reg06=0x00
Reg07=0x00

表 5-3：频率综合器相关寄存器

寄存器地址	位数	R/W	比特名	功能说明
0x00	6:0	R/W	rf_f0[30:24]	信道频率 f0[30:24]。 信道频率 rf_freq 设置公式： rf_freq=rf_f0+ch_num*f_step rf_f0 由{Reg00,Reg01,Reg02,Reg03}配置，单位为 MHz，低 20 位为小数； ch_num 由 Reg04 配置； 步进频率 f_step 由{Reg05,Reg06,Reg07}配置，单位为 MHz，低 20 位为小数。
0x01	7:0	R/W	rf_f0[23:16]	信道频率 f0[23:16]
0x02	7:0	R/W	rf_f0[15:8]	信道频率 f0[15:8]
0x03	7:0	R/W	rf_f0[7:0]	信道频率 f0[7:0]

寄存器地址	位数	R/W	比特名	功能说明
0x04	7:0	R/W	ch_num	信道号 ch_num, 参考 Reg00 描述
0x05	7:0	R/W	f_step[23:16]	信道步进频率 f_step[23:16], 单位为 MHz, 低 20 位为小数, 参考 Reg00 描述
0x06	7:0	R/W	f_step[15:8]	信道步进频率 f_step[15:8], 参考 Reg00 描述
0x07	7:0	R/W	f_step[7:0]	信道步进频率 f_step[7:0], 参考 Reg00 描述
0x20	7:6	R/W	frequency_band	工作频段设置: 00: 800MHz~1GHz 频段 01: 400MHz~500MHz 频段 10: 267MHz~350MHz 频段 11: 200MHz~250MHz 频段

5.4.5 晶体振荡器

晶体振荡器用于为 RF 的锁相环提供基准时钟，为 RF 数字模块提供系统时钟。负载电容取决于晶体指定 C_L 参数。XI 与 XO 之间的总的负载电容应该等于 CL ，以使晶体准确振荡在参考值范围内。

$$C_L = \frac{1}{\frac{1}{C_1} + \frac{1}{C_2}} + C_{PCB} + C_{IC}$$

C_1 和 C_2 分别为晶体两端挂的负载匹配电容， C_{PCB} 为 PCB 寄生引入的等效电容（约 0.5pF）。 C_{IC} 为芯片 XI 和 XO 引脚引入的等效电容（约 2.5pF）。晶体的等效串联电阻应在指定规格之内，以保证晶体能可靠起振。也可以用外部信号源连接至 XI 引脚来取代传统的晶体。外部时钟信号的峰峰值在 300 ~ 700mV 之间，并通过隔直电容耦合到 XI 引脚。晶体的精度越高，接收性能越好。RF 推荐使用负载电容（CL）为 20pF，精度为 10ppm 的 26MHz 晶体，板级外挂的负载匹配电容 C_1 和 C_2 的容值约 20pF。

RF 发射和接收的频点可以通过调整负载匹配电容和软件修正频点，软件修正通过寄存器 Reg08/Reg09/Reg0A/Reg0B，其中低 24 位为小数。

表 5-4：晶振相关寄存器

寄存器地址	位数	R/W	比特名	功能说明
0x08	6:0	R/W	ref_freq[30:24]	参考频率（晶振频率）设置，ref_freq 由 {Reg08[6:0],Reg09,Reg0A,Reg0B}配置，单位为 MHz，低 24 位为小数部分。

寄存器地址	位数	R/W	比特名	功能说明
0x09	7:0	R/W	ref_freq[23:16]	参考 Reg08 描述
0x0A	7:0	R/W	ref_freq[15:8]	参考 Reg08 描述
0x0B	7:0	R/W	ref_freq[7:0]	参考 Reg08 描述

5.4.6 AGC

RF 的接收通路集成了 AGC 功能，Mixer、LPF、PGA 的增益受 AGC 环路调节。AGC 的环路控制在数字域完成，通过设定合理的 AGC 控制参数，RF 接收机的灵敏度、选择性和线性度可以达到最佳的性能。

表 5-5: AGC 相关寄存器

寄存器地址	位数	R/W	比特名	功能说明
0x28	4	R/W	agc_en	AGC 使能： 0：禁止 1：使能

5.4.7 RSSI

RF 内部集成的输入信号强度指示 (RSSI) 功能可以对天线端接收到的信号强度进行评估。RSSI 检测必须在 RX 状态下进行，检测到的是信道内的信号强度。

RF 内置信号检测模块，用于接收数据包时测量 ADC 的数据信号强度 RSSI_RAW，在 AGC（自动增益控制）使能时，使用 RSSI_RAW 值对接收链路增益自动调节，同时可自动计算出 RF 输入端的 RSSI（信号强度指示）值。也可以使用 RSSI_RAW 值手动调节（AGC 关闭）接收链路增益，并根据增益值自行计算 RSSI。RSSI_RAW 值或自动计算的 RSSI 值可以从寄存器 Reg64 读取。

表 5-6: RSSI 相关寄存器列表

寄存器地址	位数	R/W	比特名	功能说明
0x10	3	R/W	rss_val_sel	读取 RSSI 时，选择读取的数据内容： 0：经过计算的 RSSI 值 1：原始信号幅度值
0x28	7	R/W	rss_Reg_clr	写“1”清除被锁定的 RSSI
	6	R/W	rss_sel	收到同步字后锁定 RSSI 值： 0：禁止 1：使能

寄存器地址	位数	R/W	比特名	功能说明
	5	R/W	rx_rssi_thr_en	接收时判断 RSSI，如果收到前导码的 RSSI 低于设置的 RSSI 门限值(Reg2C)，会复位解调器重新接收： 0：禁止 1：使能
0x2C	7:0	R/W	rx_rssi_thr	接收 RSSI 门限值，参考 Reg28[5]
0x56	7:0	R/W	delta_rssi	计算 RSSI 时的预偏移量
0x64	7:0	R	rssi	读出的值为实际 RSSI 的绝对值

5.4.7.1 自动计算 RSSI

自动计算 RSSI 需使能 AGC，芯片通过内部自动增益计算得到，在进入接收状态后，可以读取 Reg64 寄存器的值 rssi_value[7:0]，步进为 0.5dBm，且是真实信号 RSSI 的绝对值，其计算公式如下：

$$RSSI = -(rssi_value/2)$$

其寄存器优化值如下：

- Reg10[3] = 0
- Reg28[4] = 1
- Reg56 = 0xB7

RF 在整个接收数据包期间，RSSI 值实时更新到寄存器 Reg64，当退出接收状态，Reg64 值也恢复到 0；如果设置寄存器 Reg28[6]=1，则收到同步字后寄存器 Reg64 值锁定，RSSI 值不再更新，直到下次重新进入接收时 RSSI 才会被更新。

RF 在自动计算 RSSI 模式下，输入功率值与读出的 RSSI 值有很好的对应关系。如果在某些频段或速率在线性区间内读出的值与输入功率值有固定的差值，可以写寄存器 Reg56 微调偏移量。下图为在 $F_{RF}=433MHz$ ， $DR=10kbps$ ， $F_{DEV}=22K$ 条件下实测的 RSSI 值对应的输入功率。从图上看信号强度在[-120dBm,-35dBm]区间，RSSI 与信号强度存在线性关系。

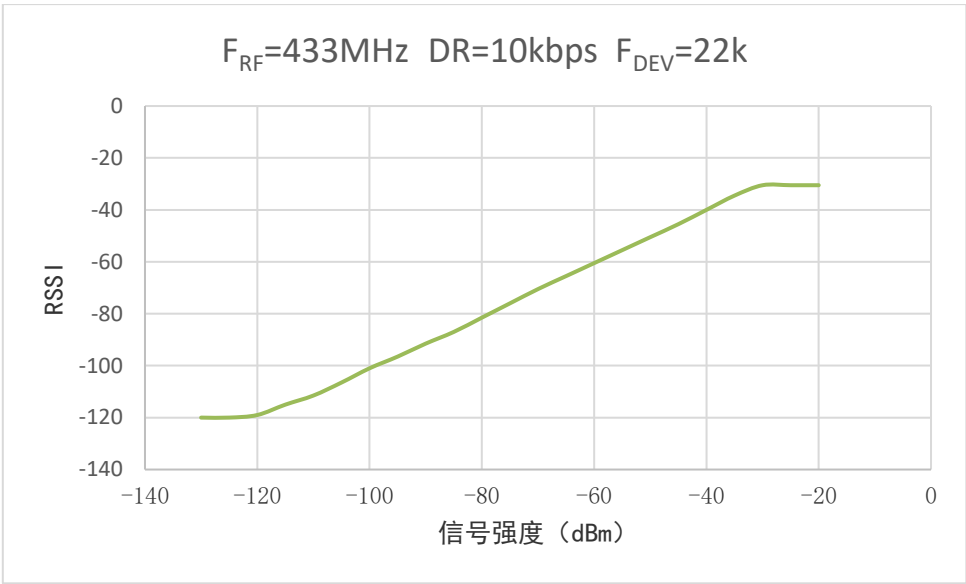


图 5-2: RSSI 线性图

5.4.7.2 手动调节增益

在正常(G)FSK 模式时，都使用自动增益调节。在 OOK 模式时，可以手动调节增益并计算 RSSI 值。其计算公式为： $RSSI = (Reg64/2) - Gain + \Delta_{RSSI}$

其寄存器优化值设置如下：

- $Reg10[3] = 1$
- $Reg28[6] = 0$
- $Reg28[4] = 0$
- $Reg2B[5:0] = 0x3F$

进入接收状态后，读取 Reg64 的值，根据 Reg64 的值上下调节 Reg2B 的增益控制。Reg2B 对应增益表如表 5-7 所示：

表 5-7: Reg2B 增益控制表

Reg2B[5:0]	Gain (dBm)
111111	54
111110	48
111101	42
111100	36
111000	30

Reg2B[5:0]	Gain (dBm)
110100	24
110000	18
100000	12
010000	6
000000	0

5.4.7.3 RSSI 自动过滤功能

RSSI 自动过滤功能，即设置 RSSI 门限值，在解调器接收到前导信号时，会判断 RSSI 值，如接收到的信号强度低于设置的门限，则解调器自动复位，丢弃该信号重新进入接收状态。

如设置信号强度低于-96dBm 时，丢弃该包数据，则进入接收之前设置：

- Reg28[5] = 1
- Reg2C[7:0] = 0xC0

5.4.8 AFC

在接收模式下，自动频率控制（AFC）可有效补偿晶体偏差导致的载波频率失配问题。当AFC功能启用后，接收信号将被精准校准至中频滤波器的通带范围内，从而在更宽泛的频率偏移区间内实现灵敏度与选择性的最优平衡。值得注意的是,AFC的正常工作依赖于足够长度的前导码(preamble)，通常需额外增加一定长度（推荐4个字节）的前导码序列，以确保AFC完成频率校准。

表 5-8: AFC 相关寄存器

寄存器地址	位数	R/W	比特名	功能说明
0x37	7:0	R/W	csrst_len	在补偿功能使能时，找到信号的频谱后连续收到 Reg37*4 个前导后，则认为找到有效的 Preamble
0x38	4	R/W	demod_comp_en	解调频偏补偿： 0：禁止 1：使能

5.5 运行状态

5.5.1 SPI 接口

MCU 通过 SPI1 与 RF 通信，其中 RF 作为 SPI1 从机模块，MCU 的 SPI1 作为主机。通信速率最大 16MHz 工作运行。SPI 接口模式的时钟极性为正，在时钟下降沿采样数据 (CPOL=0, CPHA=1)，在时钟上升沿输出数据，地址和数据部分都是从 MSB 开始传送。访问 RF 都是以 SPI 读写寄存器的方式，第一个字节为地址，后面跟一个字节的数据。如果在访问 FIFO 对应的地址时，可以在一个 SPI_CS 为低的周期内按字节方式连续的访问，SPI 接口控制器会自动增加访问地址，在访问 FIFO 数据时，地址和数据之前至少要等 3 个芯片的系统时钟，以便芯片确定 FIFO 指针地址。

在没有晶振时钟时，RF 不能写寄存器，但仍然可以读寄存器数据。当访问寄存器的时候，SPI_CS 要拉低。然后首先发送一个 R/W 位，之后是 7 位的寄存器地址。RW = 0 表示写，RW = 1 表示读。

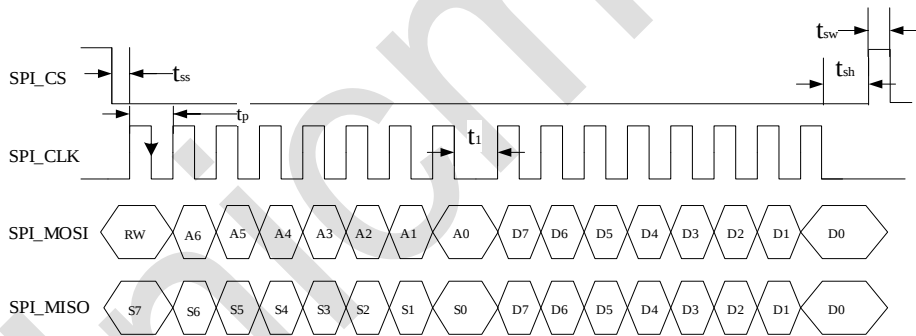


图 5-3: SPI 读写寄存器时序图

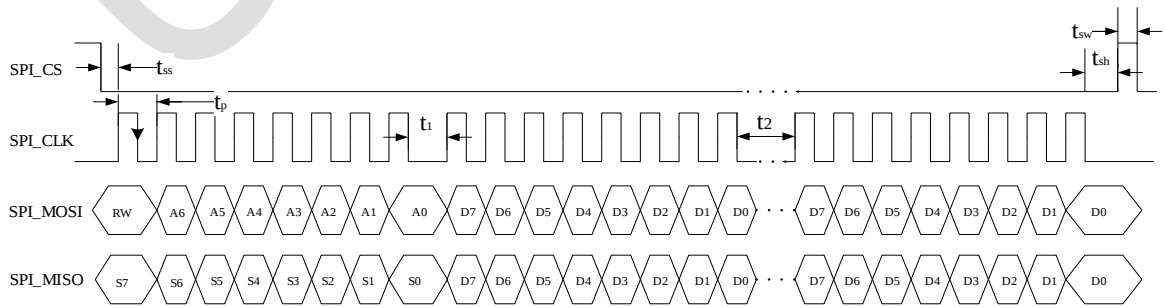


图 5-4: SPI 读写 FIFO 时序图

表 5-9: SPI 时序参数

符号	最小值	说明
t _{sw}	100ns	两次 SPI 访问的间隔时间
t _{ss,t_{sh}}	32ns	SPI_CS 和 SPI_CLK 的间隔时间
t ₁	32ns	地址和数据间隔时间
t ₂	32ns	两个寄存器数据读取时间间隔
t _p	64ns	SPI_CLK 时钟周期

5.5.2 FIFO

RF 内置 256 字节 FIFO，分为两组 128 字节的 FIFO 作为发射和接收独立使用。FIFO 独立使用时，发射 FIFO 的数据不会被更新掉，下次如要发同样的数据就不需要再重新写入数据到 FIFO 了。也可以通过寄存器 Reg0E[3]设置成 256 字节共用模式，发射和接收都有 256 字节的 FIFO 可用。

在重新写一帧数据到发射 FIFO 时，应对发射 FIFO 写指针进行清零。在重新读一帧接收 FIFO 数据时，应对接收 FIFO 读指针进行清零。接收写指针和发射读指针芯片内部会自动清零。

表 5-10: FIFO 相关寄存器列表

寄存器地址	位数	R/W	比特名	功能说明
0x0E	3	R/W	fifo_share_en	FIFO 共享设置： 0: RX 和 TX 各 128 字节 FIFO 1: RX 和 TX 共用 256 字节 FIFO
0x50	7:0	R/W	rx_fifo_wr_ptr	RX FIFO 写指针，写 0x80 清除指针
0x51	7:0	R/W	rx_fifo_rd_ptr	RX FIFO 读指针，写 0x80 清除指针
0x52	7:0	R/W	rx_fifo_data	RX FIFO
0x53	7:0	R/W	tx_fifo_wr_ptr	TX FIFO 写指针，写 0x80 清除指针
0x54	7	R/W	tx_fifo_rd_ptr	TX FIFO 读指针，写 0x80 清除指针
0x55	7:0	R/W	tx_fifo_data	TX FIFO
0x61	3	R	fifo_flag	fifo_flag
0x9F	5:0	R/W	fifo_empty_thr	TX FIFO 空门限，FIFO 数据低于门限值时 fifo_flag 置“1”
0xA0	5:0	R/W	fifo_full_thr	RX FIFO 满门限，RX FIFO 数据高于门限值时 fifo_flag 置“1”

5.5.2.1 FIFO 的工作模式

RF默认提供两个独立的128-byte的FIFO，分别给RX和TX使用，两者互不相干。用户也可以将fifo_share_en设为1，那么两个FIFO即可合成一个256-byte的FIFO，在TX和RX下共同使用。

5.5.2.2 FIFO 的中断时序

FIFO 的中断包括空中断和满中断，对于 TX，设置 fifo_empty_thr 为将空阈值，只要发送的数据低于此门限值，fifo_flag 产生空中断。对于 RX，设置 fifo_full_thr 为将满阈值，只要 FIFO 接收的剩余空间低于此门限值，fifo_flag 产生满中断。

5.5.2.3 FIFO 的应用场景

- 预先填好数据，进入 TX 发射。

该场景适合发射的数据小于 FIFO 的大小，其流程如下：

1. 初始化寄存器优化值、数据率以及数据包的配置。
2. 清 TX FIFO 写指针：Reg53[7:0] = 0x80
3. 把数据写入 TX FIFO：Reg55 = data
4. 启动发射：Reg60 = 0x40
5. 发射完成：发射完成的判断可查询 Reg61[4]，如果为 1 表示发射完成。也可以通过 nIRQ 这个引脚产生的上升沿中断，来判断发射完成。
6. 如果需要再发射新的数据，重新清 TX FIFO 写指针，写入 TX FIFO 数据，然后启动发射。

- 在 RX 下接收数据

该场景适用于接收数据长度小于 FIFO 大小，其流程如下：

1. 初始化寄存器优化值、数据率以及数据包的配置。
2. 写 Reg60 = 0x20 进入接收，等 MCU 收到 nIRQ 中断。
3. 读 Reg61 寄存器，判断 Reg61[7:4] 是否为 0xD，如果正确表明收到一包数据。
4. 清 RX FIFO 读指针，Reg51 = 0x80。

5. 读 RX FIFO 数据 Read Reg(0x52)。
6. 读 RSSI (Reg0x64): 根据需要可读出该包的 RSSI 值。
7. 写接收命令重新进入接收数据状态。

● 进入 TX 后, 边填数据边发射

该场景适用于每次发送数据超出 FIFO 长度的数据包, 在启动发射后, 需要监控 fifo_flag 信号, 该信号可以配置到引脚输出, 也可以通过读 Reg61 寄存器获取; 在 TX FIFO 快空时要及时写入数据到 TX FIFO, 保证发射数据流的连续性。其流程如下:

1. 初始化寄存器优化值、数据率以及数据包的配置。
2. 清 TX FIFO 写指针: Reg53[7:0] = 0x80
3. 把数据写入 TX FIFO: Reg55 = data
4. 启动发射: Reg60 = 0x40
5. 等到 FIFO_flag 信号或者查询 Reg61[3]是否等于 1, 如产生中断或 Reg61[3]=1,则继续写往 FIFO 写入数据。
6. 直至所有数据发射完成: 发射完成的判断可查询 Reg61[4], 如果为 1 表示发射完成。也可以通过 nIRQ 这个引脚产生的上升沿中断, 来判断发射完成。
7. 如果需要再发射新的数据, 重新清 TX FIFO 写指针, 写入 TX FIFO 数据, 然后启动发射。

● 进入 RX 后, 边读 FIFO 数据边接收

该场景适用于每次接收数据长度超出 FIFO 长度的数据包, 在启动接收后, 需要监控 fifo_flag 信号, 该信号可以配置到引脚输出, 也可以通过读 Reg61 寄存器获取; 在 RX FIFO 快满时要及时读出数据, 保证接收到的数据不被覆盖。其流程如下:

1. 初始化寄存器优化值、数据率以及数据包的配置。
2. 使能 FIFO_flag 中断, 并清 RX FIFO 读指针, Reg51 = 0x80。
3. 写 Reg60 = 0x20 进入接收, 需要监听 NIRQ 和 FIFO_flag 中断或者查询 Reg61。
4. 如接收到 FIFO_flag 中断或者查询 Reg61[3]=1,则需要及时读取 FIFO 数据, 避免接收到的数据不被覆盖。

5. 如接收到 nIRQ 中断或者读 Reg61 寄存器。判断 Reg61[7:4] 是否为 0xD，如果正确表明收到一包数据。
6. 需要读取剩余 FIFO 的数据，读 RX FIFO 数据 Read Reg(0x52)。
7. 读 RSSI (Reg0x64)：根据需要可读出该包的 RSSI 值。
8. 写接收命令重新进入接收数据状态。

5.5.3 GPIO 和中断

RF 内部集成有 3 个 GPIO，分别是 GPIO0、GPIO2、nIRQ，每个 GPIO 都可以配置成不同的输入或者输出。GPIO 能产生两种中断信号，pkt_flag 和 FIFO_flag，该两个中断信号标志位都可读。pkt_flag 分为 4 个功能：前导匹配、同步字匹配、接收或发送包完成。FIFO_flag 表示 FIFO full 或 empty，在发送模式时表示 FIFO empty，在接收模式时表示 FIFO full。

表 5-11: GPIO 配置相关寄存器列表

寄存器地址	位数	R/W	比特名	功能说明
0x11	2:0	R/W	brclk_sel	GPIO brclk 输出时钟选择： 000: rx_clk 接收 bit rate 时钟 001: xtal_clk 晶振时钟 010: xtal_clk/8 011: xtal_clk/16 100: 频综时钟 101: tx_clk 发射 bit rate 时钟 111: ADC_clk
0x4D	7	R/W	nirq_dir	nIRQ 口方向： 0: Output 1: Input
	6	R/W	GPIO0_dir	gpio0 方向： 0: Output 1: Input
	4	R/W	GPIO2_dir	gpio2 方向，默认复位输入功能： 0: Output 1: Input
	3:0	R/W	nirq_sel	nIRQ 引脚输出功能选择： 0x00: pkt_flag(包括数据包完成中断 pkt_int,

寄存器地址	位数	R/W	比特名	功能说明
				接收前导码中断 preamble_int,接收同步字中断 syncword_int) 0x01: pkt_int 0x02: preamble_int 0x03: syncword_int 0x04: FIFO_flag 0x05: brclk (参考 Reg11[2:0]描述) 0x06: rxdata 0x07: txdata 0x08: tr_switch 0x09: tr_switch 反向 0x0A: rxdata 0x0B: txdata 0x0D: 高电平 Others: 低电平
0x4E	7:4	R/W	gpio0_sel	gpio0 输出功能选择, 参考 Reg4D[3:0]描述
0x4F	7:4	R/W	gpio2_sel	gpio2 输出功能选择, 参考 Reg4D[3:0]描述

表 5-12: GPIO 输出功能描述

Value	Name	Description
0x00	pkt_flag	pkt_flag (包括数据包完成中断 pkt_int, 接收前导码中断 preamble_int, 接收同步字中断 syncword_int)
0x01	pkt_int	包完成中断, 包含接收完成和发射完成
0x02	preamble_int	接收 Preamble 有效中断
0x03	syncword_int	接收 Syncword 有效中断
0x04	FIFO_flag	RX FIFO 快满中断, TX FIFO 快空中断
0x05	brclk	时钟输出, 由 Reg11[2:0]设置。 Reg11[2:0]: 000: rx_clk 接收同步时钟 001: xtal_clk 晶振时钟 010: xtal_clk/8 011: xtal_clk/16 100: 频综时钟 101: tx_clk 发射同步时钟 111: ADC_clk
0x06	rxdata	解调器输出的串行数据

Value	Name	Description
0x07	txdata	进入发射调制的串行数据
0x08	tr_switch	发射使能标志
0x09	tr_switch_inv	反向发射使能标志
0x0A	rxdata	解调器输出的串行数据
0x0B	txdata	进入发射调制的串行数据
0x0D	high_level	高电平
Others	low_level	低电平

5.5.4 启动时序和状态机

芯片上电稳定或 RF 退出关断模式（SDN 拉低）后，至少需要 0.5ms 的时间，RF 的 POR 才会释放。POR 释放后，RF 默认进入 IDLE 状态，并启动相关电源和晶体，启动稳定时间大约为 4ms（和晶体的特性相关），需等待晶振稳定后才可以响应 SPI 指令。RF 一共有 6 种工作状态: Shutdown、Standby、IDLE、FSON、TX、RX。

5.5.4.1 状态机控制图

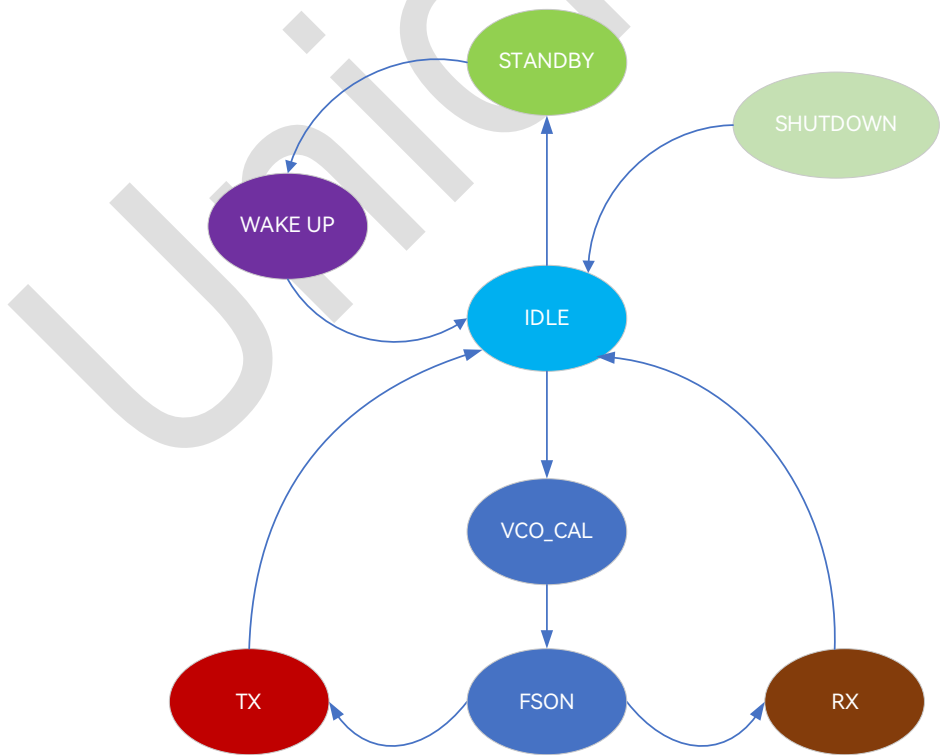


图 5-5：状态机控制图

5.5.4.2 工作模式

表 5-13: 工作模式表

状态/模式	描述	命令
SHUTDOWN	RF 处于关断状态	SDN 拉高
IDLE	RF 空闲状态，复位后进入此状态	上电复位或发送IDLE命令
STANDBY	RF 的数字模块工作，其他模块关闭，寄存器保持	SPI 写 STANDBY 命令
FSON	RF 的频综打开状态，可快速进入接收或发送状态	SPI 写 FSON 命令
RX	接收数据状态	SPI 写 RX 命令
TX	发送数据状态	SPI 写 TX 命令

5.5.4.3 状态机说明

- 关断状态（SHUTDOWN）

当 SDN 引脚拉高之后，RF 进入关闭状态，RF 内部所有的电路模块都关闭，RF 消耗电流接近 10nA。
- IDLE 状态

上电复位或发送 IDLE 命令后，RF 进入 IDLE 状态。在此状态下，晶体振荡器开启，等待 SPI 接口命令再执行其它的动作。
- STANDBY 状态

写命令进入 STANDBY 状态后，RF 的数字供电开启，RF 所有的数字电路正常供电，其它模块都关闭，寄存器值可以保持。将 RF_CS 拉低（2ms）后，RF 从 STANDBY 状态退出，进入 IDLE 状态。
- 频综打开状态（FSON）

频综打开并保持在这个状态，在此状态下，芯片收到TX/RX命令后会直接进入TX/RX状态。
- 发送状态（TX）

RF 收到发射数据包命令后，RF 先打开 PLL 及 VCO，进行校准，等待至 PLL 达到要求的发射频段，如果自动信道检测功能打开，则在进入发射数据前先进行信道检测，如果空闲则进行发送数据包，如果信道忙，则下个动作可通过寄存器设置，是直接退出发送，还是继续检测 RSSI，直到把数据包发出。当数据包发出后，如果自动应答功能开启则切换到 RX_ACK 状态，以确定

包有没有被接收方正确的接收，如果超出寄存器设定的时间没有收到应答包，则进行重发，重发最大次数可寄存器设置。

● 接收状态 (RX)

RF 收到 RX 命令后，芯片先打开 PLL，然后进行 VCO 校准并依次启动接收电路（LNA、Mixer、PGA 和 ADC）和数字解调器。当 RF 收到数据包后，会给出 nIRQ 中断指示信号然后退回到 IDLE 状态。当 AUTO_ACK 功能开启时，RF 收到数据包后会自动发射 ACK 信号然后再退回到 IDLE 状态。

5.6 数据包及包处理机制

芯片提供灵活可配置的数据控制模式，主要分为如下两大类：

- Direct 直通模式，在 TX 模式下，发射的串行数据直接从 GPIO 输入到发射模块将数据发出，在 RX 模式下，接收的串行数据可直接从 GPIO 口输出。
- Packet 数据包模式，在数据包模式下所有数据都要经过 FIFO，且支持各种数据包模式的控制，数据包模式又可以分成四种帧格式。

5.6.1 直通模式 (Direct)

数据直通 (direct) 使指 RF 通过 GPIO 输入发射数据或获取接收数据，在 TX/RX 启动前使能 Direct Mode，且将包模式配置为 Mode0，RF 的 TX/RX 启动或停止完全由 MCU 的命令控制。

表 5-14: Direct 相关寄存器列表

寄存器地址	位数	R/W	比特名	功能说明
0x0E	2	R/W	Direct_mode	直通模式使能： 0：禁止 1：使能
	1:0	R/W	Packet_mode	数据包控制模式： 00：模式 0，循环发射 TX FIFO 中的数据，需要写命令退出接收或发射状态
0x4D	7	R/W	nirq_dir	nIRQ_flag IO 口方向： 0：Output

寄存器地址	位数	R/W	比特名	功能说明
				1: Input
	6	R/W	gpio_dir[0]	gpio[0] 方向: 0: Output 1: Input
0x58	4:3	R/W	direct_tx_data_sel	direct mode 时, TX 数据源选择: 00: nIRQ 引脚 01: gpio[0]

5.6.1.1 直通模式操作流程

● RX 直通接收

在直通模式中，数据从解调器的输出直接通过 GPIO 输出，输出端口可以设置为 GPIO0/GPIO2/nIRQ。典型的直通模式的 RX 工作顺序如下：

1. 寄存器配置 GPIO 为输出状态。
2. 寄存器配置接收直通模式并配置模式为 Mode0。
3. 发送 RXON 命令进入接收状态。
4. 接收到数据解调后从 GPIO 输出。
5. 发送 Sleep / IDLE 来完成接收，节省功耗。

● TX 直通发射

直通模式下，待发射的数据直接 GPIO 送入芯片，数据率只要在芯片使用规范内都能支持发射，典型的直通模式的 TX 工作顺序如下：

1. 设置寄存器使能 GPIO 的直通数据功能并设置为输入模式。
2. 设置发射直通模式并设置模式为 Mode0。
3. 配置发送 TXON 命令，同时将发射数据送入直通数据端口，数据会立即发送出去。
4. 发送 Sleep / IDLE 来完成发射，节省功耗。

5.6.2 数据包 (Packet)

数据包 (Packet) 中的 Payload 数据都是从 FIFO 读写, RF 内部实现了两个 128 字节的 FIFO, 可分别作为发射和接收单独使用, 也可连接成为 256 字节的 FIFO 作为发射和接收共用。在数据包模式下有四种控制方式可选。在四种模式下, Preamble 和 Syncword 的配置都起作用, Preamble 最大可配置 256 个字节的长度, Syncword 可配置 0~8 个字节的长度。

数据包模式还可以对数据进行 Manchester、FEC、Whiten 编解码以及 CRC 校验。CRC 可配置 8、16、24、32 位四种长度校验, 且 CRC 多项式任意可配置。Syncword 和 Payload 可配置高位优先或低位优先。CRC 和 Syncword 数据的 Manchester 编码可单独开关。

表 5-15: Mode 相关寄存器列表

寄存器地址	位数	R/W	比特名	功能说明
0x0E	1:0	R/W	Packet_mode	数据包控制模式: 00: 模式 0, 循环发射 TX FIFO 中的数据, 需要写命令退出接收或发射状态 01: 模式 1, 由寄存器控制数据包长度 10: 模式 2, TX FIFO 中的第 1 个或 2 个字节作为包长度 11: 模式 3, 含有 length、address、seqnum 等多种数据域的模式

5.6.2.1 数据包模式

数据包支持 4 种数据包模式: Mode0 ~ Mode3

- **MODE 0**

MODE 0 为 FIFO 循环模式。在此模式下启动 TX/RX, RF 如果没有收到停止命令, TX 端会一直发射 TX FIFO 中的数据, RX 端会将接收到的数据不停地写入到 RX FIFO 中。TX 端在启动发射时会按照帧格式发射, 帧格式有 Preamble、Syncword 和 Data 三个数据域, Data 数据来自 TX FIFO, 且长度不受芯片本身控制。RX 端会检测 Preamble 和 Syncword。

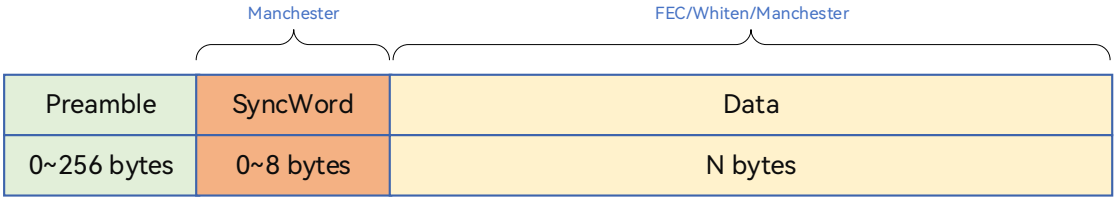


图 5-6: Mode 0 帧格式

● **MODE 1**

MODE 1 帧格式包含 Preamble、Syncword、Data 和 CRC 域。发射 Data 数据来自 TX FIFO，数据长度由 Payload_len[15:0]寄存器控制。RX 端接收 Data 的长度同样由 Payload_len[15:0]寄存器控制。

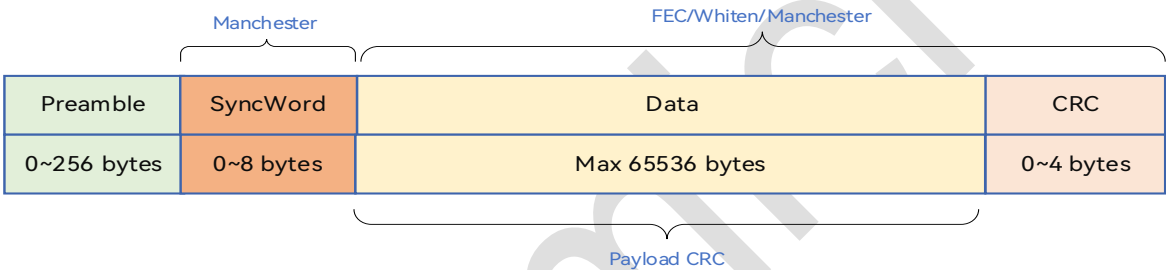


图 5-7: Mode 1 帧格式

● **MODE 2**

MODE 2 帧格式包含 Preamble、Syncword、Length、Data 和 CRC 域。TX 时，Length 和 Data 都来自于 TX FIFO，Length 为 TX FIFO 的前 1 个字或前 2 个字节，Length 为两个字节时高低字节发射顺序可互换。在 RX 端，Length 为接收到的数据的前 1 个字或前 2 个字节，Length 为两个字节时高低字节顺序可互换，根据接收到的 Length 域数据去控制 Data 域的长度。

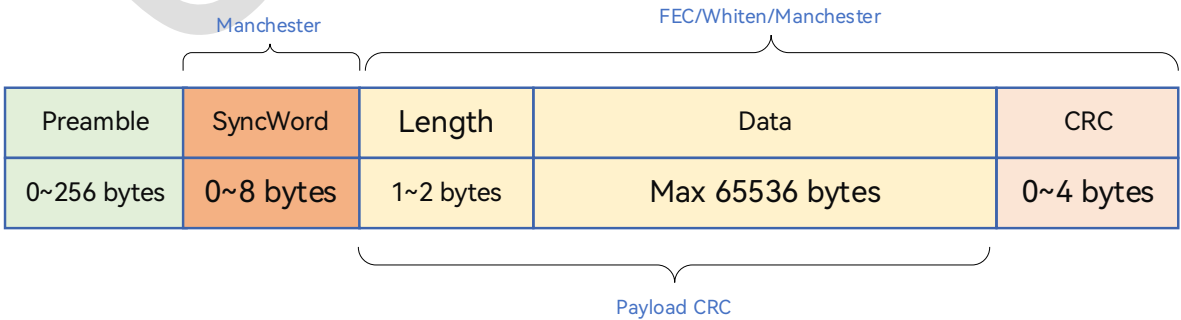


图 5-8: Mode 2 帧格式

● **MODE 3**

MODE 3 帧格式包含 Preamble、Syncword、Length、Address、SeqNum、FCS2、Data 和 CRC 域。Length、Address、SeqNum、FCS2 作为 payload 域并且可以单独使能。Address 域可选择放置在 Length 之前。Length 数据来自 Payload_len[15:0]寄存器，且 Payload 长度范围包括 Length 到 CRC 之前的所有数据域。SeqNum 为数据包计数器，可由寄存器设置也可设置为自动增加。如果 FCS2[7]位使能 ACK 请求响应，当接收端接收到 ACK 请求时，会自动发出 ACK 数据包响应发射端。如果 Length 域没有使能，则接收端根据寄存器 Payload_len 来接收 Payload 数据。

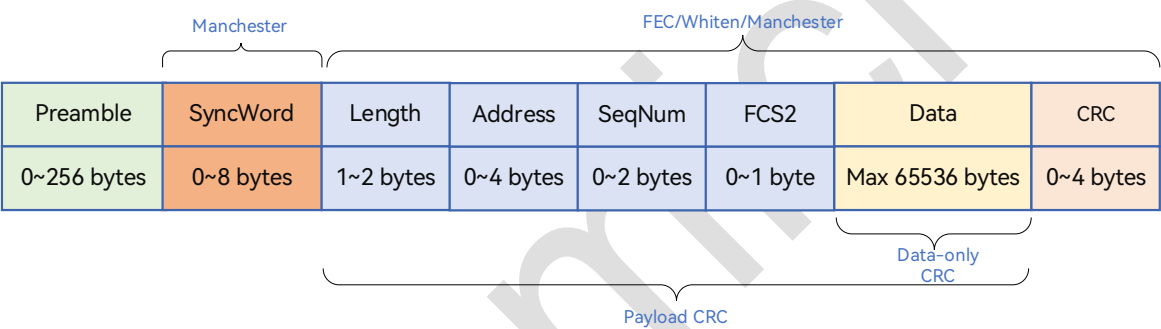


图 5-9: Mode 3 帧格式

5.6.2.2 数据包操作流程

● **RX 数据包接收**

在包模式中，从解调器输出的数据会先被移送至包处理机中进行解码，然后填入 FIFO。包处理机提供多种解码引擎和判断数据有效性的选项，这些可以减少用户的操作。典型的包模式的 RX 工作顺序如下：

1. 初始化寄存器。
2. 配置对应的接收模式。
3. 发送RX命令进入接收状态。
4. 根据相关的中断状态读取RX FIFO。
5. 自动进入IDLE状态以节省功耗。

6. 清除接收中断状态。

● TX 数据包发射

在包模式中，可以提前将数据在 IDLE 状态和 FSON 的状态下填入 FIFO 中，或者在芯片发送数据的同时写入 FIFO，或者以上两种方法的结合。典型的包模式的 TX 工作顺序如下：

1. 初始化寄存器。
2. 发送IDLE状态命令，并将数据填入TX FIFO。
3. 寄存器设置发送中断。
4. 配置TX命令进入发送状态。
5. 在相应的中断状态中将数据写入TX FIFO。
6. 完成N个数据包发送后，会自动退出到IDLE状态，无需发送命令。

5.6.2.3 数据包字段

● Preamble

Preamble 是发射数据包最前端固定的引导码，主要是用于接收端的信号检测和同步。而 Preamble 的长短直接影响链路建立速度、稳定性和能效，合理的 Preamble 长度可显著提升通信的质量，特别是在复杂环境下。对 TX 来说，Preamble 支持独立的使能，并且可控制发射的长度。对于 RX 来说，Preamble 检测成功会有 PREAM_OK 中断产生。用户可在有需要的时候去检测此中断。

表 5-16: Preamble 配置相关寄存器列表

寄存器地址	位数	R/W	比特名	功能说明
0x0C	7:0	R/W	tx_preamble_len	发射 Preamble 长度，单位字节
0x0D	4	R/W	preamble_en	设置 preamble 前导码使能： 0：禁止 1：使能
0x10	2	R/W	preamble_int_en	接收 Preamble（前导码）匹配中断使能： 0：禁止 1：使能
0x37	7:0	R/W	csrst_len	在补偿功能使能时，找到信号的频谱后连续收到 Reg37*4 个前导后，则认

寄存器地址	位数	R/W	比特名	功能说明
				为找到有效的 preamble
0x61	6	R	preamble_rec	接收到有效的 Preamble
0x73	7:0	R/W	preamble_val	发射前导码时的使用的数据

● SyncWord

SyncWord 是数据包通信位于 Preamble 和 Payload 之间的关键标识，直接影响通信的可靠性和安全性，RF 提供灵活的同步字配置选项，可根据应用场景在识别成功率、抗干扰能力和功耗之间取得平衡，且 RF 支持 SyncWord 的独立使能，支持曼彻斯特编码，支持 bit 顺序交换，在接收时支持同步字中断。

表 5-17: Sync 配置相关寄存器列表

寄存器地址	位数	R/W	比特名	功能说明
0x0D	5	R/W	syncword_en	Syncword 同步字使能： 0: 禁止 1: 使能
0x10	1	R/W	syncword_int_en	接收 Syncword(同步字)匹配中断使能： 0: 禁止 1: 使能
0x61	7	R	syncword_rec	接收到正确的 Syncword
0x72	4:0	R/W	sync_thres	同步字检测允许的错误位数
0x74	4	R/W	sync_man_en	同步字曼彻斯特编码使能： 0: 禁止 1: 使能
	3	R/W	sync_bit_order	同步字 bit 顺序： 0: 字节低位 LSB 1: 字节高位 MSB
	2:0	R/W	sync_len	同步字长度： 000: {sync_id_1} 001: {sync_id_1, sync_id_2} 010: {sync_id_1, sync_id_2, sync_id_3} 011: {sync_id_1, sync_id_2, sync_id_3, sync_id_4} 100: {sync_id_1, sync_id_2, sync_id_3, sync_id_4, sync_id_5}

寄存器地址	位数	R/W	比特名	功能说明
				101: {sync_id_1, sync_id_2, sync_id_3, sync_id_4, sync_id_5, sync_id_6} 110: {sync_id_1, sync_id_2, sync_id_3, sync_id_4, sync_id_5, sync_id_6, sync_id_7} 111: {sync_id_1, sync_id_2, sync_id_3, sync_id_4, sync_id_5, sync_id_6, sync_id_7, sync_id_8}
0x75	7:0	R/W	sync_id_1	同步字字节 1
0x76	7:0	R/W	sync_id_2	同步字字节 2
0x77	7:0	R/W	sync_id_3	同步字字节 3
0x78	7:0	R/W	sync_id_4	同步字字节 4
0x79	7:0	R/W	sync_id_5	同步字字节 5
0x7A	7:0	R/W	sync_id_6	同步字字节 6
0x7B	7:0	R/W	sync_id_7	同步字字节 7
0x7C	7:0	R/W	sync_id_8	同步字字节 8

● Payload

Payload 是数据包通信中真正承载用户信息的核心部分, RF 支持灵活配置 Payload Data 编码, 支持 NRZ、曼彻斯特、数据白化、FEC 等编码, 并支持高低位互换。

在不同模式下, Payload Data 长度设置方式不一样, 以下为各个模式设置。

- MODE 0: Payload Data 的长度是由 FIFO 长度决定, 当 FIFO 独立使用时, Payload Data 的长度为 128Bytes, 当 FIFO 共享使用时, Payload Data 的长度为 256Bytes。
- MODE 1: Payload Data 的长度由寄存器 payload_length 决定。
- MODE 2: Payload Data 的长度由 FIFO 第一个字节或者前 2 个字节决定。
- MODE 3: 当 length 长度使能时, data 域长度由 length 域决定, length 域由寄存器 payload_length 组成。当 length 不使能时, data 域长度由寄存器 payload_length 决定。

表 5-18: Payload 相关寄存器

寄存器地址	位数	R/W	比特名	功能说明
0x0D	7	R/W	payload_bit_order	Payload bit 顺序: 0: LSB 低位在前 1: MSB 高位在前
	6	R/W	manchester_inv	曼彻斯特编码: 0: 上升沿为编码 1, 下降沿为编码 0 1: 下降沿为编码 1, 上升沿为编码 0
	3:2	R/W	pkt_enc_type	数据包编码: 00: NRZ 01: 曼彻斯特编码 10: 无效 11: 交织编码
	1:0	R/W	fec_type	数据包 FEC 类型: 00: NO FEC 01: 1/3 FEC 10: 2/3 FEC 11: 1/2 FEC
0x0E	7	R/W	length_byte_swap	包长度控制域为两个字节时, 高低字节顺序: 0: 低字节在前 1: 高字节在前
	6	R/W	length_sel	包长度控制域字节配置: 0: 1 字节 1: 2 字节
	4	R/W	scramble_en	数据白化(扰码)使能: 0: 禁止 1: 使能
	3	R/W	fifo_share_en	FIFO 共享设置: 0: RX 和 TX 各 128 字节 FIFO 1: RX 和 TX 共用 256 字节 FIFO
0x70	7:0	R/W	scramble_init_val[7:0]	scramble 初始化值
0x71	6	R/W	srcamble_len	伪随机数长度选择: 0: PN9 1: PN7
	5	R/W	scramble_msb	选择首先输出的移位寄存器: 0: LSB 1: MSB
	4	R/W	scramble_type	伪随机数产生器类型:

寄存器地址	位数	R/W	比特名	功能说明
				0: 抽出模式 1: 插入模式
	3:1	R/W	scramble_poly	反馈抽头选择
	0	R/W	scramble_init_val[8]	scramble 初始化值最高位
0x7D	7:0	R/W	payload_len[7:0]	payload 长度, 在数据包模式 1 和数据包模式 3 有效
0x7E	7:0	R/W	payload_len[15:8]	payload 长度, 在数据包模式 1 和数据包模式 3 有效

● Payload 数据 bit 交换功能

Payload 数据交换功能时指发射的数据的高低位顺序, Payload_bit_order = 0 表示发送时把 Payload 的每个 byte 自身从 LSB 到 MSB 的顺序发送或进行 Manchester/Whiten 编码。Payload_bit_order = 1 表示把 Payload 的每个 byte 自身从 MSB 到 LSB 的顺序发送或进行 Manchester/Whiten 编码。如下图, Payload 数据从 MSB 转换成 LSB。

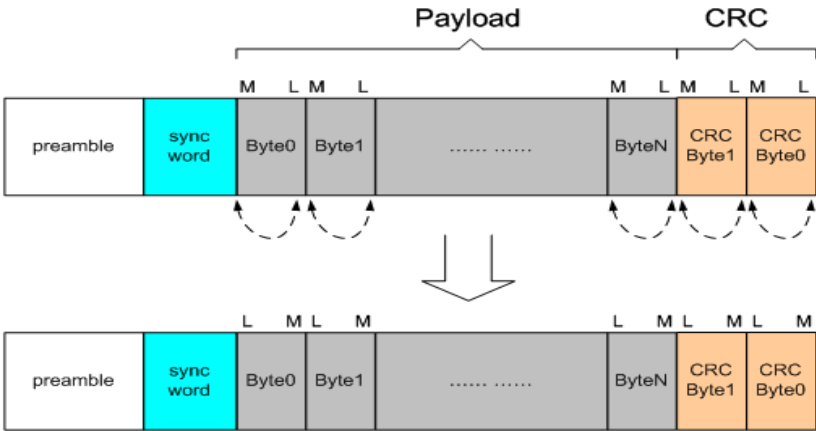


图 5-10: PAYLOAD_BIT_ORDER 操作

● Payload 数据白化功能

数据白化随机数产生器有 PN7 或 PN9 可选择, 结构上也有抽出或插入模式; 数据输出可选最低位或最高位。

➤ PN7 的抽出模式

如图 5-11 所示, 移位顺序不变, 从高位移位到低位:

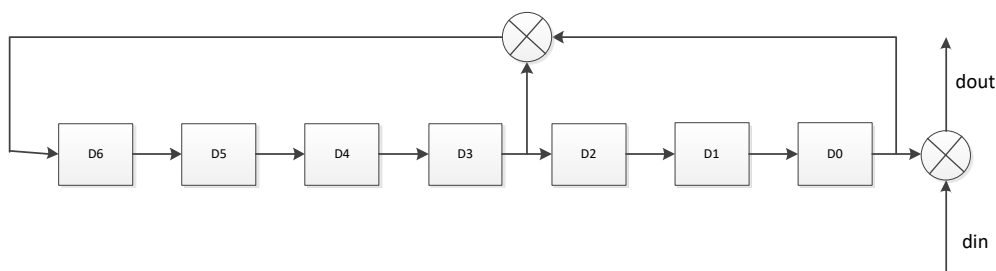


图 5-11: PN7 抽出模式移位顺序

其对应配置为:

- ✓ Reg0E[4] = 1, 打开白化使能
- ✓ Reg71[6] = 1, 选择 PN7
- ✓ Reg71[5] = 0, 选择 LSB 和数据做异或
- ✓ Reg71[4] = 0, 选择抽出模式
- ✓ Reg71[3:1] = 'b010, 选择在 D2 输入处抽出, 即 Reg71[3:1]+1 的寄存器输出处, 例如举例为 2, 则在 D3 输出处抽出或者插入

➤ PN9 的插入模式

如图 5-12 所示, 移位顺序不变, 从高位移位到低位:

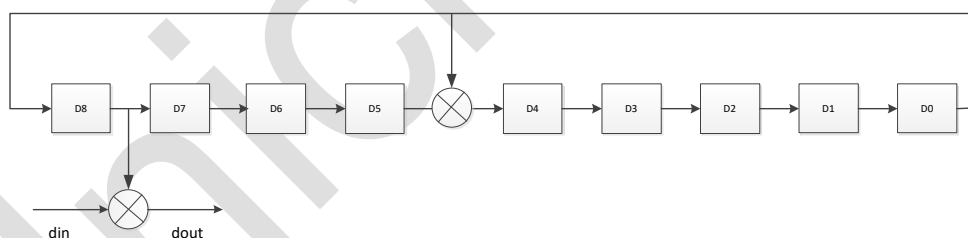


图 5-12: PN9 插入模式移位顺序

其对应配置为:

- ✓ Reg0E[4] = 1 打开白化使能
- ✓ Reg71[6] = 0 选择 PN9
- ✓ Reg71[5] = 1 选择 MSB 和数据做异或
- ✓ Reg71[4] = 1 选择插入模式
- ✓ Reg71[3:1] = 'b100 选择在 D4 输入处插入, 即 Reg71[3:1]+1 的寄存器输出处, 例如举例为 4, 则在 D5 输出处抽出或者插入

● Length

Length 域仅在 Mode 3 下支持, 是决定发射数据和接收数据的长度。Length 域支持独立使能, 并支持灵活设置。当 Length 使能时, Length 域决定 Mode 3 的 Data 域长度。当 Length 域不使能时, Mode 3 的 Data 域长度由寄存器 payload_length 决定。

表 5-19: Length 相关的寄存器列表

寄存器地址	位数	R/W	比特名	功能说明
0x0E	7	R/W	length_byte_swap	包长度控制域为两个字节时, 高低字节顺序交换: 0: 低字节在前 1: 高字节在前
	6	R/W	Length_sel	选择 1 或 2 个字节作为包长度控制域: 0: 1 字节 1: 2 字节
0x6A	7:0	R	m3_rx_payload_len[7:0]	数据包模式 3 时, 接收 length 数据
0x6B	7:0	R	m3_rx_payload_len[15:8]	数据包模式 3 时, 接收 length 数据
0x80	5	R/W	m3_length_en	数据包模式 3 时, 数据长度域控制: 0: 禁止 1: 使能
0x7D	7:0	R/W	payload_len[7:0]	payload 长度, 在数据包模式 1 和数据包模式 3 有效
0x7E	7:0	R/W	payload_len[15:8]	payload 长度, 在数据包模式 1 和数据包模式 3 有效
0x81	1	R/W	m3_addr_pos_sel	数据包模式 3 时, 地址位置选择: 0: address 放在 length 之后 1: address 放在 length 之前

在 MODE 3 中, 根据 Length 使能或者不使能来分为两种格式: 固定包格式和可变包格式, 对于 TX 来说, length 都来自寄存器, 对于 RX 来说, 如果 length 域不使能, 则来自寄存器, 如果 length 域使能, 则来自数据包格式的 length。

➤ 固定包格式

在 MODE 3 中, Length 域不使能(Reg80[5]=0, m3_length_en=0)的情况下, 对于 TX 和 RX 来说, Payload 的长度来自寄存器{Reg7D,Reg7E}, Payload 的实际长度为配置的寄存器

Payload_len 加一。 Payload 结构 = Address + Seqnum + FCS2 + Data，其中 Address / Seqnum / FCS2 单独配置使能，其结构如下：

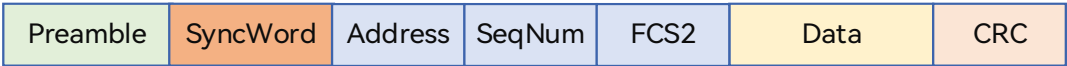


图 5-13: m3_length_en=0 时 MODE 3 帧格式

示例一：

Payload_len<15:0> = 17

Addr_en = 1

Addr_size <1:0> = 2

Seqnum_en = 1

Seqnum_size = 1

Fcs2_en = 1

当 Payload 包含 Address + Seqnum + FCS2 + Data 共 4 部分，Payload 的总长度是 17 + 1 = 18，Address 的长度是 2（Addr）+ 1 = 3，Sequence Num 的长度是 1 + 1 = 2，FCS2 长度固定为 1，所以，Data 的长度是 18 -3 -2 -1 = 12 个 bytes。

示例二：

Payload_len <15:0> = 17

Addr_en = 1

Addr_size<1:0> = 2

Seqnum_EN = 0

Seqnum_size = 1

Fcs2_en = 0

当 Address 、Seqnum 、FCS2 不使能时，Payload 由 Data 组成，那么 Data 的长度就是 18。

➤ 可变包格式

在 MODE 3 中, Length 域使能(Reg80[5]=1, m3_length_en=1)的情况下, 对于 TX 来时, payload length 来自寄存器{Reg7D,Reg7E}, 对于 RX 来说, payload length 来自数据包的 Length 域, 根据 Length 域的位置不同及 Address 域使能来决定 payload length 的计算。

✓ Address 不使能 (m3_addr_en=0)

当 Address 不使能的情况下, Payload 结构 = Length + Seqnum + FCS2 + Data, 其中 Seqnum、FCS2 可以独立配置使能, payload Length 包括 length+Seqnum_FCS2+Data。

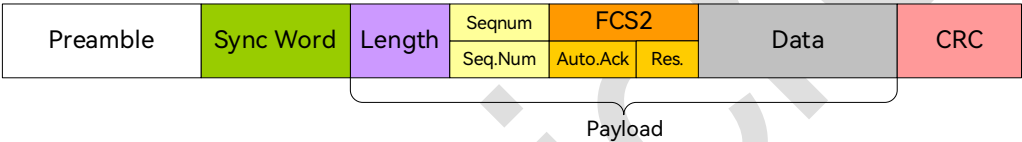


图 5-14: 可变长度包格式, Address 不存在

当 Length 域使能, Address 不使能 (m3_length_en=1, m3_addr_en=0), 整个 payload 的长度, 包括 length+Seqnum_FCS2+Data, 对于 TX 来说, Length 的内容由 Payload_len <15:0>寄存器配置。如果 Length_size = 0, 即 Payload_len <7:0>有效, 最大值为 255, Length Byte 本身是 1 个 byte 的长度。如果 Length_size = 1, 即 Payload_len<15:0>有效, 最大值为 65535, Length Byte 本身是 2 个 byte 的长度, 对于 RX 来说, RX 需要解析 length 域, length+1 表示 Payload 的长度。

示例如下:

Payload_len <15:0> = 17

Length_size = 0

Seqnum_EN = 1

Seqnum_size = 1

Fcs2_en = 1

当 Seqnum、FCS2 使能时, Payload 包含 Length + Seqnum + FCS2 + Data 共 4 部分, Payload 的总长度是 17 + 1 = 18, Length 的长度是 1, Sequence Num 的长度是 1

+ 1 = 2, FCS2 长度固定为 1, 所以, Data 的长度是 18 - 1 - 2 - 1 = 14 个 bytes。

如果 Seqnum + FCS2 不使能时, 那么 Data 的长度就是 18 - 1 = 17 个 bytes。

如果 length_size = 1, 即 Length 的长度是 2, 所以, Data 的长度是 18 - 2 - 2 - 1 = 13 个 bytes。

如果 Seqnum + FCS2 不存在, 那么 Data 的长度就是 18 - 2 = 16 个 bytes。

✓ **Address 存在, 且 Addr_pos_sel = 1 (Address 在 Length Byte 之前)**

ADDR 放在 LENGTH 之前, ADDR 单独罗列出来, Payload 结构 = Address + Length Byte + Seqnum + FCS2 + Data, Payload length 的实际长度为配置的寄存器 Payload_len 加一。

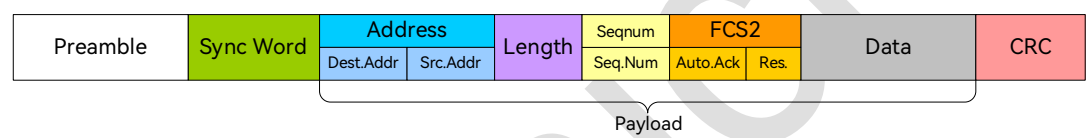


图 5-15: Address 存在, Address 在 Length Byte 之前

当 Address 存在, 且 Addr_pos_sel = 1 (Address 在 Length Byte 之前) 时, 对于 TX 来说, Payload 长度 = (addr_size<1:0> + 1) + 1 + payload_length<15:0>, 其中 Address 的长度根据 addr_size<1:0>确定, payload_length<15:0>的内容就等于 Length Byte 的内容, payload_length+1 指的是除 Address 外的 Payload 的长度。如果 length_size = 0, 即 payload_length<7:0>有效, 最大值为 255, Length Byte 本身是 1 个 byte 的长度。如果 length_size = 1, 即 payload_length<15:0>有效, 最大值为 65535, Length Byte 本身是 2 个 bytes 的长度。对于 RX 来说, RX 需要解析 length 域, length+1 表示除 Address 外的 Payload 的长度。

示例如下:

Payload_length<15:0> = 17

Addr_en= 1

Addr_size<1:0> = 2

Length_size = 0

Seqnum_EN = 1

Seqnum_size = 1

FCS2_EN = 1

当 Seqnum、 FCS2 都使能时, Payload 包含 Address + (Length +Seqnum + FCS2 + Data)共 5 部分, 其中, Address 的长度是 2 (Dest.Addr) + 1 = 3, Length 的长度是 1, Sequence Num 的长度是 1 + 1 = 2, FCS2 长度固定为 1, 那么 Payload 的总长度是 3 + (17 + 1) = 21, 所以, Data 的长度是 18 - 1 - 2 - 1 = 14 个 bytes。

如果 Seqnum + FCS2 不使能, 那么 Data 的长度就是 18 - 1 = 17 个 bytes。

如果 length_size = 1, 即 Length 的长度是 2, 所以, Data 的长度是 18 - 2 - 2 - 1 = 13 个 bytes。

如果 Seqnum + FCS2 不使能, 那么 Data 的长度就是 18 - 2 = 16 个 bytes。

✓ **Address 存在, 且 Addr_pos_sel = 0 (Address 在 Length Byte 之后)**

ADDR放在LENGTH后, 同MODE 1, Payload结构 = Length Byte + Address + Seqnum + FCS2 + Data, Real payload length的实际长度为配置的寄存器Payload_len加一。

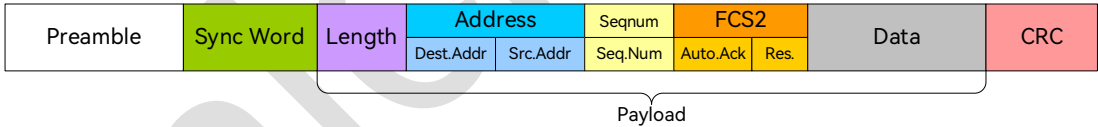


图 5-16: Address 存在, Address 在 Length Byte 之后

当 Address 存在, 且 Addr_pos_sel = 0 (Address 在 Length Byte 之后) 时, 对于 TX 来说, Payload 长度: 1 + payload_length<15:0>, 其中 payload_length<15:0>的内容就等于 Length Byte 的内容, 指的是 Payload 的长度。如果 length_size = 0, 即 payload_length<7:0>有效, 最大值为 255, Length Byte 本身是 1 个 byte 的长度。如果 length_size = 1, 即 payload_length<15:0>有效, 最大值为 65535, Length Byte 本身是 2 个 bytes 的长度。对于 RX 来说, RX 需要要解析 length 域, length+1 表示 Payload 的长度。

示例如下:

Payload_length<15:0> = 17

Addr_en = 1

Addr_size<1:0> = 2

Length_size = 0

Seq_en = 1

Seqnum_size = 1

FCS2_EN = 1

当 Seqnum、FCS2 使能时，Payload 包含 Length + Address + Seqnum + FCS2 + Data 共 5 部分，Payload 的总长度是 17 + 1 = 18，Length 的长度是 1，Address 的长度是 2 (Addr) + 1 = 3，Sequence Num 的长度是 1 + 1 = 2，FCS2 长度固定为 1，所以，Data 的长度是 18 - 1 - 3 - 2 - 1 = 11 个 bytes。

如果 Seqnum + FCS2 不使能，那么 Data 的长度就是 18 - 1 - 3 = 14 个 bytes。

如果 length_size = 1，即 Length 的长度是 2，所以，Data 的长度是 18 - 2 - 3 - 2 - 1 = 10 个 bytes。

如果 Seqnum + FCS2 不使能，那么 Data 的长度就是 18 - 2 - 3 = 13 个 bytes。

● Address

Address 域仅在 Mode 3 下支持，Address 域支持独立使能，并支持灵活设置、地址过滤等。

表 5-20: Address 配置相关寄存器列表

寄存器地址	位数	R/W	比特名	功能说明
0x68	0	R	m3_addr_comp_ok	数据包模式 3 时，接收 address 匹配正确
0x81	5:4	R/W	m3_addr_size	数据包模式 3 时，地址长度： 00: 1 字节 01: 2 字节 10: 3 字节 11: 4 字节
	2	R/W	m3_addr_split_mode	数据包模式 3 时，将地址分为源地址和目的地址，在返回 ACK 时，将收到的源地址和目的地址位置互换：

寄存器地址	位数	R/W	比特名	功能说明
				0: 不互换 1: 互换
	0	R/W	m3_addr_en	数据包模式 3 时, 地址域控制: 0: 禁止 1: 使能
0x82	7:0	R/W	m3_addr_val[7:0]	数据包模式 3 时, 地址值
0x83	7:0	R/W	m3_addr_val[15:8]	数据包模式 3 时, 地址值
0x84	7:0	R/W	m3_addr_val[23:16]	数据包模式 3 时, 地址值
0x85	7:0	R/W	m3_addr_val[31:24]	数据包模式 3 时, 地址值
0x86	7:0	R/W	m3_addr_bit_mask[7:0]	数据包模式 3 时, 地址过滤, mask 为“1”时对应地址位不作比较
0x87	7:0	R/W	m3_addr_bit_mask[15:8]	数据包模式 3 时, 地址过滤, mask 为“1”时对应地址位不作比较
0x88	7:0	R/W	m3_addr_bit_mask[23:16]	数据包模式 3 时, 地址过滤, mask 为“1”时对应地址位不作比较
0x89	7:0	R/W	m3_addr_bit_mask[31:24]	数据包模式 3 时, 地址过滤, mask 为“1”时对应地址位不作比较

● seqnum

Seqnum 域仅在 Mode 3 下支持, 支持独立使能, Seqnum 的数据来源可来自寄存器或者内部计数器。在 TX 模式, Seqnum 作为数据包结构一并发出。在接收中, Seqnum 中自动填入到映射的寄存器, 可以直接读取寄存器值。

表 5-21: seqnum 相关的寄存器列表

寄存器地址	位数	R/W	比特名	功能说明
0x68	1	R	m3_seqnum_ok	数据包模式 3 时, 接收 seqnum 匹配正确
0x8A	5	R/W	m3_seqnum_auto_inc	数据包模式 3 时, seqnum 配置方式: 0: seqnum 数据由{Reg8c,Reg8b}配置 1: 由内部 seqnum 计数器自动加 1
	4	R/W	m3_seqnum_en	数据包模式 3 时, seqnum 数据使能: 0: 禁止 1: 使能
	3	R/W	m3_seqnum_bit_size	数据包模式 3 时, seqnum 数据长度: 0: 8bits

寄存器地址	位数	R/W	比特名	功能说明
				1: 16bits
0x8B	7:0	R/W	seqnum_Reg[7:0]	数据包模式 3 时, seqnum 值
0x8C	7:0	R/W	seqnum_Reg[15:8]	数据包模式 3 时, seqnum 值
0xE0	7:0	R	rx_seqnum_val[7:0]	数据包模式 3 时, 接收 seqnum 数据
0xE1	7:0	R	rx_seqnum_val[15:8]	数据包模式 3 时, 接收 seqnum 数据
0xE2	7:0	R	seqnum_cnt[7:0]	seqnum_cnt
0xE3	7:0	R	seqnum_cnt[15:8]	seqnum_cnt

● FCS2

FCS2 域仅在 Mode 3 下支持, FCS2 支持独立使能、灵活配置。

表 5-22: FCS2 相关的寄存器列表

寄存器地址	位数	R/W	比特名	功能说明
0x10	7:4	R/W	retx_times	Auto ACK 使能时, 发射端未收到 ACK 包时自动重发的次数
0x69	7:0	R	m3_rx_fcs2_value	数据包模式 3 时, 接收 FCS2 数据
0x74	7	R/W	auto_ack	auto_ack 使能: 0: 禁止 1: 使能
	6	R/W	ack_payload_en	发射 ACK 时带 payload 数据返回: 0: 禁止 1: 使能
0x80	3:0	R/W	m3_ack_node_ctrl	数据包模式 3 且 AUTO_ACK 功能开启, 返回的 ACK 包控制: [3]: 为“1”时使能返回接收到的数据长度 [2]: 为“1”时使能返回接收到的地址 [1]: 为“1”时使能返回接收到的 seqnum [0]: 为“1”时使能返回接收到的 FCS2 数据
0x8D	0	R/W	m3_fcs2_en	数据包模式 3 时, FCS2 数据域使能: 0: 禁止 1: 使能
0x8E	7	R/W	m3_fcs2_value	数据包模式 3 时, ACK request
0xA3	5:0	R/W	auto_ack_wait_time	auto ack 功能开启时, 发射端发完数据包后等待 ACK 数据的时间, 超时会重新发射数据

● **AUTO ACK**

接收端具有自动应答功能，在收到一个数据包后，如果该包要求回复 ACK 包（auto_ack 打开，在 Mode3 时 m3_fcs2_value[7]需要为 1），则芯片直接自动切换到发射模式并发回 ACK 包。ACK 的包格式其中一种表示：



图 5-17: ACK 的包格式

ACK 数据包包含 Preamble 和 Sync Word，其他域需要 ack_m3_mode_ctrl (Reg80[3:0]) 和 ack_payload_en(Reg74[6])来决定是否打开，其中，Address 域将接收到的 DEST ADDR 和 SRC ADDR 位置交换，如有 seq 域，将接收到的 Sequence Num 直接返回。

对于 Master 端，如果要求 Slave 发送应答包，模式 1 和 2 直接打开 auto_ack，而模式 3 需要将 m3_fcs2_value 设置为 1，发送完当前包后自动进入接收状态，接收 Slave 发回的 ACK 包。如果发射端超时未收到 ACK 包则重新发射数据包。

● **CRC**

CRC 域在 Mode 1/2/3 下支持，CRC 域作为数据校验，确保数据完整性、可靠性的关键技术，RF 的 CRC 域支持独立使能，灵活配置，支持常用的标准多项式。

表 5-23: CRC 相关的寄存器列表

寄存器地址	位数	R/W	比特名	功能说明
0x0E	5	R/W	crc_en	CRC 使能： 0：禁止 1：使能
0x57	1	R/W	crc_err_st	接收 CRC 错误时，数据包的处理方法： 0：正常完成该包的接收，退出接收状态发出数据包中断和错误 CRC 标志 1：丢弃该包数据并重新进入接收状态
0x61	5	R	crc_error	高电平时表示 CRC 校验错误
0x80	4	R/W	m3_crc_sel	数据包模式 3 时，CRC 的校验范围： 0：All payload 1：仅 FIFO 中的数据
0x94	4	R/W	crc_bit_order	CRC bit 顺序：

寄存器地址	位数	R/W	比特名	功能说明
				0: LSB 1: MSB
	3	R/W	crc_inv	CRC bit 取反: 0: 禁止 1: 使能
	2	R/W	crc_man_en	CRC 曼彻斯特编码使能: 0: 禁止 1: 使能
	1:0	R/W	crc_len	CRC 长度: 00: 8 位 01: 16 位 10: 24 位 11: 32 位
0x95	7:0	R/W	crc_poly[7:0]	CRC 多项式配置
0x96	7:0	R/W	crc_poly[15:8]	CRC 多项式配置
0x97	7:0	R/W	crc_poly[23:16]	CRC 多项式配置
0x98	7:0	R/W	crc_poly[31:24]	CRC 多项式配置
0x99	7:0	R/W	crc_init_val[7:0]	CRC 移位寄存器初始化值
0x9A	7:0	R/W	crc_init_val[15:8]	CRC 移位寄存器初始化值
0x9B	7:0	R/W	crc_init_val[23:16]	CRC 移位寄存器初始化值
0x9C	7:0	R/W	crc_init_val[31:24]	CRC 移位寄存器初始化值

1. CRC 配置说明

➤ M3_crc_sel (Reg80[4])

M3_crc_sel 用来指定在包模式 3 时 CRC 的编解码校验对象。

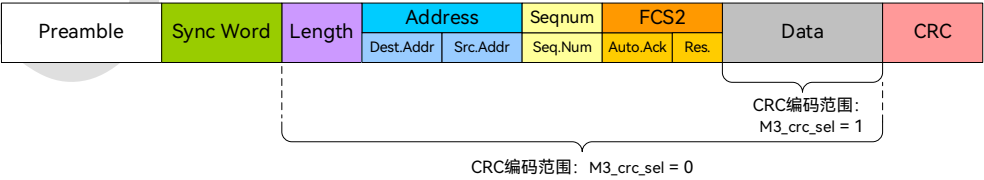


图 5-18: CRC 编码范围

➤ crc_inv (Reg94[3])

crc_inv 是将 CRC 的每一位都进行逻辑取反，由 0 变为 1，由 1 变为 0。

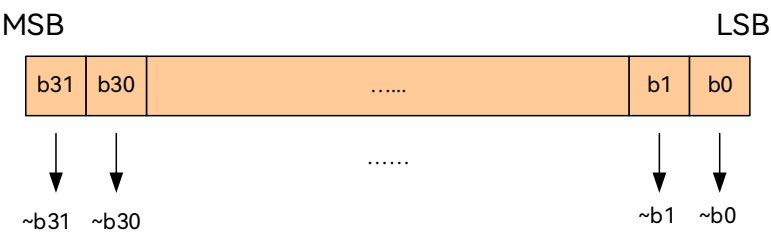


图 5-19: CRC_INV

➤ **crc_bit_order(Reg94[4])**

crc_bit_order 是将 CRC 的 b31-b0 开始送出或者从 b0- b31 发送选项。

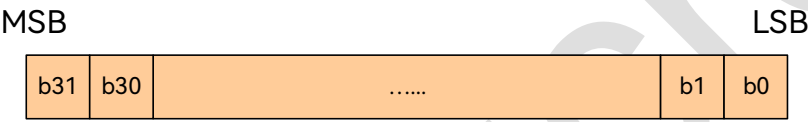


图 5-20: CRC_BIT_ORDER

2. 常用 CRC 标准和多项式

表 5-24: 常用 CRC 标准和多项式列表

名称	生成多项式	简记式	标准引用
CRC-4	x^4+x+1	0x3	ITU G.704
CRC-8	x^8+x^2+x+1	0x07	-
CRC-8/MAXIM	$x^8+x^5+x^4+1$	0x31	-
CRC-8/SAE-J1850	$x^8+x^4+x^3+x^2+1$	0x1D	-
CRC-12	$x^{12}+x^{11}+x^3+x^2+x+1$	0x80F	-
CRC-16	$x^{16}+x^{15}+x^2+1$	0x8005	IBM SDLC
CRC-CCITT	$x^{16}+x^{12}+x^5+1$	0x1021	ISO HDLC、ITU X.25 V.34/V.41/V.42、PPP-FCS
CRC-32	$X^{32}+x^{26}+x^{23}+...x^2+x+1$	0x04C11DB7	ZIP、RAR、IEEE 802 LAN / FDDI、 IEEE1394、PPP-FCS
CRC-32c	$X^{32}+x^{28}+x^{27}+...x^8+x^6+1$	0x1EDC6F41	SCTP

3. 配置 CRC8 和 CRC16 示例

➤ CRC8 的 $X^8+X^2+X^1+1$ 配置示例（简记式为 0x07）

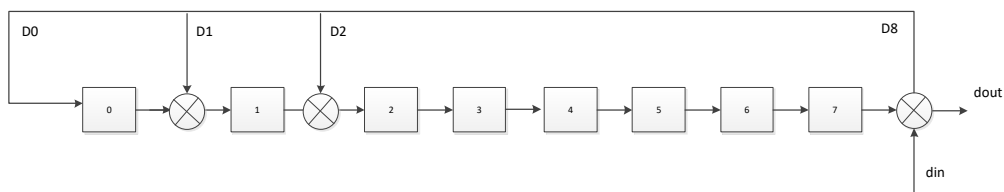


图 5-21: CRC8 的配置示例

如图 5-21 所示，对应配置为：

- ✓ Reg94[1:0] = 0x01; crc_len 设置 8bits
- ✓ Reg95 = 0x07
- ✓ Reg96 = 0x00
- ✓ Reg97 = 0x00
- ✓ Reg98 = 0x00

Reg99、Reg9A、Reg9B、Reg9C 根据实际情况配置 CRC 的初始值。

➤ CRC16 的 $X^{16}+X^{12}+X^5+1$ 的配置示例（简记式为 0x1021）

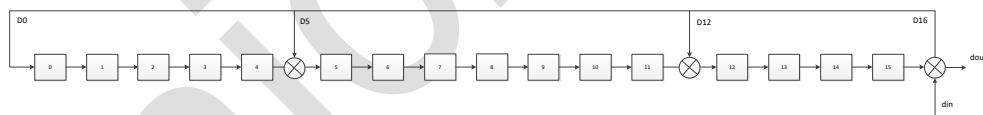


图 5-22: CRC16 的配置示例

如图 5-22 所示，对应配置为：

- ✓ Reg94[1:0] = 0x02; crc_len 设置 16bits
- ✓ Reg95 = 0x21
- ✓ Reg96 = 0x10
- ✓ Reg97 = 0x00
- ✓ Reg98 = 0x00

Reg99、Reg9A、Reg9B、Reg9C 根据实际情况配置 CRC 的初始值。

6 EFC

6.1 概述

芯片上集成了64Kbyte的eFlash存储器，用于保存芯片所有的关键脱机信息和数据。EFC为eFlash控制器，在CPU的配置下，完成Flash读、写、擦除等操作。

6.2 主要特性

- 支持 eFlash 的读写（8/16/32bit）、sector 擦除和 chip 擦除等操作流程。
- 读等待时间可以配置。
- 主区有 128 个 sector，每个 512 字节。
- NVR 区有 2 个 sector，每个 512 字节。
- 支持对 NVR 区域擦/写保护功能。
- 支持擦写保护功能。
- 支持自动锁总线功能。
- Sector 擦除时间 5ms（max），Chip 擦除时间 40ms（max），word 写 20μs（max），读时间 25ns（max）。

6.3 寄存器描述

寄存器基地址：0x01100000

表 6-1：EFC 寄存器列表

偏置	名称	描述
0x00	EFC_CTRL	控制寄存器
0x04	EFC_SEC	写擦除操作安全寄存器
0x08	EFC_STATUS	状态寄存器
0x0C	EFC_INTSTATUS	中断状态寄存器

偏置	名称	描述
0x10	EFC_INEN	中断使能寄存器
0x14	EFC_HALFUS	时间标尺寄存器
0x20	EFC_RCHTRIM	RCH TRIM寄存器
0x24	EFC_RCLTRIM	RCL TRIM寄存器

6.3.1 控制寄存器 EFC_CTRL (偏移: 0000H)

比特	名称	属性	复位值	描述
31:7	RSV	-	-	保留
6:3	Rd_Wait	R/W	0	读等待时间设置位。其为读操作eFlash端口等待的时钟周期数。eFlash要求读等待时间至少为25ns，以满足eFlash读的最大延迟。 0: eFlash端口等待1个系统时钟周期，总线无效率损失 1: eFlash端口等待2个系统时钟周期，总线Hreadyout信号拉低一个时钟周期
2	Chip_Erase_Mode	R/W	0	Chip Erase Mode模式设置位： 0: Chip Erase Mode模式禁止 1: Chip Erase Mode模式使能
1	Sector_Erase_Mode	R/W	0	Sector Erase Mode模式设置位： 0: Sector Erase Mode模式禁止 1: Sector Erase Mode模式使能
0	Write_Mode	R/W	0	Write模式设置位： 0: 写操作模式禁止 1: 写操作模式使能

6.3.2 写擦安全寄存器 EFC_SEC (偏移: 0004H)

比特	名称	属性	复位值	描述
31:0	Write_Lock_Ser	W	0	向eFlash写数据/擦除前，须向此寄存器内写0x55aaaa55值，否则控制其忽略此次写操作。

6.3.3 状态寄存器 EFC_STATUS (偏移: 0008H)

比特	名称	属性	复位值	描述
31:3	RSV	-	-	保留
2	Nvr2_Lock	R	0	Nvr2区是否锁住: 0: Nvr2区没有锁住 1: Nvr2区已经锁住
1	Nvr1_Lock	R	0/1	Nvr1区是否锁住: 0: Nvr1区没有锁住 1: Nvr1区已经锁住
0	EFlash_Ready	R	1	eFlash状态指示位。该反映eFlash工作的状态: 0: eFlash状态忙 1: eFlash状态空闲

6.3.4 中断状态寄存器 EFC_INTSTATUS (偏移: 000CH)

比特	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4	Wrong_Prog	R/W	0	1: 写或擦除操作有误。当EFC_CTRL寄存器的Chip_Erase_Mode、Sector_Erase_Mode、Write_Mode位有2位或以上为1时并进行擦写操作, 此位均会置1 0: 正常状态 1: 操作有误 写1清0
3	Boot_Err	R/W	0	0: 正常状态 1: Boot区发生操作错误。当对应的Boot区Lock住时, 对此Boot区域进行写擦操作, 或者对Boot区域进行Chip Erase操作时, 此位均会置1 写1清0

比特	名称	属性	复位值	描述
2	Nvr2_Err	R/W	0	0: 正常状态 1: Nvr2区发生操作错误。当对应的Nvr2区Lock住时, 对此Nvr2区域进行写擦操作, 或者对Nvr区域进行Chip Erase操作时, 此位均会置1 写1清0
1	Nvr1_Err	R/W	0	0: 正常状态 1: Nvr1区发生操作错误。当对应的Nvr1区Lock住时, 对此Nvr1区域进行写擦操作, 或者对Nvr区域进行Chip Erase操作时, 此位均会置1 写1清0
0	ErWr_done	R/W	0	写/擦除完成中断状态位: 0: 写/擦除未完成 1: 写/擦除完成, 写1清除该位。如果中断允许, 则产生中断

6.3.5 中断使能寄存器 EFC_INEN (偏移: 0010H)

比特	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4	Wrong_Prog_En	R/W	0	Wrong_Prog中断使能: 0: Wrong_Prog中断不使能 1: Wrong_Prog中断使能
3	Boot_Err_En	R/W	0	Boot_Err中断使能: 0: Boot_Err中断不使能 1: Boot_Err中断使能
2	Nvr2_Err_En	R/W	0	Nvr2_Err中断使能: 0: Nvr2_Err中断不使能 1: Nvr2_Err中断使能
1	Nvr1_Err_En	R/W	0	Nvr1_Err中断使能: 0: Nvr1_Err中断不使能 1: Nvr1_Err中断使能
0	Er_done_En	R/W	0	擦除完成中断使能: 0: 写/擦除中断不使能 1: 写/擦除中断使能

6.3.6 时间标尺寄存器 EFC_HALFUS (偏移: 0014H)

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	Half_Us	R/W	0xF	对eFlash进行擦写前, 需根据系统hclk的时钟频率, 设置此寄存器。公式如下: $0.5\mu s = T_{hclk} * (Half_Us + 1)$ 即hclk时钟的周期, 乘以Half_Us + 1的值, 将等于0.5μs。

6.3.7 RCH TRIM 寄存器 EFC_RCHTRIM (偏移: 0020H)

比特	名称	属性	复位值	描述
31:10	RSV	-	-	保留
9:0	ISEL	R/W	10'h6F	64M RCH TRIM值: ISEL[9:5]: 粗调位, 以5'hF值为中心, 每加减1, RCH_CLKOUT增减3% ISEL[4:0]: 精调位, 以5'h10值为中心, 每加减1, RCH_CLKOUT增减0.27%

6.3.8 RCL TRIM 寄存器 EFC_RCLTRIM (偏移: 0024H)

比特	名称	属性	复位值	描述
31:10	RSV	-	-	保留
9:8	RTRIM		2'h1	32k RCL TRIM值RTIM
7:5	RSV	-	-	保留
4:0	S	R/W	5'hF	32k RCL TRIM值S

6.4 功能描述

6.4.1 地址映射

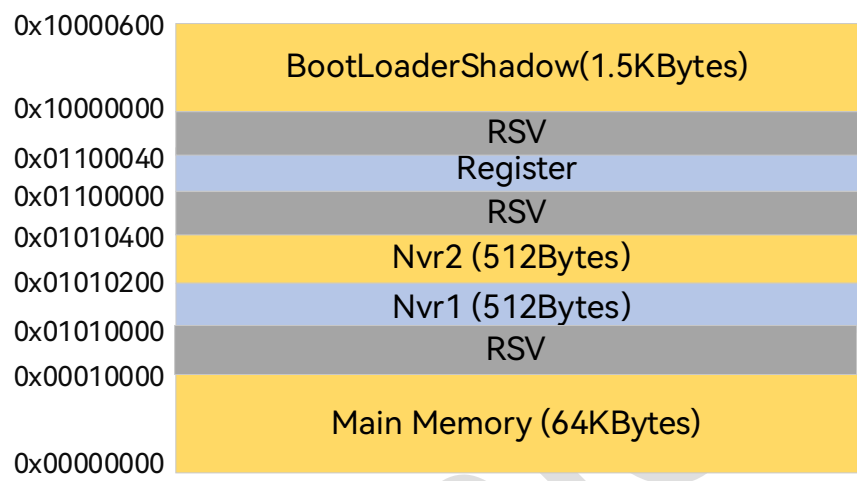


图 6-1： eFlash Main 区启动地址映射

6.4.2 自动锁总线

针对eFlash擦写的时间较长，且在eFlash擦写时间内eFlash不可读的特性，EFC控制器做了自动锁总线功能。即当eFlash在擦写时间段，如果当前CPU未访问eFlash的main区域和Nvr区域，则系统的Hready信号未被拉低，即CPU仍旧可以运行，当CPU再次访问eFlash的Main区或者Nvr区时，系统的Hready信号会立刻被拉低，CPU会被锁死，直到前次eFlash擦写完成后，且当前访问的操作也完成后，CPU才会被释放。

6.4.3 EFC 上电预读

EFC上电时，会自动从Nvr区内的指定地址读取出模拟相关的Trim参数传递给模拟相应的模块。EFC复位释放早于系统CPU复位。当EFC完成预读后，系统才会释放CPU的复位。

6.4.4 eFlash 读效率

当RD_WAIT值设置为0时，CPU取指时无效率损失，读eFlash与读取Rom在控制器端时序相同。

RD_WAIT设置为1时，EFC Ahb端接口的hreadyout信号在每个读操作时会被拉低1个周期。

6.5 软件流程

6.5.1 Read 操作

eFlash上电稳定后可以执行读操作。读操作注意配置读等待时间RD_WAIT，最小值为40ns。

6.5.2 Write 操作

eFlash上电稳定后可以执行写操作。写操作之前需要向EFC_SEC寄存器内写0x55AAAA55，否则EFC忽略此次写操作。

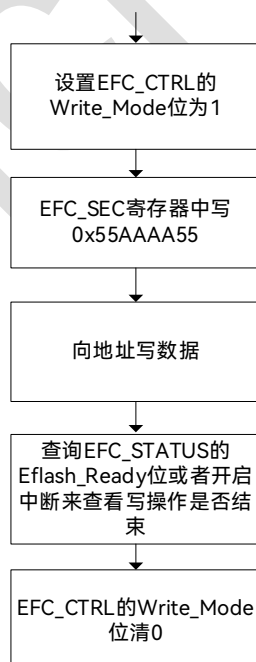


图 6-2: 写操作流程

6.5.3 Erase 操作

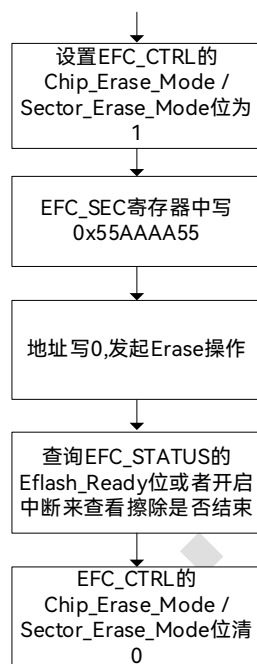


图 6-3：擦除操作流程

7NVIC

7.1概述

内嵌套向量中断控制器（NVIC）是 Cortex-M0+的一个重要组成部分。它与 CPU 处理器内核紧密耦合，实现低中断延迟以及对新到中断的有效处理，外部中断信号连接到 NVIC，NVIC 将对这些中断进行优先级排序。

Cortex-M0+处理器内置了嵌套向量中断控制器（NVIC），可支持最多 32 个中断请求（IRQ）输入：有 4 个中断优先级，可处理复杂逻辑，能进行实时控制和中断处理。

所有的 NVIC 寄存器只能采用字传输。任何试图读/写半字或字节的结果都是不可预知的。

NVIC 寄存器都是小端格式。访问处理器要正确处理处理器的大小端配置。

（关于 NVIC 更详细的内容可查看 Cortex-M0+系列内核的相关官方文档）

7.2主要特性

- 32 个外部中断，每个中断具有 4 级优先级
- 专用的不可屏蔽中断（NMI）
- 同时支持电平和脉冲中断触发
- 中断唤醒控制器，支持极低功耗休眠模式

7.3中断源

表 7-1：中断源

中断号	中断源
[0]	GPIO_PA
[1]	GPIO_PB
[2]	GPIO_PC
[3]	GPIO_PD

中断号	中断源
[4]	DMA
[5]	LPTimer1
[6]	UART0
[7]	-
[8]	UART1
[9]	I2C
[10]	SPI0
[11]	SPI1
[12]	CAN
[13]	VREF
[14]	GTimer0
[15]	GTimer1
[16]	GTimer2
[17]	QSPI
[18]	LPTimer0
[19]	-
[20]	GPIOE
[21]	-
[22]	WDT
[23]	RTC
[24]	ADC
[25]	LPTimer2
[26]	COMP2
[27]	WWDT
[28]	VDT
[29]	OPA
[30]	FLASH interrupt
[31]	-
NMI	-

8 UART0

8.1 概述

UART0 串口模块，带有 8 比特 4 级的接收 FIFO。

8.2 主要特性

- 提供标准的异步通讯位（起始位、奇偶位和停止位）
 - 生成 1 位起始位
 - 支持 8bit 的数据位宽
 - 生成 1 位校验位（可设置奇校验或偶校验），或无校验位
 - 生成 1 位停止位
 - 字节从低位到高位依次传输
- 8 比特 4 级的接收 FIFO，无发送 FIFO
- 可编程波特率（波特率可以根据参数 F/D 调整），2*8bits 波特率参数寄存器
- 支持数据通讯及错误处理中断
 - 状态位的访问可采用查询或者中断两种方式
 - FIFO 非空、半满、全满、溢出标志
 - 奇偶校验错误标志
- 具有起始位有效性检测功能
- 可支持 9600bps、19200bps、115200bps 等常见波特率的传输

8.3 寄存器描述

UART0 寄存器基地址：0x40000000

表 8-1：UART0 寄存器列表

偏置	名称	描述
0x00	UART_ISR	中断状态寄存器
0x04	UART_IER	中断使能寄存器
0x08	UART_CR	控制寄存器
0x0C	UART_TDR	发送数据寄存器
0x0C	UART_RDR	接收数据寄存器
0x10	UART_BRPL	波特率参数低位寄存器
0x14	UART_BRPH	波特率参数高位寄存器

8.3.1 中断状态寄存器 UART_ISR（偏移：00H）

比特	名称	属性	复位值	描述
31:6	RSV	-	-	保留
5	FIFO_NE	R/W	0	FIFO 非空标志： 0：FIFO 为空 1：FIFO 非空 当 FIFO 读空时，此位自动清 0。软件也可以清除此位，写 0 清除。
4	FIFO_HF	R/W	0	FIFO 半满标志： 0：FIFO 非半满 1：FIFO 半满 当 FIFO 中数据读空时，此位自动清 0。软件也可以清除此位，写 0 清除
3	FIFO_FU	R/W	0	FIFO 全满标志： 0：FIFO 非全满 1：FIFO 全满 当读取 FIFO 中数据，此位自动清 0。软件也可以清除此位，写 0 清除
2	FIFO_OV	R/W	0	Rx-FIFO 接收溢出错误： 0：没有接收溢出错误发生 1：发生了接收溢出错误 软件清除此位，写 0 清除

比特	名称	属性	复位值	描述
1	TXEND	R/W	0	UART 发送完成标志： 0：发送没有完成 1：发送完成 此位硬件置 1，软件清除，写 0 清除
0	TRE	R/W	0	UART 发送/接收奇偶校验错误标示： 0：UART 发送/接收完成时无奇偶校验错误 1：UART 发送/接收完成时有奇偶校验错误 此位硬件置 1，软件清除，写 0 清除

8.3.2 中断使能寄存器 UART_IER（偏移：04H）

比特	名称	属性	复位值	描述
31:6	RSV	-	-	保留
5	FIFO_EN	R/W	0	FIFO 非空中断使能： 0：禁止 1：使能
4	FIFO_HFEn	R/W	0	FIFO 半满中断使能： 0：禁止 1：使能
3	FIFO_FUEn	R/W	0	FIFO 全满中断使能： 0：禁止 1：使能
2	FIFO_OVEn	R/W	0	Rx-FIFO 接收溢出中断使能： 0：禁止 1：使能
1	TXENDEn	R/W	0	Uart 发送完成中断使能： 0：禁止 1：使能
0	TREEN	R/W	0	Uart 发送/接收奇偶校验错误中断使能： 0：禁止 1：使能

8.3.3 控制寄存器 UART_CR（偏移：08H）

比特	名称	属性	复位值	描述
31:5	RSV	-	-	保留

比特	名称	属性	复位值	描述
4	UART_LB	R/W	0	Uart 自测模式使能控制： 0：不使能 1：使能
3	UART_P0	R/W	0	奇偶校验使能控制： 0：有奇偶校验 1：没有奇偶校验
2	FLUSH	R/W	0	清除 UART 接收 FIFO 中的数据和指针： 0：不清除 1：清除
1	TRS	R/W	0	UART 发送数据标志： 0：发送数据不使能 1：发送数据使能
0	ODD_EN	R/W	0	奇偶校验方式选择： 0：偶校验 Even Parity 1：奇校验 Odd Parity

8.3.4 发送数据寄存器 UART_TDR（偏移：0CH）

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	UARTDATA	W	00	存放待发送的数据

8.3.5 接收数据寄存器 UART_RDR（偏移：0CH）

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	UARTDATA	R	00	存放待接收到的数据

8.3.6 波特率参数低位寄存器 UART_BRPL（偏移：10H）

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留

比特	名称	属性	复位值	描述
7:0	UARTBRPL	R/W	74	波特率参数寄存器 UARTBPRH、UARTBPRL 构成 16 位分频器。 例如：系统时钟为 32MHz，为获得 9600 波特率，则 $UARTBPR = 32 \times 1000000 \div 9600 = 0D05H$ ，即 $UARTBPRH = 0DH$ ， $UARTBPRL = 05H$ 。 例如：系统时钟为 32MHz，为获得 19200 波特率，则 $UARTBPR = 0682H$ ，即 $UARTBPRH = 06H$ ， $UARTBPRL = 82H$

8.3.7 波特率参数高位寄存器 UART_BRPH（偏移：14H）

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	UARTBRPH	R/W	01	波特率参数寄存器 UARTBPRH、UARTBPRL 构成 16 位分频器。 例如：系统时钟为 32MHz，为获得 9600 波特率，则 $UARTBPR = 32 \times 1000000 \div 9600 = 0D05H$ ，即 $UARTBPRH = 0DH$ ， $UARTBPRL = 05H$ 。 例如：系统时钟为 32MHz，为获得 19200 波特率，则 $UARTBPR = 0682H$ ，即 $UARTBPRH = 06H$ ， $UARTBPRL = 82H$

8.4 使用流程

8.4.1 串口的发送和接收

1. 配置系统配置寄存器的串口模块时钟。
2. 配置系统配置寄存器的串口模块复位使能。
3. 配置系统配置寄存器的串口引脚复用功能。
4. 配置串口中断。
5. 配置串口中断使能寄存器(是否使用中断)。

6. 配置串口控制寄存器（清除接收 FIFO 中的数据和指针）。
7. 配置串口控制寄存器（奇偶校验位等）。
8. 配置波特率。
9. 使能串口。

8.4.2 串口初始化

1. 清除 UART_ISR 寄存器，写 0 清除。
2. 配置 UART_IER，中断使能寄存器，是否产生相应的中断脉冲。
3. 设置 UART_TCR.FLUSH，清除 FIFO 中数据及 FIFO 指针。
4. 清除 UART_TCR 寄存器，写 0 清除。
5. 配置 UART_BPRL[7:0]和 UARTBPRH[7:0]。

8.4.3 串口发送字节

1. 发送、接收数据前软件可以配置波特率参数，奇偶校验类型、中断使能。
2. 设置 UART_TCR.TRS=1。
3. 写入第一个字节数据到 UART_TDR。
4. 查询发送完成标志 UART_ISR.TXEND，如果 TXEND=1 表示当前数据发送完成；软件清除此位（写 0 清除）。
5. 如果发送出错：UART 产生中断或者查询 SCCISR 寄存器标志，判断错误类型，执行相应的错误处理，处理完之后软件清除标志位。
6. 可以继续写入下一个字节到 UART_TDR。

8.4.4 串口接收字节

1. 发送、接收数据前软件可以配置波特率参数、奇偶校验类型、中断使能。
2. 接收数据，查询 UART_ISR 标志位或者等待中断，FIFO_NE（即接收数据 FIFO 非空），或者

FIFO_HF（即接收数据 FIFO 半满），或者 FIFO_FU（即接收数据 FIFO 全满）；查询到相应标志位则读取 UART_RDR 中的数据，FIFO 相应的标志位自动清除。

3. 接收错误处理：等待中断或者查询 UART_ISR 寄存器标志位，判断错误类型，执行相应的错误处理，处理完之后软件清除标志位。
4. 继续接收数据。

9 UART1

9.1 概述

UART1 串口模块，带有 16 字节的 FIFO，可小数分频。

9.2 主要特性

- 16 字节的硬件 FIFO
- 波特率支持整数和小数分频
- 支持 CTS，RTS 流控制
- 错误起始位侦测
- 帧中断检测
- 可编程位宽，奇偶校验，停止位个数
- 支持 DMA 传输方式

9.3 寄存器描述

UART1 寄存器基地址：0x40003000

表 9-1：UART1 寄存器列表

偏置	名称	描述
0x00	UART1_RBR	接收缓冲寄存器
0x00	UART1_THR	发送缓冲寄存器
0x00	UART1_DLL	波特率分频低位寄存器
0x04	UART1_DLH	波特率分频高位寄存器
0x04	UART1_IER	中断使能寄存器
0x08	UART1_IIR	中断状态寄存器
0x08	UART1_FCR	FIFO 控制寄存器
0x0C	UART1_LCR	LINE 控制寄存器

偏置	名称	描述
0x10	UART1_MCR	流控制寄存器
0x14	UART1_LSR	LINE 中断状态寄存器
0x18	UART1_MSR	流状态寄存器
0x7C	UART1_USR	状态寄存器
0x80	UART1_TFL	发送 FIFO 数据个数寄存器
0x84	UART1_RFL	接收 FIFO 数据个数寄存器
0xC0	UART1_DLF	小数分频寄存器
0xC4	UART1_RAR	接收地址匹配寄存器
0xC8	UART1_TAR	发送地址匹配寄存器

9.3.1 接收缓冲寄存器 UART1_RBR (偏移: 00H)

比特	名称	属性	复位值	描述
31:9	RSV	-	-	保留
8:0	RBR	R	0	接收数据寄存器。此字段为接收 FIFO 入口, 仅当 UART1_LCR 的 DLAB 位为 0 时, 此字段才可以访问。

9.3.2 发送缓冲寄存器 UART1_THR (偏移: 00H)

比特	名称	属性	复位值	描述
31:9	RSV	-	-	保留
8:0	THR	W	0	发送数据寄存器。此字段为发送 FIFO 入口, 仅当 UART1_LCR 的 DLAB 位为 0 时, 此字段才可以访问。

9.3.3 波特率分频低位寄存器 UART1_DLL (偏移: 00H)

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	DLL	R/W	0	波特率配置寄存器低位。仅当 UART1_LCR 的 DLAB 位为 1 时, 此字段才可以访问。 波特率整数部分计算公式: $\text{baud rate} = \text{fclk} / (16 * \{\text{DLH}, \text{DLL}\})$

9.3.4 波特率分频高位寄存器 UART1_DLH (偏移: 04H)

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	DLH	R/W	0	波特率配置寄存器高位。仅当 UART1_LCR 的 DLAB 位为 1 时，此字段才可以访问。 波特率整数部分计算公式： $\text{baud rate} = \text{fclk} / (16 * \{\text{DLH}, \text{DLL}\})$

9.3.5 中断使能寄存器 UART1_IER (偏移: 04H)

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7	PTIME	R/W	0	THRE 中断使能，仅当 UART1_LCR 的 DLAB 位为 0 时，此字段才可以访问。 0: 禁止 THRE 中断 1: 使能 THRE 中断
6:3	RSV	-	-	保留
2	ELSI	R/W	0	LINE 中断使能，仅当 UART1_LCR 的 DLAB 位为 0 时，此字段才可以访问。 0: 禁止 LINE 中断 1: 使能 LINE 中断
1	ETBEI	R/W	0	发送 FIFO 空中断使能，仅当 UART1_LCR 的 DLAB 位为 0 时，此字段才可以访问。 0: 禁止发送 FIFO 空中断 1: 使能发送 FIFO 空中断
0	ERBFI	R/W	0	接收数据中断使能，仅当 UART1_LCR 的 DLAB 位为 0 时，此字段才可以访问。 0: 禁止接收 FIFO 非空中断 1: 使能接收 FIFO 非空中断

9.3.6 中断状态寄存器 UART1_IIR (偏移: 08H)

比特	名称	属性	复位值	描述
31:4	RSV	-	-	保留
7:6	FIFOSE	R	0	FIFO 使能标志 00: FIFO 禁止 11: FIFO 使能
5:4	RSV	-	-	保留
3:0	IID	R	0001	状态 ID: 0000: CTS/RTS 中断状态 0001: 无中断 0010: 发送 FIFO 空 0100: 接收 FIFO 非空 0110: LINE 中断状态 0111: Busy 状态 1100: TimeOut 状态, 当使能 FIFO 和接收 FIFO 非空中断后, 如果在接收 FIFO 中存在至少 1 个数据, 在 4 个 UART 帧内, CPU 如果未读 FIFO, 则此字段会进入 TimeOut 中断状态。 其它: 保留

9.3.7 FIFO 控制寄存器 UART1_FCR (偏移: 08H)

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:6	RT	W	0	接收 FIFO 非空中断设置, 当 FIFO 中数据大于或等于此设置对应的 FIFO 状态时, 接收 FIFO 非空中断置位: 00: 1 帧数据 01: 4 帧数据 10: 8 帧数据 11: 14 帧数据

比特	名称	属性	复位值	描述
5:4	TET	W	0	发送 FIFO 空中断设置，当 FIFO 中数据少于或等于此设置对应的 FIFO 状态时，发送 FIFO 空中断置位： 00: FIFO 空 01: 2 帧数据 10: 4 帧数据 11: 8 帧数据
3	RSV	-	-	保留
2	XFIFOR	W	0	发送 FIFO 复位位： 0: 不复位发送 FIFO 1: 复位发送 FIFO
1	RFIFOR	W	0	接收 FIFO 复位位： 0: 不复位接收 FIFO 1: 复位接收 FIFO
0	FIFOE	W	0	FIFO 使能位： 0: 禁止 FIFO 1: 使能 FIFO 改变此位的值将会同时复位接收和发送 FIFO。

9.3.8 LINE 控制寄存器 UART1_LCR（偏移：0CH）

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7	DLAB	R/W	0	UART1_DLL 和 UART1_DLH 寄存器访问设置位。 0: UART1_RBR/UART1_THR 可以通过偏移地址 0x0 访问，UART1_IER 可以通过偏移地址 0x4 访问。 1: UART1_DLL可以通过偏移地址 0x0 访问，UART1_DLH可以通过偏移地址 0x4 访问。
6	RSV	-	-	保留
5	SEPS	R/W	0	奇偶校验位强制设置位，仅当 UART 处于空闲状态时可写： 0: 奇偶校验位强制设置功能禁止 1: 当 PEN 为 1， EPS 为 1，奇偶校验位被

比特	名称	属性	复位值	描述
				传输并检查为逻辑 0 ； PEN 为 1，当 EPS 为 0 时，奇偶校验位被传输并检查为逻辑 1；当 PEN 为 0 时，发送和接收均无奇偶校验
4	EPS	R/W	0	奇偶校验选择位，仅当 UART 处于空闲状态时可写： 0：奇校验 1：偶校验
3	PEN	R/W	0	奇偶校验位使能设置，仅当 UART 处于空闲状态时可写： 0：奇偶校验位禁止 1：奇偶校验位使能
2	STOP	R/W	0	STOP 比特长度设置，仅当 UART 处于空闲状态时可写： 0：1 比特 STOP 位 1：1.5 比特 STOP 位
1:0	DLS	R/W	0	UART 帧数据长度设置位，仅当 UART 处于空闲状态时可写： 00：5 比特 01：6 比特 10：7 比特 11：8 比特

9.3.9 流控制寄存器 UART1_MCR（偏移：10H）

比特	名称	属性	复位值	描述
31:6	RSV	-	-	保留
5	AFCE	R/W	0	CTS/RTS 自动流控制选择： 0：禁止 1：使能
4:2	RSV	-	-	保留
1	RTS	R/W	0	RTS 接口软件控制位： 0 ： RTS 请求输出无效 1 ： RTS 请求输出有效
0	RSV	-	-	保留

9.3.10 LINE 中断状态寄存器 UART1_LSR (偏移: 14H)

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7	RFE	R	0	接收 FIFO 错误标志 (可触发 LINE 中断): 0: 接收 FIFO 中数据没有错误 1: 接收 FIFO 中数据至少有一个有奇偶校验错误或者 UART 帧格式错误 当接收 FIFO 中出错的数据是下一个将要读取的数据, 且接收 FIFO 中其它的数据没有错误时, 读取此寄存器清 0
6	TEMT	R	1	发送完成标志: 0: 发送未完成 1: 发送完成, 发送 FIFO 为空, 且移位寄存器为空
5	THRE	R	1	发送 FIFO 空标志: 0: 发送 FIFO 满 1: 发送 FIFO 空
4	RSV	-	-	保留
3	FE	R	0	帧格式出错标志 (可触发 LINE 中断): 0: 帧格式未出错 1: 帧格式错误 读此寄存器清 0
2	PE	R	0	奇偶校验出错标志 (可触发 LINE 中断): 0: 奇偶校验未出错 1: 奇偶校验错误 读此寄存器清 0
1	OE	R	0	接收 FIFO 溢出标志 (可触发 LINE 中断): 0: 接收 FIFO 非溢出 1: 接收 FIFO 溢出 读此寄存器清 0
0	DR	R	0	接收 FIFO 非空标志: 0: 接收 FIFO 空 1: 接收 FIFO 非空

9.3.11 流状态寄存器 UART1_MSR (偏移: 18H)

比特	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4	CTS	R	0	CTS 标志位: 0: 无 CTS 请求 1: 有 CTS 请求
3:0	RSV	-	-	保留

9.3.12 状态寄存器 UART1_USR (偏移: 7CH)

比特	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4	RFF	R	0	接收 FIFO 满标志: 0: 接收 FIFO 非满 1: 接收 FIFO 满
3	RFNE	R	0	接收 FIFO 非空标志: 0: 接收 FIFO 空 1: 接收 FIFO 非空
2	TFE	R	1	发送 FIFO 空标志: 0: 发送 FIFO 非空 1: 发送 FIFO 空
1	TFNF	R	1	发送 FIFO 非满标志: 0: 发送 FIFO 满 1: 发送 FIFO 非满
0	BUSY	R	0	1: UART1 正在传输 0: UART1 处于空闲状态

9.3.13 发送 FIFO 数据个数寄存器 UART1_TFL (偏移: 80H)

比特	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4:0	TFL	R	0	发送 FIFO 中数据个数位

9.3.14 接收 FIFO 数据个数寄存器 UART1_RFL (偏移: 84H)

比特	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4:0	RFL	R	0	接收 FIFO 中数据个数位

9.3.15 小数分频寄存器 UART1_DLF (偏移: C0H)

比特	名称	属性	复位值	描述
31:5	RSV	-	-	保留
3:0	DLF	R/W	0	小数分频寄存器: 小数部分波特率为 DLF/16。 计算公式为: $(PCLK\%(BAUDRATE*16)) / BAUDRATE$ 。

9.3.16 接收地址匹配寄存器 UART1_RAR (偏移: C4H)

比特	名称	属性	复位值	描述
31:5	RSV	-	-	保留
7:0	RAR	R/W	0	接收地址匹配寄存器。此寄存器只有在 UART 处于空闲状态时可写。

9.3.17 发送地址匹配寄存器 UART1_TAR (偏移: C8H)

比特	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4:0	TAR	R/W	0	发送地址匹配寄存器。此寄存器只有在 UART 处于空闲状态时可写。

9.4 使用流程

9.4.1 UART1 发送流程

1. 设置 UART1_MCR 寄存器。
2. 设置 UART1_LCR 寄存器的第 7 比特 DLAB 为 1。
3. 设置 UART1_DLL/UART1_DLH/UART1_DLF 寄存器。
4. 设置 UART1_LCR 寄存器的第 7 比特 DLAB 为 0，设置 UART1_LCR 寄存器的其它位。
5. 设置 UART1_FCR 寄存器。
6. 设置 UART1_IER 寄存器。
7. 写 UART1_THR 寄存器，向发送 FIFO 中填写数据。
8. 查询 UART1_IIR 中断状态。
9. 完成传输。

9.4.2 UART1 接收流程

1. 设置 UART1_MCR 寄存器。
2. 设置 UART1_LCR 寄存器的第 7 比特 DLAB 为 1。
3. 设置 UART1_DLL/UART1_DLH/UART1_DLF 寄存器。
4. 设置 UART1_LCR 寄存器的第 7 比特 DLAB 为 0，设置 UART1_LCR 寄存器的其它位。
5. 设置 UART1_FCR 寄存器。
6. 设置 UART1_IER 寄存器。
7. 查询 UART1_IIR 中断状态。
8. 读取 UART1_RBR，取走收到的数据。
9. 完成传输。

9.4.3 CTS 和 RTS 控制流功能设置流程

● CTS

CTS为UART输入端口，低电平有效，表示UART可以发送数据。如果CTS输入状态为 1，写 UART1_THR 寄存器时，数据只会保存在发送FIFO中不会被发出，为0时开始发送。

CTS配置流程如下：

1. 配置 UART1_CTS 引脚。
2. 配置 REG_UART1_MCR，使能 CTS/RTS 自动流控制。
3. 配置 REG_UART1_FCR，使能 FIFO。
4. UART1_CTS 引脚输入为低电平时，UART 正常发送数据；输入为高电平时，数据保存在发送 FIFO 中。



● RTS

RTS为UART输出端口，低电平有效，输出为低电平时，表示UART已经准备好可以接收数据了；当接收FIFO中数据个数大于FIFO控制寄存器UART1_FCR中接收FIFO非空中断设置的帧数据个数触发中断条件时，RTS输出状态为高电平，表示UART不能接收更多数据。

RTS 配置流程如下：

1. 配置 UART1_RTS 引脚。
2. 配置 REG_UART1_MCR，使能 CTS/RTS 自动流控制，RTS 接口软件控制位为请求输出有效。
3. 配置 REG_UART1_FCR，使能 FIFO，接收 FIFO 非空中断设置。

9.4.4 UART1 DMA 传输配置流程

UART1 模块可支持 DMA 传输功能，支持三种传输模式：Memory to Peripheral 模式、Peripheral to Memory 模式、Peripheral to Peripheral 模式。配置流程如下：

1. 配置开启 DMA 模块时钟与复位 PERI_RESET / PERI_CLKEN。
2. 配置通道控制信息寄存器 DMA_CH_CTRL_Cx。
3. 根据实际应用配置数据位宽，传输模式（8 位位宽、内存到外设模式）。
 - 例如：REG_DMA_CHCTRLC(channel_index) |= DMA_TR_WIDTH_8。
 - 例如：REG_DMA_CHCTRLC(channel_index) |= DMA_MEM_TO_PERIP。
4. 配置【目的外设】和【源外设】（目的外设为 uart1_tx，源外设为 mem）。

此位在用到外设的模式下起效。

 - 例如：REG_DMA_CHCTRLC(channel_index) |= DMA_DST_PER_UART1_TX。
 - 例如：REG_DMA_CHCTRLC(channel_index) |= DMA_SRC_PER_MEMORY。
5. 配置【目标地址】和【源地址】是否随数据传输递增（原地址递增、目标地址不变）。
 - 例如：REG_DMA_CHCTRLC(channel_index) |= DMA_SINC_INC。
 - 例如：REG_DMA_CHCTRLC(channel_index) |= DMA_DINC_NOC。
6. 如需使用中断，则配置 DMA 中断指示寄存器 DMA_INT_STATUS，使能对应的通道中断。
7. 配置【源地址】和【目标地址】及【数据块尺寸】。

DMA_SRC_ADDR_Cx、DMA_DST_ADDR_Cx、DMA_CH_CTRL_Cx

 - 例如：REG_DMA_SRCADDRRC(channel_index) = (uint32_t)src_addr。
 - 例如：REG_DMA_DSTADDRRC(channel_index) = (uint32_t)dest_addr。
 - 例如：REG_DMA_CHCTRLC(channel_index) |= (length<<15)。
8. 等待上述配置、以及相应的原地址和目标准备就绪，使能 DMA (DMAC_EN)。
9. 根据实际使用情况，检测 DMA 中断状态寄存器 DMA_INT_STATUS，跟踪传输状态。

10 I2C

10.1 概述

I2C 总线接口连接微控制器和串行 I2C 总线。I2C 模块接收和发送数据，并将数据从串行转换成并行，或并行转换成串行。I2C 模块通过数据引脚 SDA 和时钟引脚 SCL 连接到 I2C 总线，控制所有 I2C 总线规定的时序。本模块支持主模式和从模式。

10.2 主要特性

- 支持主机接收、发送，从机接收、发送四种工作模式
- 支持标准（100Kbps）、快速（400Kbps）、高速（1Mbps）三种工作速率
- 支持 7 位寻址功能和 10 位寻址功能
- 支持广播地址
- 支持中断查询功能

10.3 寄存器描述

I2C 寄存器基地址：0x40005400

表 10-1：I2C 寄存器列表

偏置	名称	描述
0x00	I2C_CR	I2C 配置寄存器
0x04	I2C_CLR	I2C 配置清除寄存器
0x08	I2C_STAT	I2C 状态寄存器
0x0C	I2C_DATA	I2C 数据寄存器
0x10	I2C_CCR	I2C 波特率配置寄存器
0x14	I2C_SAD0	I2C SLAVE 地址寄存器 0
0x18	I2C_SADM0	I2C SLAVE 地址屏蔽寄存器 0
0x1C	I2C_XSAD0	I2C SLAVE 扩展地址寄存器

偏置	名称	描述
0x20	I2C_XSADM0	I2C SLAVE 扩展地址屏蔽寄存器
0x24	I2C_SRST	I2C 复位寄存器
0x28	I2C_SAD1	I2C SLAVE 地址寄存器 1
0x2C	I2C_SADM1	I2C SLAVE 地址屏蔽寄存器 1
0x30	I2C_SAD2	I2C SLAVE 地址寄存器 2
0x34	I2C_SADM2	I2C SLAVE 地址屏蔽寄存器 2
0x38	I2C_SAD3	I2C SLAVE 地址寄存器 3
0x3c	I2C_SADM3	I2C SLAVE 地址屏蔽寄存器 3

10.3.1 I2C 配置寄存器 I2C_CR (偏移: 00H)

比特	名称	属性	复位值	描述
31:9	RSV	-	-	保留
8	GCAVAL	R	0	General Call 地址标志位。 0: 未收到 General Call 地址 1: 收到 General Call 地址
7	IEN	R/W	0	I2C 模块中断使能。 0: 中断禁止 1: 中断使能
6	ENAB	R/W	0	从模式下, I2C 模块使能位。 0: I2C 从模式下, 模块禁止; I2C 不进行地址匹配, 忽略掉 SCL/SDA 线上的信息 1: I2C 从模式下, 模块使能
5	STA	R/W	0	开始标志使能。 0: 不发送 START 标志 1: 发送 START 标志; 发送 START 表之后, 自动清 0
4	STP	R/W	0	停止标志使能。 0: 不发送 STOP 标志 1: 发送 STOP 标志; 发送 STOP 表之后, 自动清 0
3	IFLG	R/W	0	中断标志位: I2C_STAT 寄存器处于 0xf8 以外的任何状态, 此位都会置位。 写 I2C_CLR 寄存器的 CLR_IFLG 位, 清

比特	名称	属性	复位值	描述
				0。STP 位写 1 时，即发送 STOP 标志后，此位也将清 0
2	AAK	R/W	0	应答标志使能。 0：应答 NACK 1：应答 ACK 置 1 后，写 I2C_CLR 寄存器的 CLR_AAK 位清 0
1	SLAV10M	R	0	作为 SLAVE 时，收到的数据与扩展地址寄存器中数据相匹配的标志位。 0：接收到的数据与 SLAVE 扩展地址寄存器中的数据不相同 1：接收到的数据与 SLAVE 扩展地址寄存器中的数值相匹配
0	SLAV7M	R	0	作为 SLAVE 时，收到的数据与地址寄存器中数据相匹配的标志位。 0：接收到的数据与 SLAVE 地址寄存器中的数据不相同 1：接收到的数据与 SLAVE 地址寄存器中的数值相匹配

10.3.2 I2C 配置清除寄存器 I2C_CLR（偏移：04H）

比特	名称	属性	复位值	描述
31:9	RSV	-	-	保留
8	GCAVAL	R/W	0	General Call 功能使能位： 0：禁止 1：使能
7	CLR_IEN	W	0	I2C 模块中断使能清除寄存器： 0：中断使能状态保持 1：清除中断使能
6	CLR_ENAB	W	0	I2C 模块使能清除寄存器： 0：保持 I2C 当前状态 1：关闭 I2C 模块
5	CLR_STA	W	0	开始标志清除寄存器： 0：保持 START 当权设置不变 1：清除发送 START 标志

比特	名称	属性	复位值	描述
4	RSV	-	-	保留
3	CLR_IFLG	W	0	中断标志清除寄存器： 0：保持中断标志不变 1：清除中断标志
2	CLR_AAK	W	0	应答标志清除寄存器： 0：保持应答标志不变 1：清除应答标志
1:0	RSV	-	-	保留

10.3.3 I2C 状态寄存器 I2C_STAT（偏移：08H）

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	STA	R	0xF8	I2C 状态寄存器

I2C_STAT 寄存器 STA 字段不同代码代表的意义：

状态代码	I2C 总线和硬件状态
0x08	已发送 START 标志
0x10	已发送 RESTART 标志
0x18	已发送 SLAVE 地址加 W 标志，并接收 ACK 位
0x20	已发送 SLAVE 地址加 W 标志，并接收 NAK 位
0x28	已发送 I2C_DATA 中的数据，已接收 ACK
0x30	已发送 I2C_DATA 中的数据，接收到 NAK
0x38	丢失仲裁
0x40	已发送 SLAVE 地址加 R 标志，并加收到 ACK
0x48	已发送 SLAVE 地址加 R 标志，并加收到 NAK
0x50	已接收数据字节，ACK 已发出
0x58	已接收数据字节，NAK 已发出
0x60	已接收自身的 SLAVE 寄存器地址加 W 标志，ACK 已发出。
0x68	丢失掉仲裁，并且以接收到自身的 SLAVE 寄存器地址加 W 标志，ACK 已发出
0x70	已接收通用调用地址（0x00）；已发出 ACK
0x78	丢失掉仲裁，并且已接收到通用调用地址加 W 标志，ACK 已发出
0x80	前一次寻址使用自身从地址；已接收数据字节；已返回 ACK
0x88	前一次寻址使用自身从地址；已接收数据字节；已返回非 ACK

状态代码	I2C 总线和硬件状态
0x90	前一次寻址使用通用调用地址；已接收数据；已返回 ACK
0x98	前一次寻址使用通用调用地址；已接收数据；已返回非 ACK
0xA0	当使用从接收/从发送模式中静态寻址时, 接收到停止条件或重复起始条件
0xA8	已接收自身的从地址加 R 标志；已返回 ACK
0xB0	丢失掉仲裁, 并且已接收到通用调用地址加 R 标志, ACK 已发出
0xB8	已发送数据；已接收 ACK
0xC0	已发送数据字节；已接收非 ACK
0xC8	装入的数据字节已被发送；已接收 ACK
0xF8	无可用的相关状态信息
0x00	由于非法的起始或停止条件的出现, 在主机或被选中的从机将出现总线错误; 当外部干扰使I2C 进入未定义的状态时也会出0x00状态
0xE0	已发送第二次设备地址, 已接收 ACK
0xE8	已发送第二次设备地址, 已接收非 ACK

10.3.4 I2C 数据寄存器 I2C_DATA（偏移：0CH）

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	DATA	R/W	0x0	I2C 数据寄存器。 在 I2C 发送模式下, 写发送数据到这个寄存器 在 I2C 接收模式下, 读接收数据从这个寄存器

10.3.5 I2C 波特率配置寄存器 I2C_CCR（偏移：10H）

比特	名称	属性	复位值	描述
31:7	RSV	-	-	保留
6:4	CCRM	R/W	0x0	波特率配置位 M
3:0	CCRN	R/W	0x0	波特率配置位 N

$F_{OSCL} = F_{SCL} = P_{CLK} / (2^M \times (N+1) \times 10)$; 其中, F_{OSCL} 是 I2C 接口输出的 SCL 得频率。

10.3.6 I2C SLAVE 地址寄存器 0 I2C_SAD0 (偏移: 14H)

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:1	ADR0	R/W	0x0	I2C 从机模式地址 0
0	GC0	R/W	0x0	广播地址应答使能。 0: 不使能 1: 使能

10.3.7 I2C SLAVE 地址屏蔽寄存器 0 I2C_SADM0 (偏移: 18H)

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:1	AMR0	R/W	0x7f	I2C 从机模式地址屏蔽寄存器 0。每一位与 ADR0 中的位相对应: 0: 在此 I2C 模块作为从机时, 不比较 ADR0 中对应位的值 1: 在此 I2C 模块作为从机时, 比较 ADR0 中对应位的值。
0	RSV	-	-	保留

10.3.8 10 比特 I2C SLAVE 地址寄存器 I2C_XSAD (偏移: 1CH)

比特	名称	属性	复位值	描述
31:11	RSV	-	-	保留
10:1	XADR	R/W	0x0	I2C 从机模式 10 比特地址位。
0	XGC	R/W	0x0	10 比特地址模式下, 广播地址应答使能 0: 不使能 1: 使能

10.3.9 10 比特 I2C SLAVE 地址屏蔽寄存器 I2C_XSADM (偏移: 20H)

比特	名称	属性	复位值	描述
31:9	RSV	-	-	保留

比特	名称	属性	复位值	描述
8:1	XAMR	R/W	0xff	10 比特 I2C 从机模式地址屏蔽寄存器 0。每一位与 XADR 中的位相对应。 对应位为 1 表示，在此 I2C 模块作为从机时，比较 XADR 中对应位的值。 对应位为 0 表示，在此 I2C 模块作为从机时，不比较 XADR 中对应位的值
0	RSV	-	-	保留

10.3.10 I2C 复位寄存器 I2C_SRST（偏移：24H）

比特	名称	属性	复位值	描述
31:0	SRST	W	0x0	写此寄存器，复位 I2C 模块

10.3.11 I2C SLAVE 地址寄存器 1 I2C_SAD1（偏移：28H）

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:1	ADR1	R/W	0x0	I2C 从机模式地址 1
0	GC1	R/W	0x0	广播地址应答使能： 0：不使能 1：使能

10.3.12 I2C SLAVE 地址屏蔽寄存器 1 I2C_SADM1（偏移：2CH）

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:1	AMR1	R/W	0x7f	I2C 从机模式地址屏蔽寄存器 1。每一位与 ADR1 中的位相对应。 对应位为 1 表示，在此 I2C 模块作为从机时，比较 ADR1 中对应位的值。 对应位为 0 表示，在此 I2C 模块作为从机时，不比较 ADR1 中对应位的值
0	RSV	-	-	保留

10.3.13 I2C SLAVE 地址寄存器 2 I2C_SAD2 (偏移: 30H)

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:1	ADR2	R/W	0x0	I2C 从机模式地址 2
0	GC2	R/W	0x0	广播地址应答使能: 0: 不使能 1: 使能

10.3.14 I2C SLAVE 地址屏蔽寄存器 2 I2C_SADM2 (偏移: 34H)

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:1	AMR2	R/W	0x7f	I2C 从机模式地址屏蔽寄存器 2。每一位与 ADR2 中的位相对应。 对应位为 1 表示, 在此 I2C 模块作为从机时, 比较 ADR2 中对应位的值。 对应位为 0 表示, 在此 I2C 模块作为从机时, 不比较 ADR2 中对应位的值
0	RSV	-	-	保留

10.3.15 I2C SLAVE 地址寄存器 2 I2C_SAD3 (偏移: 38H)

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:1	ADR3	R/W	0x0	I2C 从机模式地址 3
0	GC3	R/W	0x0	广播地址应答使能: 0: 不使能 1: 使能

10.3.16 I2C SLAVE 地址屏蔽寄存器 3 I2C_SADM3 (偏移: 3CH)

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:1	AMR3	R/W	0x7f	I2C 从机模式地址屏蔽寄存器 2。每一位与 ADR3 中的位相对应

比特	名称	属性	复位值	描述
				对应位为 1 表示，在此 I2C 模块作为从机时，比较 ADR3 中对应位的值。 对应位为 0 表示，在此 I2C 模块作为从机时，不比较 ADR3 中对应位的值
0	RSV	-	-	保留

10.4 协议描述

标准 I2C 协议包含了五个部分：起始信号或重复起始信号、从机地址传输和 R/W 位传输、数据信号、确认信号和结束信号。

10.4.1 I2C 通信协议（7 位寻址）

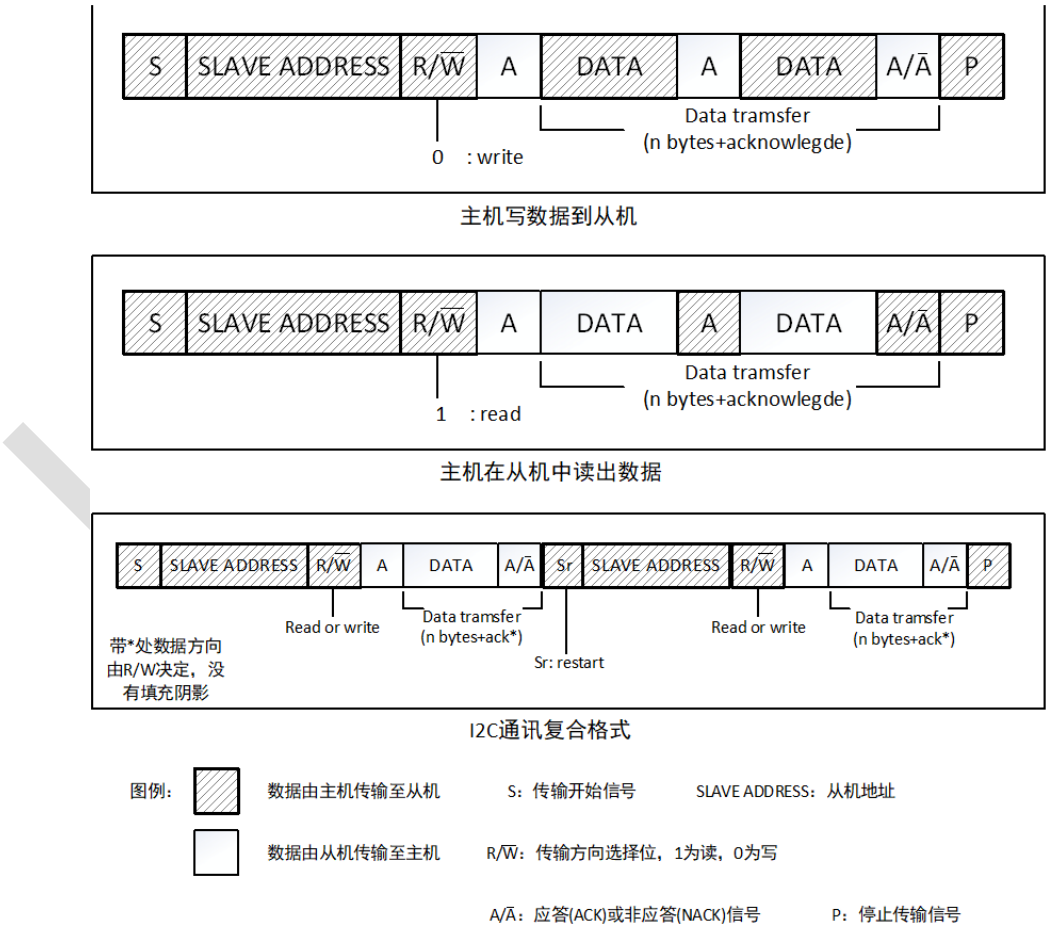


图 10-1: I2C 通信协议（7 位寻址）框图

I2C 的基本读写过程如下:

在第一幅图中, 配置的方向为“写数据 (W)”。主机在广播完从机地址, 接收到应答信号后, 开始正式向从机发送数据 (DATA), 数据包的大小为 8 位, 主机每发送完一个字节数据后都要等待从机发来的应答信号 (ACK), 然后再发送下一个字节数据。发送数据包的数量没有限制。最后当数据传输结束时, 主机向从机发送一个停止传输信号 (P), 传输停止。

在第二幅图中, 配置的方向为“读数据 (R)”。主机在广播完从机地址, 接收到应答信号后, 开始接收从机发送的数据包 (DATA), 数据包的大小为 8 位, 主机每接收完一个字节数据后都要发送一个应答信号 (ACK), 从机在收到此应答信号以后再发送下一个字节数据。接收数据包的数量没有限制。最后当主机希望停止数据传输时, 要向从机发送一个非应答信号 (NACK), 则从机停止传输。

除了基本的读写, I2C 通讯更常用的是复合格式, 即第三幅图的情况, 在该传输中, 主机会在第一次传输的数据段 (DATA 部分) 中发送从设备内部的寄存器或存储器地址 (注意不是 SLAVE ADDRESS); 在第二次传输中, 对该地址的内容进行读写。

10.4.2 I2C 通信协议 (10 位寻址)

I2C 总线的 10bit 寻址和 7bit 寻址是兼容的, 这样就可以在同一个总线上同时使用 7bit 地址和 10bit 地址模式的设备。10bit 的从机地址由开始条件(S)或重复开始条件(Sr)后的两个字节数据组成。第一个字节的前 7 位是 1111 0XX, XX 是 10bit 地址的最高有效位的前两位(A9, A8), 第 8bit 是读写位, 决定传输方向。第二个字节为 10bit 地址剩下的 8 位(A7-A0)。

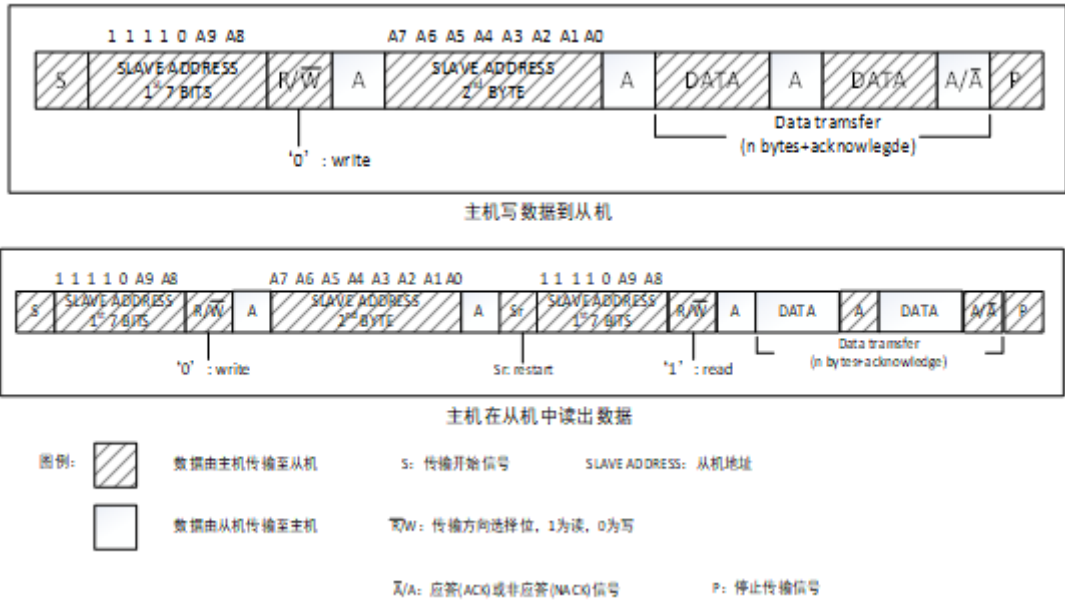


图 10-2: I2C 通信协议 (10 位寻址) 框图

I2C 的基本读写过程如下:

在第一幅图中, 主机作为发送器向从机发送数据。当接收到 START 条件后的 7bit 地址, 从机会用存在 XSAD 寄存器里的地址与接收到的第一个字节 (11110XX) 进行比较, 并检查第八个 bit (读写位) 是否为 0。有可能多个设备都匹配并产生应答 ACK。接下来从机开始匹配自己的地址与第二个字节的 8 个 bit (XXXXXXXX), 这时就只有一个从机匹配并产生应答 ACK。匹配完成后主机可开始向从机传输数据。被主机寻址匹配的从机会保持被寻址的状态直到接收到终止条件或者是重复开始条件后跟着一个不同的从机地址。

在第二幅图中, 主机作为接收器从从机接收数据。在第二个应答 ACK 之前, 处理过程与上面图一致。在重复开始条件 (Sr) 之后, 匹配的从机会保持被寻址的状态, 此时从机会检查 Sr 之后的第一个字节的前 7bit 是否正确, 并测试第 8 个 bit 是否为 1 (读)。如果是, 从机就认定它被作为一个发送器被寻址并产生应答 ACK。匹配完成后从机可开始向主机传输数据。

10.5 使用流程

10.5.1 初始化程序

将 I2C 接口初始化用作从机和/或主机的例子：

1. 将自身的从机地址装入 I2C_SAD0/ I2C_SAD1/ I2C_SAD2/ I2C_SAD3/ I2C_XSAD，设置好地址匹配寄存器 I2C_SADM0/ I2C_SADM1/ I2C_SADM2/ I2C_SADM3/ I2C_XSADM，I2C_SADx 寄存器 GC 位置位，使能广播地址应答（如果需要）。
2. I2C_CR 寄存器 IEN 位置位，使能 I2C 中断。
3. 写 I2C_CCR 寄存器设置 I2C 通信速率（标准/快速/高速）。
4. 对于从机模式，设置 I2C_CR 寄存器的 AAK、ENAB 位。

10.5.2 主机发送功能

1. 向 I2C_CR 寄存器的 STA 位写 1，发出 START 标志。
2. 等待 I2C_STAT 寄存器数值变为 0x08（已发送 START 标志）。
3. 向 I2C_CLR 中的 CLR_STA 位写 1，清除 STA 发送标志。
4. 向 I2C_DATA 寄存器写入 SLA（7 位/10 位）+W（0）。
5. 向 I2C_CLR 中的 CLR_IFLG 位写 1，发送 SLA+W。
6. 等待 I2C_STAT 寄存器数值变为 0x18（已发送 SLAVE 地址加 W 标志，并接收 ACK 位）。
7. 向 I2C_DATA 寄存器写入待发送的数据/要写入的从设备内存地址（10bit 寻址为发送第二次设备地址+w）。
8. 接收到从设备 I2C_STAT_SECOND_ADD_ACK=0xe0 状态为第二次地址已经发送并收到 ACK，接着循环发送数据（10bit 寻址才有此处通讯动作）。
9. 向 I2C_CLR 中的 CLR_IFLG 位写 1，发送数据。
10. 等待 I2C_STAT 寄存器数值变为 0x28（已发送 I2C_DATA 中的数据，已接收 ACK）。
11. 重复上述的步骤，直到待发的数据全部发送完毕。

12. 向 I2C_CR 寄存器的 STP 位写 1，发送 STOP 标志，传输完成。

10.5.3 主机接收功能

1. 向 I2C_CR 寄存器的 STA 位写 1，发出 START 标志。
2. 等待 I2C_STAT 寄存器数值变为 0x08（已发送 START 标志）。
3. 向 I2C_CLR 中的 CLR_STA 位写 1，清除 STA 发送标志。
4. 向 I2C_DATA 寄存器写入 SLA（7 位/10 位）+W(0)。
5. 向 I2C_CLR 中的 CLR_IFLG 位写 1，发送 SLA+W。
6. 等待 I2C_STAT 寄存器数值变为 0x18（已发送 SLAVE 地址加 W 标志，并接收 ACK 位）。
7. 向 I2C_DATA 寄存器写入待发送的数据/要写入的从设备内存地址（10 寻址方式则为发送第二次从设备地址+W）。
8. 等待第二次地址发送成功 I2C_STAT=e0，继续发出 START 标志（这里在 10 寻址方式出现）。
9. 向 I2C_CLR 中的 CLR_IFLG 位写 1，发送数据。
10. 向 I2C_CR 寄存器的 STA 位写 1，发出 RESTART 标志。
11. 等待 I2C_STAT 寄存器数值变为 0x10（已发送 RESTART 标志）。
12. 向 I2C_DATA 寄存器写入 SLA（7 位或 10 位）+R(1)。
13. 向 I2C_CLR 中的 CLR_IFLG 位写 1，发送 SLA+R。
14. 等待 I2C_STAT 寄存器数值变为 0x40（已发送 SLAVE 地址加 R 标志，并加收到 ACK）。
15. 向 I2C_CR 寄存器的 AAK 位写 1，设置 I2C_CR 寄存器的 AAK 位。
16. 向 I2C_CLR 中的 CLR_IFLG 位写 1，开始接收数据。
17. 等待 I2C_STAT 寄存器数值变为 0x50（已接收数据字节，ACK 已发出），读取 I2C_DATA 中收到的数据。
18. 重复上述的步骤，直到接收完所有数据。
19. 向 I2C_CR 寄存器的 STP 位写 1，发送 STOP 标志，传输完成。

10.5.4 从机接收功能

- I2C_STAT: 0x60 (已接收自身的 SLAVE 寄存器地址加 W 标志, ACK 已发出)
 - 向 I2C_CR 寄存器的 AAK 位写 1, 设置 I2C_CR 寄存器的 AAK 位
 - 向 I2C_CLR 中的 CLR_IFLG 位写 1, 开始接收数据
- I2C_STAT: 0x68 (用作主机时丢失掉仲裁, 并且已接收到自身的 SLAVE 寄存器地址加 W 标志, ACK 已发出)
 - 向 I2C_CR 寄存器的 AAK 位写 1, 设置 I2C_CR 寄存器的 AAK 位
 - 向 I2C_CLR 中的 CLR_IFLG 位写 1, 开始接收数据
- I2C_STAT: 0x78 (用作主机时丢失掉仲裁, 并且已接收到通用调用地址加 W 标志, ACK 已发出)
 - 向 I2C_CR 寄存器的 AAK 位写 1, 设置 I2C_CR 寄存器的 AAK 位
 - 向 I2C_CLR 中的 CLR_IFLG 位写 1, 开始接收数据
- I2C_STAT: 0x70 (已接收通用调用地址 (0x00); 已发出 ACK)
 - 向 I2C_CR 寄存器的 AAK 位写 1, 设置 I2C_CR 寄存器的 AAK 位
 - 向 I2C_CLR 中的 CLR_IFLG 位写 1, 开始接收数据
- I2C_STAT: 0x80 (前一次寻址使用自身从地址; 已接收数据字节; 已返回 ACK)
 - 读取 I2C_DATA 中收到的数据
 - 向 I2C_CR 寄存器的 AAK 位写 1, 设置 I2C_CR 寄存器的 AAK 位
 - 接收数据长度加 1
 - 向 I2C_CLR 中的 CLR_IFLG 位写 1
- I2C_STAT: 0x88 (前一次寻址使用自身从地址; 已接收数据字节; 已返回非 ACK)
 - 读取 I2C_DATA 中收到的数据
 - 向 I2C_CR 寄存器的 AAK 位写 1, 设置 I2C_CR 寄存器的 AAK 位
 - 接收数据长度加 1
 - 向 I2C_CLR 中的 CLR_IFLG 位写 1

- I2C_STAT: 0x90（前一次寻址使用通用调用地址；已接收数据；已返回 ACK）
 - 读取 I2C_DATA 中收到的数据
 - 向 I2C_CR 寄存器的 AAK 位写 1，设置 I2C_CR 寄存器的 AAK 位
 - 接收数据长度加 1
 - 向 I2C_CLR 中的 CLR_IFLG 位写 1
- I2C_STAT: 0x98（前一次寻址使用通用调用地址；已接收数据；已返回非 ACK）
 - 向 I2C_CR 寄存器的 AAK 位写 1，设置 I2C_CR 寄存器的 AAK 位
 - 向 I2C_CLR 中的 CLR_IFLG 位写 1
- I2C_STAT: 0xA0（当使用从接收/从发送模式中静态寻址时，接收到停止条件或重复起始条件）
 - 向 I2C_CR 寄存器的 AAK 位写 1，设置 I2C_CR 寄存器的 AAK 位
 - 向 I2C_CLR 中的 CLR_IFLG 位写 1

10.5.5 从机发送功能

- I2C_STAT: 0x60，已接收自身的 SLAVE 寄存器地址加 W 标志，ACK 已发出（10 位寻址出现）
 - 向 I2C_CR 寄存器的 AAK 位写 1，设置 I2C_CR 寄存器的 AAK 位
 - 向 I2C_DATA 寄存器写入待发送的数据
 - 向 I2C_CLR 中的 CLR_IFLG 位写 1
 - 发送数据长度加 1
- I2C_STAT: 0x68，丢失掉仲裁，并且以接收到自身的 SLAVE 寄存器地址加 W 标志，ACK 已发出（10 位寻址出现）
 - 向 I2C_CR 寄存器的 AAK 位写 1，设置 I2C_CR 寄存器的 AAK 位
 - 向 I2C_DATA 寄存器写入待发送的数据
 - 向 I2C_CLR 中的 CLR_IFLG 位写 1
 - 发送数据长度加 1

- I2C_STAT: 0x70, 已接收通用调用地址 (0x00); 已发出 ACK (10 位寻址出现)
 - 向 I2C_CR 寄存器的 AAK 位写 1, 设置 I2C_CR 寄存器的 AAK 位
 - 向 I2C_DATA 寄存器写入待发送的数据
 - 向 I2C_CLR 中的 CLR_IFLG 位写 1
 - 发送数据长度加 1
- I2C_STAT: 0xA0, 当使用从接收/从发送模式中静态寻址时, 接收到停止条件或重复起始条件 (10 位寻址出现)
 - 向 I2C_CR 寄存器的 AAK 位写 1, 设置 I2C_CR 寄存器的 AAK 位
 - 向 I2C_DATA 寄存器写入待发送的数据
 - 向 I2C_CLR 中的 CLR_IFLG 位写 1
 - 发送数据长度加 1
- I2C_STAT: 0xA8 (已接收自身的从地址加 R 标志; 已返回 ACK)
 - 向 I2C_CR 寄存器的 AAK 位写 1, 设置 I2C_CR 寄存器的 AAK 位
 - 向 I2C_DATA 寄存器写入待发送的数据
 - 向 I2C_CLR 中的 CLR_IFLG 位写 1
 - 发送数据长度加 1
- I2C_STAT: 0xB0 (用作从机时, 丢失掉仲裁, 并且已接收到通用调用地址加 R 标志, ACK 已发出)
 - 向 I2C_CR 寄存器的 AAK 位写 1, 设置 I2C_CR 寄存器的 AAK 位
 - 向 I2C_DATA 寄存器写入待发送的数据
 - 向 I2C_CLR 中的 CLR_IFLG 位写 1
 - 发送数据长度加 1
- I2C_STAT: 0xB8 (已发送数据; 已接收 ACK)
 - 向 I2C_CR 寄存器的 AAK 位写 1, 设置 I2C_CR 寄存器的 AAK 位
 - 向 I2C_DATA 寄存器写入待发送的数据
 - 向 I2C_CLR 中的 CLR_IFLG 位写 1

- 发送数据长度加 1
- I2C_STAT: 0xC8 (装入的数据字节已被发送; 已接收 ACK)
 - 向 I2C_CR 寄存器的 AAK 位写 1, 设置 I2C_CR 寄存器的 AAK 位
 - 向 I2C_DATA 寄存器写入待发送的数据
 - 向 I2C_CLR 中的 CLR_IFLG 位写 1
 - 发送数据长度加 1
- I2C_STAT: 0xC0 (已发送数据字节; 已接收非 ACK)
 - 向 I2C_CR 寄存器的 AAK 位写 1, 设置 I2C_CR 寄存器的 AAK 位
 - 向 I2C_CLR 中的 CLR_IFLG 位写 1

11 SPI0

11.1 概述

串行外设接口（Serial Peripheral Interface，SPI）是外部设备通过单线交换数据的串行同步通讯手段。芯片提供了一个 SPI0 接口模块，可配置为主设备或从设备，实现与外部的 SPI 通信。

11.2 主要特性

- 全双工或半双工单数据线串行同步收发
- 主从模式
- 可编程时钟极性和相位（支持模式 0、1、2、3）
- 可编程比特速率
- 从模式最大频率为 $F_{sys}/2$
- 传输结束中断标志
- 写冲突错标志
- 主模式错误检测、保护和中断标志
- 支持 DMA
- 8 个 byte fifo 深度

11.3 寄存器描述

SPI0 寄存器基地址：0x40000800

表 11-1: SPI0 寄存器列表

偏置	名称	描述
0x0	SPI0_CR	SPI0 配置寄存器
0x4	SPI0_CS0	SPI0 主模式控制寄存器 0
0x8	SPI0_CS1	SPI0 主模式控制寄存器 1

偏置	名称	描述
0x14	SPI0_OPCR	SPI0 过程控制寄存器
0x18	SPI0_IE	SPI0 中断控制寄存器
0x1C	SPI0_IF	SPI0 中断标志寄存器
0x20	SPI0_TXBUF	SPI0 发送缓存寄存器
0x24	SPI0_RXBUF	SPI0 接收缓存寄存器
0x28	SPI0_DMARXLEV	SPI0 DMA 接收设置寄存器
0x2c	SPI0_DMATXLEV	SPI0 DMA 发送设置寄存器

11.3.1 SPI0 配置寄存器 SPI0_CR (偏移: 00H)

比特	名称	属性	复位值	描述
31:14	RSV	-	-	保留
13	DMA_TX_EN	R/W	0	DMA Tx 使能: 0: 关闭 DMA TX 请求 1: 使能 DMA TX 请求
12	DMA_RX_EN	R/W	0	DMA Rx 使能: 0: 关闭 DMA RX 请求 1: 使能 DMA RX 请求
11	FLTEN	R/W	1	Slave 输入引脚滤波使能 (SSN/SCK/MOSI): 0: 不滤波 1: 使能 4ns 滤波
10	SSNM	R/W	0	Master 模式下 SSN 控制模式选择: 0: 每发送完 8bit 后 Master 保持 SSN 为低, 维持低电平时间由 WAIT 寄存器控制 1: 每发送完 8bit 后 Master 拉高 SSN, 维持高电平时间由 WAIT 寄存器控制
9	TXO_AC	R/W	1	TXONLY 硬件自动清零的使能: 0: 关闭 TXONLY 硬件自动清零 1: TXONLY 硬件自动清零有效, 软件使能 TXO 后, 等待发送完毕后, 硬件清零
8	TXO	R/W	0	TXONLY 控制位: 0: 关闭单发送模式 1: 启动 Master 的单发送模式

比特	名称	属性	复位值	描述
7	MSPA	R/W	0	Master Sampling Position Adjustment, Master 对 MISO 信号的采样位置调整, 用于高速通信时补偿 PCB 走线延迟: 0: 不调整 1: 采样点延迟半个 SCK 周期
6	SSPA	R/W	0	Slave Sending Position Adjustment , Slave MISO 发送位置调整: 0: 不调整 1: 提前半个 SCK 周期发送
5	MM	R/W	1	Master/Slave 模式选择: 0: Slave 模式 1: Master 模式
4:3	WAIT	R/W	0	Master 模式下, 每发完 8Bit 后加入至少 (1+WAIT)个 SCK cycle 等待时间再传输下一个 8Bit 的数据
2	RSV	-	-	-
1	SSNSEN	R/W	0	Master 模式下, 软件控制 SSN 使能: 0: Master 模式下 SSN 输出由硬件自动控制 1: Master 模式下 SSN 输出由软件控制
0	SPI0EN	R/W	0	SPI0 使能。采用关闭时钟的方式来关闭使能: 0: 关闭 SPI0, 清空发送接收缓存 1: 使能 SPI0

11.3.2 SPI0 主模式控制寄存器 0 SPI0_CS0 (偏移: 04H)

比特	名称	属性	复位值	描述
31:7	RSV	-	-	保留
6	SSN0	R/W	0	SPI0 主模式下, CS0 对应 Master 模式下, 如果 SSNSEN 为 1, 软件可以通过此位控制 SSN 输出电平: 0: SSN 输出高电平 1: SSN 输出低电平
5:3	BAUD0	R/W	001	SPI0 主模式下, CS0 对应 Master 模式波特率配置位: 000: fPCLK/2 001: fPCLK/4

比特	名称	属性	复位值	描述
				010: fPCLK/8 011: fPCLK/16 100: fPCLK/32 101: fPCLK/64 110: fPCLK/128 111: fPCLK/256 当通信正在进行的时候，不能修改这些位。
2	LSBF0	R/W	0	SPI0 主模式下，CS0 对应帧格式 (Frame format): 0: 先发送 MSB 1: 先发送 LSB 注：当通信在进行时不能改变该位的值。
1	CPHOL0	R/W	0	SPI0 主模式下，CS0 对应时钟极性选择。 0: 串行时钟停止在低电平 1: 串行时钟停止在高电平 注：当通信在进行时不能改变该位的值。 当 SSN 为低时不能改变该位的值
0	CPHA0	R/W	0	SPI0 主模式下，CS0 对应时钟相位选择： 0: 第一个时钟边沿是第一个捕捉边沿 1: 第二个时钟边沿是第一个捕捉边沿 注：当通信在进行时不能改变该位的值。

11.3.3 SPI0 主模式控制寄存器 1 SPI0_CS1（偏移：08H）

比特	名称	属性	复位值	描述
31:7	RSV	-	-	保留
6	SSN1	R/W	0	SPI0 主模式下，CS1 对应 Master 模式下，如果 SSNSEN 为 1，软件可以通过此位控制 SSN 输出电平： 0: SSN 输出高电平 1: SSN 输出低电平

比特	名称	属性	复位值	描述
5:3	BAUD1	R/W	001	SPI0 主模式下，CS1 对应 Master 模式波特率配置位： 000: fPCLK/2 001: fPCLK/4 010: fPCLK/8 011: fPCLK/16 100: fPCLK/32 101: fPCLK/64 110: fPCLK/128 111: fPCLK/256 当通信正在进行的时候，不能修改这些位
2	LSBF1	R/W	0	SPI0 主模式下，CS1 对应帧格式（Frame format）： 0: 先发送 MSB 1: 先发送 LSB 注：当通信在进行时不能改变该位的值
1	CPHOL1	R/W	0	SPI0 主模式下，CS1 对应时钟极性选择： 0: 串行时钟停止在低电平 1: 串行时钟停止在高电平 注：当通信在进行时不能改变该位的值。 当 SSN 为低时不能改变该位的值
0	CPHA1	R/W	0	SPI0 主模式下，CS1 对应时钟相位选择： 0: 第一个时钟边沿是第一个捕捉边沿 1: 第二个时钟边沿是第一个捕捉边沿 注：当通信在进行时不能改变该位的值

11.3.4 SPI0 过程控制寄存器 SPI0_OPCR（偏移：14H）

比特	名称	属性	复位值	描述
31:4	RSV	-	-	保留，读为 0
3	TXBFC	W1C	0	Transmit Buffer Clear 软件写 1 清除发送缓存，写 0 无效
2	RXBFC	W1C	0	Receive Buffer Clear 软件写 1 清除接收缓存，写 0 无效
1	MERRC	W1C	0	Master Error Clear 软件写 1 清除 SPIIF. MERR 寄存器

比特	名称	属性	复位值	描述
0	SERRC	W1C	0	Slave Error Clear 软件写 1 清除 SPIIF.SERR 寄存器

11.3.5 SPI0 中断控制寄存器 SPI0_IE (偏移: 18H)

比特	名称	属性	复位值	描述
31:9	RSV	-	-	保留，读为 0
8	RNFIE	R/W	0	Rx Fifo Full 中断使能 0: 不使能 1: 使能
7	TNFIE	R/W	0	Tx Fifo Not Full 中断使能 0: 不使能 1: 使能
6	MERRIE	R/W	0	Master Error 中断使能 0: 不使能 1: 使能
5	SERRIE	R/W	0	Slave Error 中断使能 0: 不使能 1: 使能
4	RXCOLIE	R/W	0	接收缓存溢出中断使能 0: 不使能 1: 使能 软件写 1 清零
3	TXCOLIE	R/W	0	发送缓存溢出中断使能 0: 不使能 1: 使能 软件写 1 清零
2	IDLEIE	R/W	0	SPI0 空闲标志中断使能 0: 不使能 1: 使能
1	TXBEIE	R/W	0	TX Buffer Empty 中断使能 0: 不使能 1: 使能
0	RXBFIE	R/W	0	RX Buffer 中断使能 0: 不使能 1: 使能

11.3.6 SPI0 中断标志寄存器 SPI0_IF (偏移: 1CH)

比特	名称	属性	复位值	描述
31:9	RSV	-	-	保留, 读为 0
8	RNF	R	0	Spi Rx Fifo Full: 0: SPI0 Rx Fifo 未 1: SPI0 Rx Fifo 满
7	TNF	R	1	Spi Tx Fifo Not Full: 0: SPI0 Tx Fifo 满 1: SPI0 Tx Fifo 未
6	MERR	R	0	Master Error 标志: 当 Master 下传输未 8 位 SSN 就被拉高时, MERR 置位
5	SERR	R	0	Slave Error 标志: 当 Slave 下传输未 8 位 SSN 就被拉高时, SERR 置位
4	RXCOL	R/W	0	接收缓存溢出 软件写 1 清零
3	TXCOL	R/W	0	发送缓存溢出 软件写 1 清零
2	IDLE	R	1	SPI0 空闲标志, 只读: 0: SPI0 传输进行中 1: SPI0 传输空闲
1	TXBE	R	1	TX Buffer Empty 标志位: 0: 发送缓存非空 1: 发送缓存空, 软件写 TXBUF 清零
0	RXBF	R	0	RX Buffer 非空标志位: 0: 接收缓存空 1: 接收缓存非空

11.3.7 SPI0 发送缓存寄存器 SPI0_TXBUF (偏移: 20H)

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留, 读为 0
7:0	TXBUF	W	0	SPI0 发送缓存, 发送 FIFO 入口地址。此 IP 一 共含有 8 个 Byte 的发送 FIFO, 写此地址, 将 要发送的数据写入 FIFO 中

11.3.8 SPI0 接收缓存寄存器 SPI0_RXBUF (偏移: 24H)

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留, 读为 0
7:0	RXBUF	R	0	SPI0 接收缓存, 接收 FIFO 入口地址。此 IP 一共含有 8 个 Byte 的接收 FIFO, 读此地址, 将接收的数据从 FIFO 中读取出来

11.3.9 SPI0 DMA 接收设置寄存器 SPI0_DMARXLEV (偏移: 28H)

比特	名称	属性	复位值	描述
31:3	RSV	-	-	保留, 读为 0
2:0	DMA_RX_LEV	R/W	0	SPI0 接收 FIFO DMA 请求设置。 当 RX FIFO 中的数据个数大于此寄存器设置值时, 产生 DMA RX 请求

11.3.10 SPI0 DMA 发送设置寄存器 SPI0_DMATXLEV (偏移: 2CH)

比特	名称	属性	复位值	描述
31:3	RSV	-	-	保留, 读为 0
2:0	DMA_TX_LEV	R/W	0	SPI0 发送 FIFO DMA 请求设置。 当 TX FIFO 中的数据个数小于此寄存器设置值时, 产生 DMA TX 请求

11.4 接口时序

为了兼容不同的 SPI 外设, SPI0 串行时钟的时序可以通过时钟相位选择位 (SPI0_CSx.CPHA) 和时钟极性选择位 (SPI0_CSx.CPOL) 设置产生 4 种不同组合。为保证数据正确传输, 主从器件的时序配置必需一致。

当处于从器件模式或 SPI0 系统使能位 (SPI0_CR.SPIEN) 位为 0 时, SPI0 的 SCK 引脚无串行时钟输出。

11.4.1 CPHA=0

CPHA=0 时，SPI0 模块在串行时钟的第一个跳变沿采样数据，即：

若 CPOL=1，在串行时钟的下降沿采样数据。

若 CPOL=0，在串行时钟的上升沿采样数据。如图 11-1 所示：

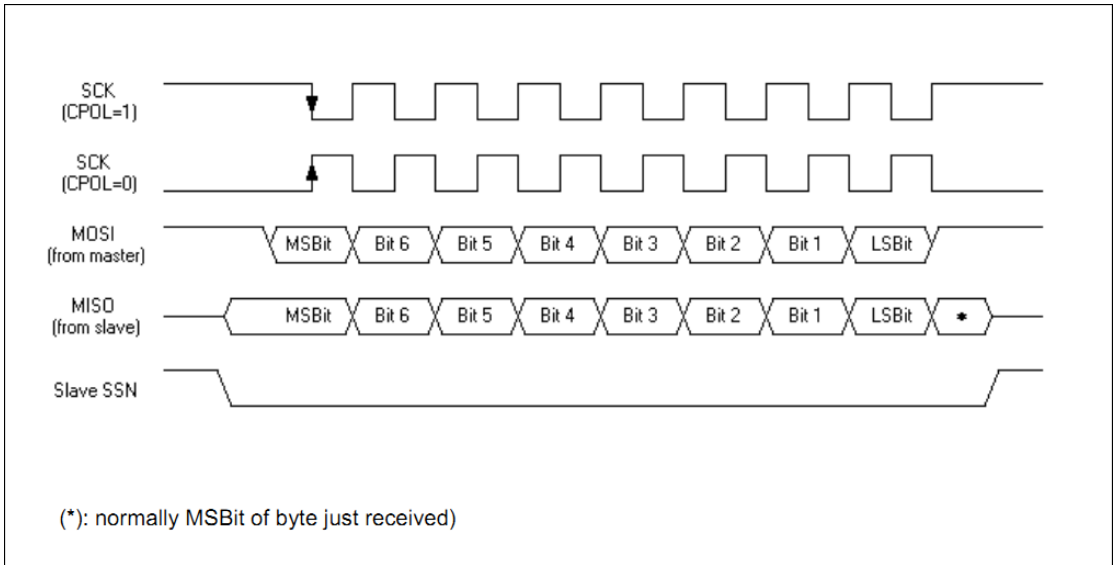


图 11-1：SPI0 数据/时钟时序图（CPHA=0）

11.4.2 CPHA=1

CPHA=1 时，SPI0 模块在串行时钟的第二个跳变沿采样数据，即：

若 CPOL=1，在串行时钟的上升沿采样数据。

若 CPOL=0，在串行时钟的下降沿采样数据。如图 11-2 所示：

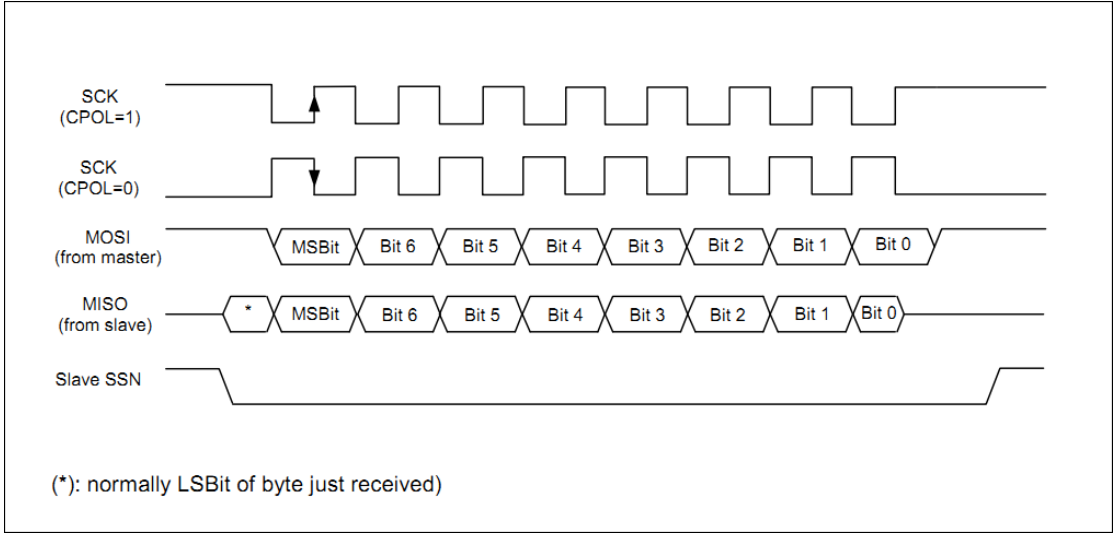


图 11-2: SPI0 数据/时钟时序图 (CPHA=1)

11.4.3 从器件 SSN

若 SPI0 为从器件，则 CPHA=0 时，SSN 引脚必须在每字节数据传输后拉高，以便可以拉低启动下一字节传输，并避免产生写冲突错误。如图 11-3 所示：

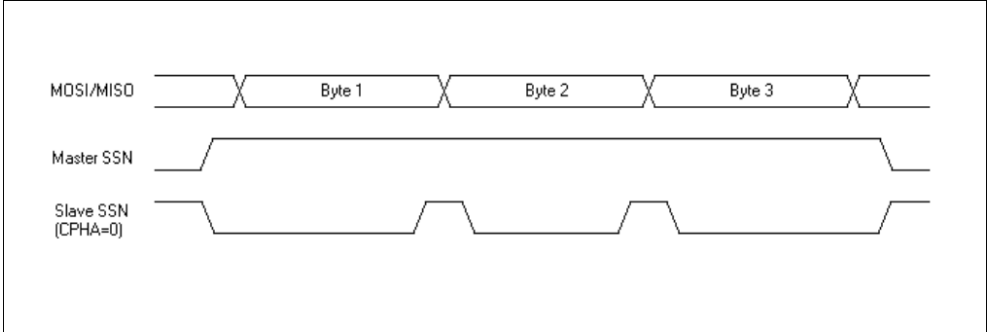


图 11-3: SPI0 SSN 时序图 (CPHA=0)

CPHA=1 时，从器件的 SSN 引脚可以在连续数据传输时一直为低，如图 11-4 所示：

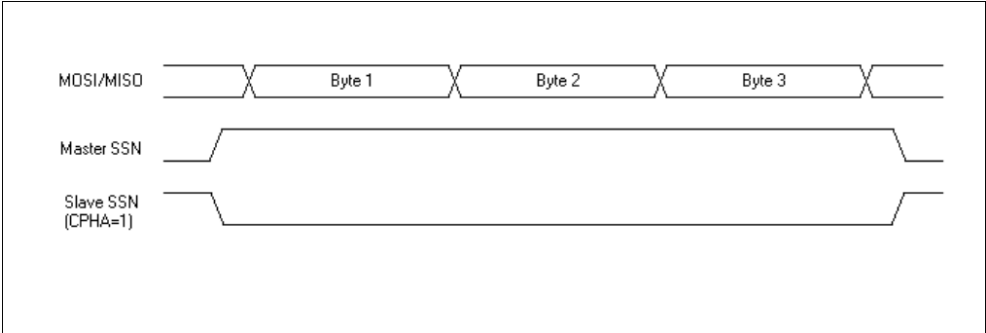


图 11-4: SPI0 SSN 时序图 (CPHA=1)

11.5 使用流程

表 11-2: SPI0 两种引脚搭配方式

信号描述功能位	SPI0 配置 1	SPI0 配置 2
CS	SPI0_CS0	SPI0_CS1
MISO	SPI0_MISO0	SPI0_MISO1
MOSI	SPI0_MOSI	SPI0_MOSI
CLK	SPI0_SCK	SPI0_SCK

注：引脚 SPI0_CS1 和 SPI0_MISO1 搭配使用，SPI0_CS0 与 SPI0_MISO0 搭配使用。

11.5.1 初始化程序

- 配置开启 SPI0 模块时钟与复位 PERI_RESET / PERI_CLKEN。
- 配置 SPI0CR.MM 位，设置主从模式。
- 配置 SPI0CR.SSNM 位，设置 SSN 控制模式。
- 配置 SPI0CR.SSNSSEN 位，设置 SSN 输出由软件还是硬件控制。
- 配置 SPI0CSx.SSNx 位和 SPI0CSx.LSBFx 位，设置 SSN 输出电平和帧格式。
- 配置 SPI0CSx.CPHAx 位和 SPI0CSx.CPOLx 位，以设置串行时钟相位和极性。
- 配置 SPI0CSx.BAUDx[2:0]位，以设置串行时钟波特率（若为从器件模式则不用设置，串行时钟速率由主器件决定）。需要时，配置中断，SPI0IE 和 SPI0IF 位。
- 配置 SPI0CR.SPIEN，使能 SPI0。

11.5.2 发送流程

- 主器件发送流程

配置 SPI0CSx.SSNx 拉低 SSN 引脚启动传输，配置 SPI0CSx.TXO 位为高，将数据写入 SPI0_TXBUF 寄存器，等待 SPI0IF.IDLE 置位发送完成，配置 SPI0CSx.TXO 位为低，传输完成后将 SSN 拉高。

- 从器件发送流程

配置 SPI0CSx.TXO 位为高，将数据写入 SPI0_TXBUF 寄存器，等待 SPI0IF.IDLE 置位发送完成，配置 SPI0CSx.TXO 位为低。

11.5.3 接收流程

- 主器件接收流程

配置 SPI0CSx.SSNx 拉低 SSN 引脚启动传输，将数据写入 SPI0_TXBUF 寄存器，等待 SPI0IF.RXBF 置位，读取 SPI0_RXBUF 寄存器数据完成数据接收，传输完成后将 SSN 拉高。

- 从器件接收流程

等待 SPI0IF.RXBF 置位，读取 SPI0_RXBUF 寄存器数据完成数据接收。

11.5.4 SPI0 DMA 发送流程

1. 配置SPI0 DMA发送设置寄存器DMA_SPI0TX_LEV，设置产生DMA RX请求的FIFO数据个数。
2. 使能 SPI0 CR.DMA_TX_EN 位，使能 DMA TX 请求。
3. 配置开启 DMA 模块时钟与复位 PERI_RESET / PERI_CLKEN。
4. 配置通道控制信息寄存器 DMA_CH_CTRL_Cx。
5. 根据实际应用配置数据位宽，传输模式（8 位位宽、内存到外设模式）。
6. 配置【目的外设】和【源外设】（目的外设为 SPI0 发送，源外设为 MEM），此位在用到外设的模式下起效。

7. 配置【目标地址】和【源地址】是否随数据传输递增（源地址递增、目标地址不变）。
8. 如需使用中断，则配置 DMA 中断指示寄存器 DMA_INT_STATUS，使能对应的通道中断。
9. 配置【源地址】、【目标地址】及【数据块尺寸】。

DMA_SRC_ADDR_Cx、DMA_DST_ADDR_Cx、DMA_CH_CTRL_Cx

10. 等待上述配置、以及相应的原地址和目标准备就绪，使能 DMA (DMAC_EN)。
11. 根据实际使用情况，检测 DMA 中断状态寄存器 DMA_INT_STATUS。

11.5.5 SPI0 DMA 接收流程

1. 配置SPI0 DMA发送设置寄存器DMA_SPI0RX_LEV，设置产生DMA RX请求的FIFO数据个数。
2. 使能 SPI0 CR.DMA_RX_EN 位，使能 DMA RX 请求。
3. 配置开启 DMA 模块时钟与复位 PERI_RESET / PERI_CLKEN。
4. 配置通道控制信息寄存器 DMA_CH_CTRL_Cx。
5. 根据实际应用配置数据位宽，传输模式（8 位位宽、内存到外设模式）。
6. 配置【目的外设】和【源外设】（目的外设为 MEM，源外设为 SPI0 接收），此位在用到外设的模式下起效。
7. 配置【目标地址】和【源地址】是否随数据传输递增（源地址递增、目标地址不变）。
8. 如需使用中断，则配置 DMA 中断指示寄存器 DMA_INT_STATUS，使能对应的通道中断。
9. 配置【源地址】和【目标地址】及【数据块尺寸】。

DMA_SRC_ADDR_Cx、DMA_DST_ADDR_Cx、DMA_CH_CTRL_Cx

10. 等待上述配置、以及相应的原地址和目标准备就绪，使能 DMA (DMAC_EN)。
11. 根据实际使用情况，检测 DMA 中断状态寄存器 DMA_INT_STATUS。

12 SPI1

12.1 概述

串行外设接口（Serial Peripheral Interface，SPI）是外部设备通过单线交换数据的串行同步通讯手段。芯片提供了一个 SPI1 接口模块，可配置为主设备或从设备，实现与外部的 SPI 通信。

12.2 主要特性

- 全双工或半双工单数据线串行同步收发
- 主从模式
- 可编程时钟极性和相位（支持模式 0、1、2、3）
- 可编程比特速率
- 从模式最大频率为 $F_{sys}/2$
- 传输结束中断标志
- 写冲突错标志
- 主模式错误检测、保护和中断标志
- 支持 DMA
- 8 个 byte FIFO 深度

12.3 寄存器描述

SPI1 寄存器基地址：0x40005800

表 12-1: SPI1 寄存器列表

偏置	名称	描述
0x0	SPI1_CR	SPI1 配置寄存器
0x4	SPI1_CS0	SPI1 主模式控制寄存器 0
0x14	SPI1_OPCR	SPI1 过程控制寄存器

偏置	名称	描述
0x18	SPI1_IE	SPI1 中断控制寄存器
0x1C	SPI1_IF	SPI1 中断标志寄存器
0x20	SPI1_TXBUF	SPI1 接收缓存寄存器
0x24	SPI1_RXBUF	SPI1 接收缓存寄存器
0x28	SPI1_DMARXLEV	SPI1 DMA 接收设置寄存器
0x2c	SPI1_DMATXLEV	SPI1 DMA 发送设置寄存器

12.3.1 SPI1 配置寄存器 SPI1_CR (偏移: 00H)

比特	名称	属性	复位值	描述
31:14	RSV	-	-	保留
13	DMA_TX_EN	R/W	0	Dma Tx 使能: 0: 关闭 DMA TX 请求 1: 使能 DMA TX 请求
12	DMA_RX_EN	R/W	0	Dma Rx 使能: 0: 关闭 DMA RX 请求 1: 使能 DMA RX 请求
11	FLTEN	R/W	1	Slave 输入引脚滤波使能 (SSN/SCK/MOSI): 0: 不滤波 1: 使能 4ns 滤波
10	SSNM	R/W	0	Master 模式下 SSN 控制模式选择: 0: 每发送完 8bit 后 Master 保持 SSN 为低, 维持低电平时间由 WAIT 寄存器控制 1: 每发送完 8bit 后 Master 拉高 SSN, 维持高电平时间由 WAIT 寄存器控制
9	TXO_AC	R/W	1	TXONLY 硬件自动清空的使能: 0: 关闭 TXONLY 硬件自动清零 1: TXONLY 硬件自动清零有效, 软件使能 TXO 后, 等待发送完毕后, 硬件清零
8	TXO	R/W	0	TXONLY 控制位: 0: 关闭单发送模式 1: 启动 Master 的单发送模式

比特	名称	属性	复位值	描述
7	MSPA	R/W	0	Master Sampling Position Adjustment, Master 对 MISO 信号的采样位置调整, 用于高速通信时补偿 PCB 走线延迟: 0: 不调整 1: 采样点延迟半个 SCK 周期
6	SSPA	R/W	0	Slave Sending Position Adjustment , Slave MISO 发送位置调整: 0: 不调整 1: 提前半个 SCK 周期发送
5	MM	R/W	1	Master/Slave 模式选择: 0: Slave 模式 1: Master 模式
4:3	WAIT	R/W	0	Master 模式下, 每发完 8Bit 后加入至少 (1+WAIT)个 SCK cycle 等待时间再传输下一个 8Bit 的数据
2	RSV	-	-	-
1	SSNSEN	R/W	0	Master 模式下, 软件控制 SSN 使能: 0: Master 模式下 SSN 输出由硬件自动控制 1: Master 模式下 SSN 输出由软件控制
0	SPI1EN	R/W	0	SPI1 使能。采用关闭时钟的方式来关闭使能: 0: 关闭 SPI1, 清空发送接收缓存 1: 使能 SPI1

12.3.2 SPI1 主模式控制寄存器 0 SPI1_CS0 (偏移: 04H)

比特	名称	属性	复位值	描述
31:7	RSV	-	-	保留
6	SSN0	R/W	0	SPI1 主模式下, CS0 对应 Master 模式下, 如果 SSNSEN 为 1, 软件可以通过此位控制 SSN 输出电平: 0: SSN 输出高电平 1: SSN 输出低电平
5:3	BAUD0	R/W	001	SPI1 主模式下, CS0 对应 Master 模式波特率配置位: 000: fPCLK/2 001: fPCLK/4

比特	名称	属性	复位值	描述
				010: fPCLK/8 011: fPCLK/16 100: fPCLK/32 101: fPCLK/64 110: fPCLK/128 111: fPCLK/256 当通信正在进行的时候，不能修改这些位
2	LSBF0	R/W	0	SPI1 主模式下，CS0 对应帧格式 (Frame format): 0: 先发送 MSB 1: 先发送 LSB 注：当通信在进行时不能改变该位的值
1	CPHOL0	R/W	0	SPI1 主模式下，CS0 对应时钟极性选择： 0: 串行时钟停止在低电平 1: 串行时钟停止在高电平 注：当通信在进行时不能改变该位的值 当 SSN 为低时不能改变该位的值
0	CPHA0	R/W	0	SPI1 主模式下，CS0 对应时钟相位选择： 0: 第一个时钟边沿是第一个捕捉边沿 1: 第二个时钟边沿是第一个捕捉边沿 注：当通信在进行时不能改变该位的值

12.3.3 SPI1 过程控制寄存器 SPI1_OPCR (偏移: 14H)

比特	名称	属性	复位值	描述
31:4	RSV	-	-	保留，读为 0
3	TXBFC	W1C	0	Transmit Buffer Clear，软件写 1 清除发送缓存，写 0 无效
2	RXBFC	W1C	0	Receive Buffer Clear，软件写 1 清除接收缓存，写 0 无效
1	MERRC	W1C	0	Master Error Clear，软件写 1 清除 SPIIF. MERR 寄存器

12.3.4 SPI1 中断控制寄存器 SPI1_IE (偏移: 18H)

比特	名称	属性	复位值	描述
31:9	RSV	-	-	保留, 读为 0
8	RNFIE	R/W	0	Rx Fifo Full 中断使能
7	TNFIE	R/W	0	Tx Fifo Not Full 中断使能
6	MERRIE	R/W	0	Master Error 中断使能
5	SERRIE	R/W	0	Slave Error 中断使能
4	RXCOLIE	R/W	0	接收缓存溢出中断使能, 软件写 1 清零
3	TXCOLIE	R/W	0	发送缓存溢出中断使能, 软件写 1 清零
2	IDLEIE	R/W	0	SPI1 空闲标志中断使能
1	TXBEIE	R/W	0	TX Buffer Empty 中断使能
0	RXBFIE	R/W	0	RX Buffer 中断使能

12.3.5 SPI1 中断标志寄存器 SPI1_IF (偏移: 1CH)

比特	名称	属性	复位值	描述
31:9	RSV	-	-	保留, 读为 0
8	RNF	R	0	Spi Rx Fifo Full: 0: SPI1 Rx FIFO 未满 1: SPI1 Rx FIFO 满
7	TNF	R	1	Spi Tx FIFO Not Full: 0: SPI1 Tx FIFO 满 1: SPI1 Tx FIFO 未满
6	MERR	R	0	Master Error 标志: 当 Master 下传输未满 8 位 SSN 就被拉高时, MERR 置位
5	SERR	R	0	Slave Error 标志, 当 Slave 下传输未满 8 位 SSN 就被拉高时, SERR 置位
4	RXCOL	R/W	0	接收缓存溢出, 软件写 1 清零
3	TXCOL	R/W	0	发送缓存溢出, 软件写 1 清零
2	IDLE	R	1	SPI1 空闲标志, 只读: 0: SPI1 传输进行中 1: SPI1 传输空闲
1	TXBE	R	1	TX Buffer Empty 标志位: 0: 发送缓存非空 1: 发送缓存空, 软件写 TXBUF 清零

比特	名称	属性	复位值	描述
0	RXBF	R	0	RX Buffer 非空标志位: 0: 接收缓存空 1: 接收缓存非空

12.3.6 SPI1 发送缓存寄存器 SPI1_TXBUF (偏移: 20H)

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留, 读为 0
7:0	TXBUF	W	0	SPI1 发送缓存, 发送 FIFO 入口地址。此 IP 一共含有 8 个 Byte 的发送 FIFO, 写此地址, 将要发送的数据写入 FIFO 中

12.3.7 SPI1 接收缓存寄存器 SPI1_RXBUF (偏移: 24H)

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留, 读为 0
7:0	RXBUF	R	0	SPI1 接收缓存, 接收 FIFO 入口地址。此 IP 一共含有 8 个 Byte 的接收 FIFO, 读此地址, 将接收的数据从 FIFO 中读取出来

12.3.8 SPI1 DMA 接收设置寄存器 SPI1_DMARXLEV (偏移: 28H)

比特	名称	属性	复位值	描述
31:3	RSV	-	-	保留, 读为 0
2:0	DMA_RX_LEV	R/W	0	SPI1 接收 FIFO DMA 请求设置。 当 RX FIFO 中的数据个数大于此寄存器设置值时, 产生 DMA RX 请求

12.3.9 SPI1 DMA 发送设置寄存器 SPI1_DMATXLEV (偏移: 2CH)

比特	名称	属性	复位值	描述
31:3	RSV	-	-	保留, 读为 0
2:0	DMA_TX_LEV	R/W	0	SPI1 发送 FIFO DMA 请求设置。 当 TX FIFO 中的数据个数小于此寄存器设置值时, 产生 DMA TX 请求

12.4 接口时序

为了兼容不同的 SPI 外设，SPI1 串行时钟的时序可以通过时钟相位选择位（SPI1_CSx.CPHA）和时钟极性选择位(SPI1_CSx.CPOL)设置产生 4 种不同组合。为保证数据正确传输，主从器件的时序配置必需一致。

当处于从器件模式或 SPI1 系统使能位（SPI1_CR.SPIEN）位为 0 时，SPI1 的 SCK 引脚无串行时钟输出。

12.4.1 CPHA=0

CPHA=0 时，SPI1 模块在串行时钟的第一个跳变沿采样数据，即：
若 CPOL=1，在串行时钟的下降沿采样数据。
若 CPOL=0，在串行时钟的上升沿采样数据。如图 12-1 所示：

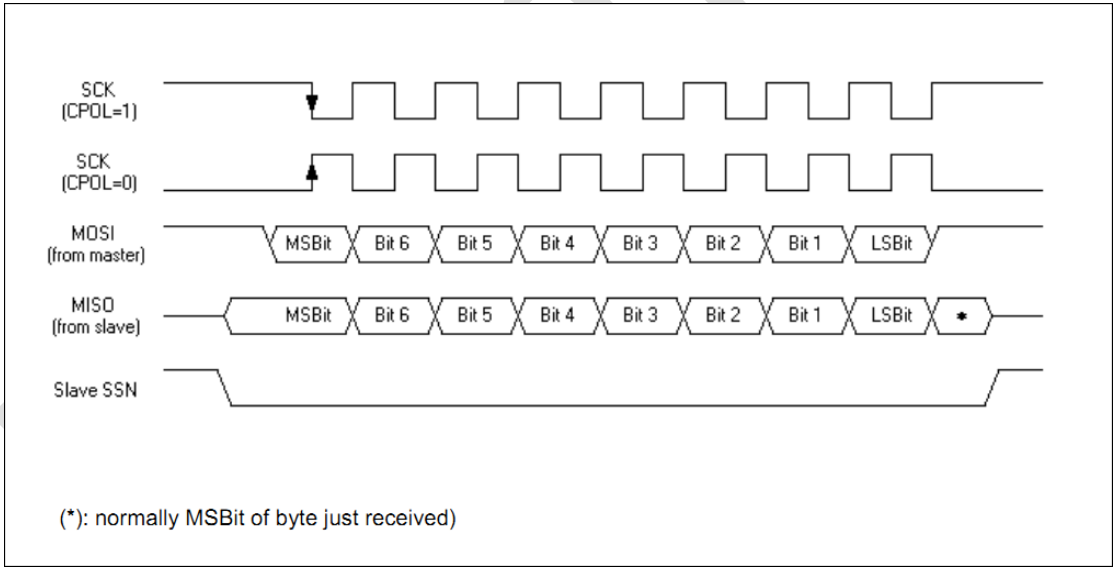


图 12-1：SPI1 数据/时钟时序图（CPHA=0）

12.4.2 CPHA=1

CPHA=1 时，SPI1 模块在串行时钟的第二个跳变沿采样数据，即：
若 CPOL=1，在串行时钟的上升沿采样数据。

若 CPOL=0，在串行时钟的下降沿采样数据。如图 12-2 所示：

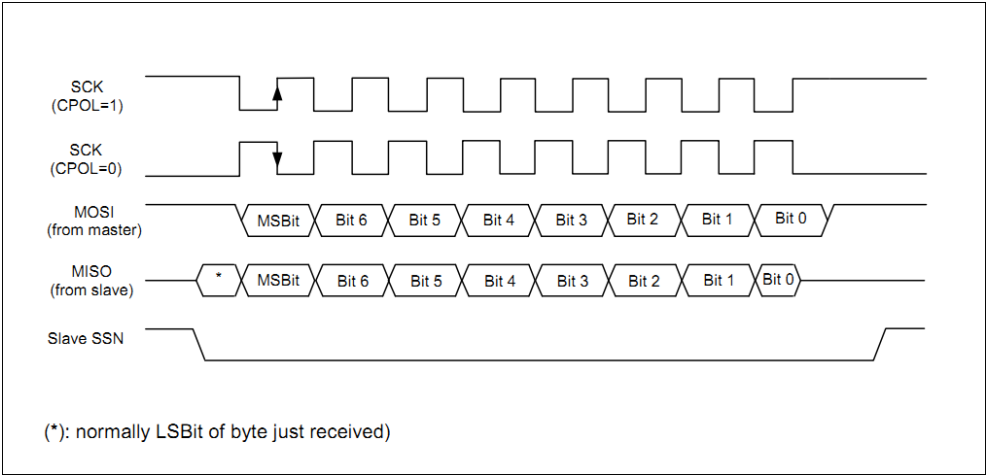


图 12-2：SPI1 数据/时钟时序图（CPHA=1）

12.4.3 从器件 SSN

若 SPI1 为从器件，则 CPHA=0 时，SSN 引脚必须在每字节数据传输后拉高，以便可以拉低启动下一字节传输，并避免产生写冲突错误。如图 12-3 所示：

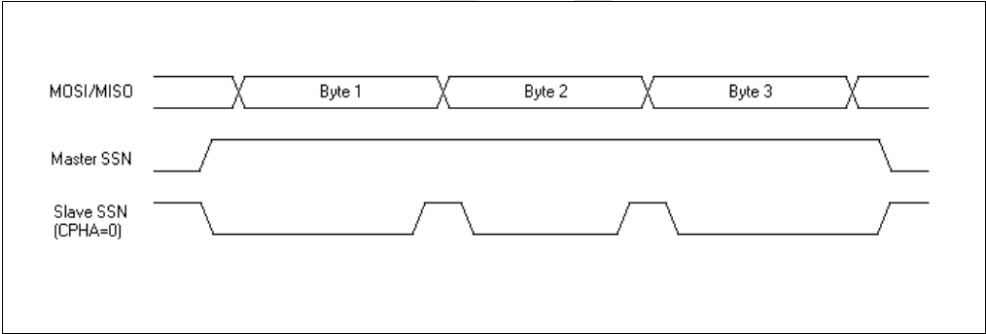


图 12-3：SPI1 SSN 时序图（CPHA=0）

CPHA=1 时，从器件的 SSN 引脚可以在连续数据传输时一直为低，如图 12-4 所示：

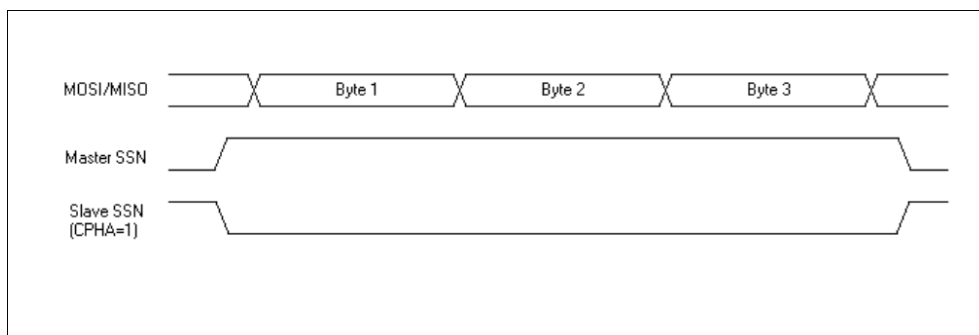


图 12-4: SPI1 SSN 时序图 (CPHA=1)

12.5 使用流程

12.5.1 初始化程序

配置开启 SPI1 模块时钟与复位 PERI_RESET / PERI_CLKEN。

1. 配置 SPI1CR.MM 位，设置主从模式。
2. 配置 SPI1CR.SSNM 位，设置 SSN 控制模式。
3. 配置 SPI1CR.SSENSEN 位，设置 SSN 输出由软件还是硬件控制。
4. 配置 SPI1CSx.SSNx 位和 SPI1CSx.LSBFx 位，设置 SSN 输出电平和帧格式。
5. 配置 SPI1CSx.CPHAx 位和 SPI1CSx.CPOLx 位，以设置串行时钟相位和极性。
6. 配置 SPI1CSx.BAUDx[2:0]位，以设置串行时钟波特率（若为从器件模式则不用设置，串行时钟速率由主器件决定）。需要时，配置中断，SPI1IE 和 SPI1IF 位。
7. 配置 SPI1CR.SPIEN，使能 SPI1。

12.5.2 发送流程

● 主器件发送流程

配置 SPI1CSx.SSNx 拉低 SSN 引脚启动传输，配置 SPI1CSx.TXO 位为高，将数据写入 SPI1_TXBUF 寄存器，等待 SPI1IF.IDLE 置位发送完成，配置 SPI1CSx.TXO 位为低，传输完成后将 SSN 拉高。

- 从器件发送流程

配置 SPI1CSx.TXO 位为高, 将数据写入 SPI1_TXBUF 寄存器, 等待 SPI1IF.IDLE 置位发送完成, 配置 SPI1CSx.TXO 位为低。

12.5.3 接收流程

- 主器件接收流程

配置 SPI1CSx.SSNx 拉低 SSN 引脚启动传输, 将数据写入 SPI1_TXBUF 寄存器, 等待 SPI1IF.RXBF 置位, 读取 SPI1_RXBUF 寄存器数据完成数据接收, 传输完成后将 SSN 拉高。

- 从器件接收流程

等待 SPI1IF.RXBF 置位, 读取 SPI1_RXBUF 寄存器数据完成数据接收。

12.5.4 SPI1 DMA 发送流程

1. 配置 SPI1 DMA 发送设置寄存器 DMA_SPI1TX_LEV, 设置产生 DMA RX 请求的 FIFO 数据个数。
2. 使能 SPI1 CR.DMA_TX_EN 位, 使能 DMA TX 请求。
3. 配置开启 DMA 模块时钟与复位 PERI_RESET / PERI_CLKEN。
4. 配置通道控制信息寄存器 DMA_CH_CTRL_Cx。
5. 根据实际应用配置数据位宽, 传输模式 (8 位位宽、内存到外设模式)。
6. 配置【目的外设】和【源外设】(目的外设为 SPI1 发送, 源外设为 MEM), 此位在用到外设的模式下起效。
7. 配置【目标地址】和【源地址】是否随数据传输递增 (源地址递增、目标地址不变)。
8. 如需使用中断, 则配置 DMA 中断指示寄存器 DMA_INT_STATUS, 使能对应的通道中断。
9. 配置【源地址】和【目标地址】及【数据块尺寸】。
DMA_SRC_ADDR_Cx、DMA_DST_ADDR_Cx、DMA_CH_CTRL_Cx
10. 等待上述配置、以及相应的原地址和目标准备就绪, 使能 DMA (DMAC_EN)。
11. 根据实际使用情况, 检测 DMA 中断状态寄存器 DMA_INT_STATUS。

12.5.5 SPI1 DMA 接收流程

1. 配置SPI1 DMA发送设置寄存器DMA_SPI1RX_LEV，设置产生DMA RX请求的FIFO数据个数。
2. 使能 SPI1 CR.DMA_RX_EN 位，使能 DMA RX 请求。
3. 配置开启 DMA 模块时钟与复位 PERI_RESET / PERI_CLKEN。
4. 配置通道控制信息寄存器 DMA_CH_CTRL_Cx。
5. 根据实际应用配置数据位宽，传输模式（8 位位宽、内存到外设模式）；
6. 配置【目的外设】和【源外设】（目的外设为 MEM，源外设为 SPI1 接收），此位在用到外设的模式下起效。
7. 配置【目标地址】和【源地址】是否随数据传输递增（源地址递增、目标地址不变）。
8. 如需使用中断，则配置 DMA 中断指示寄存器 DMA_INT_STATUS，使能对应的通道中断。
9. 配置【源地址】和【目标地址】及【数据块尺寸】。
DMA_SRC_ADDR_Cx、DMA_DST_ADDR_Cx、DMA_CH_CTRL_Cx
10. 等待上述配置、以及相应的原地址和目标准备就绪，使能 DMA（DMAC_EN）。
11. 根据实际使用情况，检测 DMA 中断状态寄存器 DMA_INT_STATUS。

13 QSPI

13.1 概述

QSPI 是一种专用的通信接口，连接单、双或四线 SPI Flash 存储介质。该接口支持取址功能。

13.2 主要特性

- 允许 8、16 和 32 位数据访问
- 支持 CPOL/CPHA 控制
- 支持外扩 SPI FLASH 读写控制，支持取指令操作
- 支持指令和参数，完全可编程帧格式
- 写指令操作完成以及发生访问错误时可产生中断
- 支持普通 SPI 读写

13.3 寄存器描述

QSPI 寄存器基地址：0x01100400

表 13-1: QSPI 寄存器列表

偏置	名称	描述
0x00	SPI_CTRL	SPI 配置寄存器
0x04	SPI_BAUD	SPI 波特率配置寄存器
0x08	SPI_MEMO_ACC	SPI 存储器访问配置寄存器
0x0C	SPI_CMD	SPI 命令寄存器
0x10	SPI_PARA_R	SPI 读参数寄存器
0x14	SPI_PARA_W	SPI 写参数寄存器
0x18	SPI_PGT_SET	SPI 擦写时间设置寄存器
0x1C	SPI_INTEN	SPI 中断使能控制寄存器
0x20	SPI_INTUS	SPI 中断状态寄存器
0x24	SPI_STATUS	SPI 状态寄存器

偏置	名称	描述
0x28	SPI_RXBUF	SPI 接收缓存寄存器

13.3.1 配置寄存器 SPI_CTRL（偏移：00H）

比特	名称	属性	复位值	描述
31:12	RSV	-	-	保留
11	CPU_HOLD	R/W	0	此寄存器设置在 QSPI 写数据过程中，是否停掉 CPU。 0: SPI_PGT_SET 寄存器设置的时间内，CPU 正常运行 1: SPI_PGT_SET 寄存器设置的时间内，CPU 停止运行
10:5	HALF_US	R/W	001111	根据系统时钟频率，设置时间标尺；此寄存器的值为：(AHB 时钟/2) -1。 例如系统时钟频率为 32M, 那么此寄存器应该设置为 15
4:3	X_MODE	R/W	10	SPI 配置： 00: 单数据线 SPI 协议传输；（标准 SPI 协议） 01: 2 数据线 SPI 协议传输 10: 4 数据线 SPI 协议传输 11: 保留
2	LSB	R/W	0	MSB/LSB 在前选择： 0: SPI 总线传输中 MSB 在前 1: SPI 总线传输中 LSB 在前
1	CPOL	R/W	0	CSN 对应时钟极性选择。 0: 串行时钟停止在低电平 1: 串行时钟停止在高电平
0	CPHA	R/W	0	CSN 对应时钟相位选择： 1: 第二个时钟边沿是第一个捕捉边沿 0: 第一个时钟边沿是第一个捕捉边沿

13.3.2 SPI 波特率配置寄存器 SPI_BAUD（偏移：04H）

比特	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:8	BAUD1	R/W	0	SPI 串行时钟二级分频因子

比特	名称	属性	复位值	描述
7:0	BAUD0	R/W	10	SPI 串行时钟一级分频因子。 该分频因子必须是 2 到 254 之间的偶数（包括 2 和 254）。第 0 位返回值总是为 0 $F = \text{AHB 时钟} / \{\text{BAUD1}, \text{BAUD0}\}$

13.3.3 SPI 存储器访问配置寄存器 SPI_MEMO_ACC (偏移: 08H)

比特	名称	属性	复位值	描述
31:21	RSV	-	-	保留
20	WRITE_NORMAL	R/W	0	普通 SPI，写模式使能： 0：关闭普通 SPI 写模式 1：使能普通 SPI 写模式，不发送地址和数据字段
19	READ_NORMAL	R/W	0	普通 SPI，读模式使能： 0：关闭普通 SPI 读模式 1：使能普通 SPI 读模式，不发送地址和数据字段
18	ADDR_LINE_SIZE_R	R/W	0	读操作： 0：Address 使用单线模式发送 1：Address 发送位宽等于 X_MODE 的设置值
17:16	Para_No2_R	R/W	01	读操作： 00：不使用参数 2 01：参数 2 长度为 1 个 byte 10：参数 2 长度为 2 个 byte 11：保留
15	Para_No1_R	R/W	1	读操作： 0：不使用参数 1 1：使用参数 1，参数 1 为一个 byte
14	Para_Ord2_R	R/W	0	读操作决定在发送地址前后加入参数 2 1：在地址后发送 0：在地址前发送
13	Para_Ord1_R	R/W	1	读操作决定在发送地址前后加入参数 1 0：在地址前发送 1：在地址后发送
12	WRITE_NO_DATA	R/W	1	0：写操作含有数据字段 1：写操作没有数据字段

比特	名称	属性	复位值	描述
11	READ_NO_ADDR	R/W	0	0: 读操作发送 Address 字段 1: 读操作不发送 ADDRESS 字段
10	WRITE_NO_ADDR	R/W	1	0: 写操作发送 ADDR 字段 1: 写操作不发送 ADDR 字段
9	ADDR_LINE_SIZE_W	R/W	1	擦写操作: 0: Address 使用单线模式发送 1: Address 发送位宽等于 X mode 的设置值
8:7	ADDR_WIDTH	R/W	10	地址位数配置: 00: 地址位宽为 1 个字节 01: 地址位宽为 2 个字节 10: 地址位宽为 3 个字节 11: 保留
6:5	Para_No2_W	R/W	00	擦写操作: 00: 不使用参数 2 01: 参数 2 长度为 1 个 byte 10: 参数 2 长度为 2 个 byte 11: 保留
4	Para_No1_W	R/W	0	擦写操作: 0: 不使用参数 1 1: 使用参数 1, 参数 1 为一个 byte
3	CON_RD_EN	R/W	0	连读使能位。 0: 不使能连读 1: 使能连读
2	Para_Ord2_W	R/W	1	擦写操作决定在发送地址前后加入参数 2 0: 在地址前发送 1: 在地址后发送
1	Para_Ord1_W	R/W	1	擦写操作决定在发送地址前后加入参数 1 0: 在地址前发送 1: 在地址后发送
0	SPI_EN	R/W	1	SPI 访问使能位: 0: 禁止 SPI 1: 使能 SPI

13.3.4 SPI 命令寄存器 SPI_CMD (偏移: 0Ch)

比特	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:8	WR_CMD	R/W	0	存放写 SPI 存储器写的指令
7:0	RD_CMD	R/W	0	存放读 SPI 存储器的读指令

13.3.5 SPI 读参数寄存器 SPI_PARA_R (偏移: 10H)

比特	名称	属性	复位值	描述
31:24	RSV	-	-	保留
23:8	SPI_PARA_R2	R/W	0	SPI 参数 2 寄存器, 内部存储参数 2 的值
7:0	SPI_PARA_R1	R/W	1	SPI 参数 1 寄存器, 内部存储参数 1 的值

13.3.6 SPI 写参数寄存器 SPI_PARA_W (偏移: 14H)

比特	名称	属性	复位值	描述
31:24	RSV	-	-	保留
23:8	SPI_PARA_W2	R/W	0	SPI 写参数 2 寄存器, 内部存储参数 2 的值
7:0	SPI_PARA_W1	R/W	1	SPI 写参数 1 寄存器, 内部存储参数 1 的值

13.3.7 SPI 擦写时间设置寄存器 SPI_PGT_SET (偏移: 18H)

比特	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	SPI_PGT_SET	R/W	0	设置 QSPI 发出写操作命令后, 等待时间; 每个时间单位为 0.5μs

13.3.8 SPI 中断使能控制寄存器 SPI_INTEN (偏移: 1CH)

比特	名称	属性	复位值	描述
31:2	RSV	-	-	保留
1	OP_ERREN	R/W	0	错误操作中断使能位: 0: 错误操作中断禁止 1: 错误操作中断使能

比特	名称	属性	复位值	描述
0	INTEN	R/W	0	完成中断使能寄存器： 0：中断禁止 1：中断使能

13.3.9 SPI 中断状态寄存器 SPI_INTUS (偏移：20H)

比特	名称	属性	复位值	描述
31:2	RSV	-	-	保留
1	OP_ERR_STATUS	R/W	0	错误操作中断状态寄存器。在擦写时间未结束时对另一内存空间进行擦写，触发错误中断 0：错误中断未产生 1：错误中断产生 写 1 清 0
0	INT_STATUS	R/W	0	擦写指令完成中断状态寄存器，SPI_PGT_SET 设置时间到达后，此位为 1 0：中断未产生 1：中断产生 写 1 清 0

13.3.10 SPI 状态寄存器 SPI_STATUS (偏移：24H)

比特	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	SPI_READY	R/W	1	状态寄存器： 0：QSPI 模块忙，正在进行 FLASH 操作 1：QSPI 模块空闲

13.3.11 SPI 接收缓存寄存器 SPI_RXBUF (偏移：28H)

比特	名称	属性	复位值	描述
31:0	RXBUF	R	0	SPI 接收缓存, 支持 8 位/16 位/32 位 SPI 数据帧

13.4 使用流程

13.4.1 QSPI 读 FLASH

1. 开启 QSPI 模块时钟，释放复位。
2. 配置引脚复用，打开引脚的输入使能。
3. REG_QSPI_BAUD 寄存器设置传输速率，根据系统时钟设置 REG_QSPI_CTRL 寄存器的 BIT5-10 的时间标尺值。
4. 根据 SPI 通信模式设置 REG_QSPI_CTRL 寄存器的 BIT0-1 的时钟极性和时钟相位。
5. 设置 REG_QSPI_MEMO_ACC 寄存器的 BIT0，使能 SPI。
6. 根据发送指令所需要的模式设置 REG_QSPI_CTRL 寄存器的 BIT3-4 选择单/双/四线通信模式。
7. 根据发送指令所需要的格式设置 REG_QSPI_MEMO_ACC 寄存器，选择是否发送地址/数据，是否发送参数。
8. 若需要发送参数，则设置 REG_QSPI_PARA_R 寄存器。
9. 写 REG_QSPI_CMD 寄存器，写入要发送的指令号。
10. 读取 FLASH 地址中的数据。

13.4.2 QSPI 写 FLASH

1. 开启 QSPI 模块时钟，释放复位。
2. 配置引脚复用，打开引脚的输入使能。
3. REG_QSPI_BAUD 寄存器设置传输速率，根据系统时钟设置 REG_QSPI_CTRL 寄存器的 BIT5-10 的时间标尺值。
4. 根据 SPI 通信模式设置 REG_QSPI_CTRL 寄存器的 BIT0-1 的时钟极性和时钟相位。
5. 设置 REG_QSPI_MEMO_ACC 寄存器的 BIT0，使能 SPI。
6. 根据发送指令所需要的模式设置 REG_QSPI_CTRL 寄存器的 BIT3-4 选择单/双/四线通信模式。
7. 根据发送指令所需要的格式设置 REG_QSPI_MEMO_ACC 寄存器，选择是否发送地址/数据，是否发送参数。

否发送参数。

8. 若需要发送参数，则设置 REG_QSPI_PARA_W 寄存器。
9. 写 REG_QSPI_CMD 寄存器，写入要发送的指令号。
10. 设置 REG_QSPI_PGT_SET 寄存器，写入擦写所需要等待的时间。
11. 向 FLASH 地址中写入数据。
12. 等待写入完成（可通过向 FLASH 发送询问是否 busy 的指令来获取 FLASH 状态，也可通过设置 REG_QSPI_PGT_SET 寄存器，硬件在发送完指令后等待所设置的事件后，触发完成中断）。

13.4.3 普通 SPI 读写

1. 设置 REG_QSPI_MEMO_ACC 寄存器的 BIT18-19，使能普通 SPI 读写模式。
2. SPI 默认为主机模式，通过向 QSPI FLASH 地址中写入数据，写入要发送的 SPI 数据（可写入 QSPI FLASH 地址范围内地址(0x0001_0000-0x0101_0000)，此地址将不会发出，硬件只会发出数据，写入的地址需要满足地址对其规则，推荐地址字对齐）。
3. 读取 REG_QSPI_RXBUF 寄存器接收 SPI 数据。

14 Cache

14.1 概述

Cache 处于 QSPI 和 CortexM0 之间，实现对 QSPI 中存储的指令进行预取指，用于提高 QSPI 取指速度，提升系统性能。

此 Cache 在系统中的位置如图 14-1 所示：

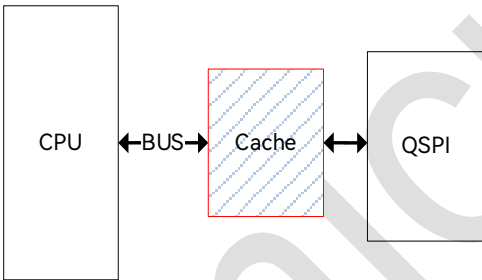


图 14-1：Cache 位置示意图

14.2 主要特性

- Data Space 为 2K 字节大小
- Data SRAM 在 Cache 未使能时可以当系统 SRAM 用，支持 word、half word 和 byte 读写

14.3 寄存器描述

Cache 寄存器基地址：0x20004400

表 14-1：Cache 寄存器列表

偏置	名称	描述
0x00	CACHE_CR	Cache 控制寄存器

14.3.1 控制寄存器 CACHE_CR (偏移: 00H)

比特	名称	属性	复位值	描述
31:2	RSV	-	-	保留
1	CACHE_CLR	W	0	写 1 清除 Cache 中所有缓存的内容, 每次 Cache 使能时, 均需要向此位写 1 0: 不清除Cache中缓存的数据 1: 清除 Cache 中的所有缓存数据 读该寄存器, 可清除 COUNTFLAG标志
0	CACHE_EN	R/W	0	Cache 使能位: 0: Cache 禁止 1: Cache 使能

14.4 使用流程

1. 使能 Cache 时, 需要向 CACHE_EN 和 CACHE_CLR 位写 1。
2. 在 CACHE_EN 不为 1, 即 Cache 未使能时, Cache Data Space 可以作为系统 SRAM 使用。在 CACHE_EN 为 1 后, 读写 Cache Data Space 将发生数据错误。

15 CAN

15.1 概述

CAN（Controller Area Network）控制器可以用于汽车电子和工业控制领域，支持 CAN2.0A/B 协议。

15.2 主要特性

- 具有实时性强、传输距离较远、抗电磁干扰能力强、成本低等优点
- 采用双线串行通信方式，检错能力强，可在高噪声干扰环境中工作
- 具有优先权和仲裁功能，多个控制模块通过CAN 控制器挂到CAN-bus上，形成多主机局部网络
- 可根据报文的 ID 决定接收或屏蔽该报文
- 可靠的错误处理和检错机制
- 发送的信息遭到破坏后，可自动重发
- 节点在错误严重的情况下具有自动退出总线的功能
- 报文不包含源地址或目标地址，仅用标志符来指示功能信息、优先级信息

15.3 寄存器描述

CAN 寄存器基地址：0x40005C00

表 15-1: CAN 寄存器列表

偏置	名称	描述
0x00	CAN_MR	模式寄存器
0x04	CAN_CMR	指令寄存器
0x08	CAN_SR	控制寄存器
0x0C	CAN_ISR	中断状态寄存器
0x10	CAN_IMR	中断使能寄存器
0x14	CAN_RMC	接收 FIFO 数据个数寄存器

偏置	名称	描述
0x18	CAN_BTR0	总线时序寄存器 0
0x1C	CAN_BTR1	总线时序寄存器 1
0x20	CAN_TXBUF	发送缓存寄存器
0x24	CAN_RXBUF	接收缓存寄存器
0x28	CAN_ACR	接收过滤匹配寄存器
0x2C	CAN_AMR	接收过滤屏蔽寄存器
0x30	CAN_ECC	错误码捕捉寄存器
0x34	CAN_RXERR	接收错误计数寄存器
0x38	CAN_TXERR	发送错误计数寄存器
0x3C	CAN_ALC	仲裁丢失捕获寄存器
0x40	CAN_RXADDR	接收缓存基地址设置寄存器

15.3.1 模式寄存器 CAN_MR (偏移: 00H)

比特	名称	属性	复位值	描述
31:8	RSV	R	0	保留
7	RXF_CLR	W	0	RX_FIFO 指针清除位: 0: 无变化 1: 复位 RX_FIFO 的读写指针, 复位后此位自动恢复为 0
6:3	RSV	R	0	保留
2	RM	R/W	1	复位模式设置位: 0: CAN 工作在其他模式 1: CAN 工作在复位模式 在复位模式中不进行数据的发送和接收, 此模式用于进行一些硬件的配置 (某些寄存器只能在复位模式下进行写操作), 在复位模式以后, 可进入监听模式或者正常模式
1	LOM	R/W	0	监听模式设置位: 0: 若 RM=0, CAN 进入正常模式 1: 若 RM=0, CAN 进入监听模式* 此位只能在复位模式设置
0	AFM	R/W	0	硬件匹配数据选择位: 0: 使用双过滤器 1: 使用单过滤器 此位只能在复位模式设置

注：在监听模式下，即使成功接收到消息，CAN 控制器也不会对 CAN 总线进行应答（不会发送 ACK 响应）。错误计数器将停止在当前值。监听模式主要用于比特率检测，不会干扰网络流量，监听模式还可用于 CAN 总线分析仪。

15.3.2 指令寄存器 CAN_CMRR (偏移: 04H)

比特	名称	属性	复位值	描述
31:3	RSV	R	0	保留
2	TR	W	0	发送请求设置位： 0：禁止发送 1：启动发送，进行帧传输
1	AT	W	0	中止传输允许位： 0：禁止中止传输 1：允许中止传输 同时设置 TR 和 AT 可启动单发传输，在总线错误或者仲裁丢失的情况下，不会执行帧的重新传输。中止只对即将要传输的帧作用，已经发出的帧无法中止。如果在上一个命令中将 TR 设为 1 来启动传输，则无法通过将 TR 位设为 0 来取消，此时可通过设置 AT 为 1 来取消传输
0	RSV	R	0	保留

15.3.3 状态寄存器 CAN_SR (偏移: 08H)

比特	名称	属性	复位值	描述
31:8	RSV	R	0	保留
7	RBS	R	0	接收 FIFO 状态： 0：FIFO 中没有消息 1：FIFO 中至少有一条消息
6	DSO	R	0	数据溢出状态： 0：自上次清除数据溢出以来未发生溢出 1：RX FIFO 溢出，RX 溢出中断触发（如使能）

比特	名称	属性	复位值	描述
5	TBS	R	1	发送 BUFFER 状态： 0：发送 BUFFER 已锁定。正在发送消息或正在等待发送。如果 CPU 在锁定状态下（TBS = 0）尝试写入发送缓冲区，则不接受已写入的数据 1：发送 BUFFER 可被 CPU 写入
4	RSV	R	0	保留
3	RS	R	0	接收状态位： 0：CAN 未处于接收状态 1：CAN 正在接收
2	TS	R	0	发送状态位： 0：CAN 未处于传输状态 1：CAN 正在传输
1	ES	R	0	错误状态位： 0：正常状态 1：至少一个 CAN 错误计数器达到错误警告限制（96）
0	BS	R	0	总线状态位 0：正常状态。可进行帧传输和接收 1：离线状态。CAN 控制器处于复位模式，错误警告中断触发（如使能）。发送错误计数器设为 127，接收错误计数器设为 0。CAN 将一直处于复位模式，直到 CPU 将 RM 位清掉。完成此操作后，CAN 将等待 128 次总线空闲信号的出现（11 个连续的隐性位），发送错误计数器向下计数。然后 BS 位清 0，错误计数器复位，错误警告中断触发（如使能）

15.3.4 中断状态/应答寄存器 CAN_ISR（偏移：0CH）

比特	名称	属性	复位值	描述
31:7	RSV	R	0	保留
6	ALI	R/W	0	仲裁丢失中断状态位： 当 CAN 在消息传输过程中丢失仲裁并成为接收端时，此位置位，可以读取 ALC 寄存器以检查丢失了仲裁段中的哪一位，写 1 清除此中断

比特	名称	属性	复位值	描述
5	EWI	R/W	0	错误警告中断状态位： 当 SR 寄存器的 ES 或 BS 位改变时，错误警告中断置位。因此，它可用于检测 CAN 是否进入或退出总线关闭状态。写 1 清除中断
4	EPI	R/W	0	错误被动中断状态位： 当 CAN 总线控制器达到或退出错误被动级别（即在状态更改为主动到被动或被动到主动）时，此位置位。写 1 清除中断
3	RI	R/W	0	接收中断状态位： 当接收 FIFO 中至少有一条 CAN 帧数据时，CAN 将此位置 1。读取消息后，CPU 必须将 RI 位写 1（消息读取确认），以减少 RX 消息计数器(RMC)计数，RMC 不会自动递减
2	TI	R/W	0	发送中断状态位： 成功发送后，发送中断位被置位。在写入新的数据帧之前可通过清除 TI 位（写 1 清除）将写指针复位到 TX RAM
1	BEI	R/W	0	总线错误中断状态位： 当 CAN 在发送或接收消息时遇到总线错误时，将 BEI 置位。写 1 清除中断
0	DOI	R/W	0	接收数据溢出中断状态位： 发生接收FIFO溢出时，DOI置位。写1清除中断

15.3.5 中断使能寄存器 CAN_IMR（偏移：10H）

比特	名称	属性	复位值	描述
7	RSV	R	0	保留
6	ALIM	R/W	0	仲裁丢失中断使能位。使能 CAN 发送器在发送期间丢失仲裁并成为 CAN 接收器时触发中断： 0：禁止 ALI 中断 1：使能 ALI 中断
5	EWIM	R/W	0	错误警告中断使能位。使能当 CAN_SR 寄存器的 BS 或 ES 位状态改变时触发中断： 0：禁止 EWI 中断 1：使能 EWI 中断

比特	名称	属性	复位值	描述
4	EPIM	R/W	0	错误被动中断使能位。使能当 CAN 控制器进入或离开被动错误模式时触发中断： 0：禁止 EPI 中断 1：使能 EPI 中断
3	RIM	R/W	0	接收中断使能位： 0：禁止 RI 中断 1：使能 RI 中断
2	TIM	R/W	0	发送中断使能位： 0：禁止 TI 中断 1：使能 TI 中断
1	BEIM	R/W	0	总线错误中断使能位。使能当 CAN 在发送或接收过程中发生总线错误时触发中断： 0：禁止 BEI 中断 1：使能 BEI 中断
0	DOIM	R/W	0	接收数据溢出中断使能位： 0：禁止 DOI 中断 1：使能 DOI 中断

15.3.6 接收数据计数寄存器 CAN_RMC（偏移：14H）

比特	名称	属性	复位值	描述
7:5	RSV	R	0	保留
4:0	RMC	R	0	接收 FIFO 中 CAN 帧个数 接收 FIFO 最多可以存储 16 条消息。以下等式允许计算要存储的最大消息数-RX FIFO： $n = \frac{64}{3 + data_length_code}$ 注：此处 data_length_code 至少为 1，若 CAN 数据段长度为 0，data_length_code=1

15.3.7 总线时序寄存器 CAN_BTR0（偏移：18H）

此寄存器只能在复位模式写入，可在任何模式读取。

比特	名称	属性	复位值	描述
31:8	RSV	R	0	保留

比特	名称	属性	复位值	描述
7:6	SJW	R/W	0	同步跳跃宽度： $t_{SJW}=t_{SCLK} \times (2 \times SJW.1 + SJW.0 + 1)$ 为了补偿不同 CAN 总线控制器的时钟振荡器之间的相移，必须相应地缩短或延长位周期。SJW 定义了一个重新同步可以改变一个位周期的最大时钟周期数。再同步过程中，硬件会通过 在 PBS1 段内增加 $1+SJW$ 个 t_{SCLK} ，或者在 PBS2 段内减少 $1\sim (1+SJW)$ 个 t_{SCLK} 来与接收信号达到同步
5:0	BRP	R/W	0	波特率预分频值： $t_{SCLK}=2 \times t_{CLK} \times (32 \times BRP.5 + 16 \times BRP.4 + 8 \times BRP.3 + 4 \times BRP.2 + 2 \times BRP.1 + BRP.0 + 1)$ 其中， $t_{CLK} = 1/f_{PCLK}$

15.3.8 总线时序寄存器 CAN_BTR1（偏移：1CH）

此寄存器只能在复位模式写入，可在任何模式读取。

比特	名称	属性	复位值	描述
31:8	RSV	R	0	保留
7	SAM	R/W	0	总线电平采样数选择位： 0：采样一次总线电平（适用于高速总线） 1：采样三次总线电平（适用于中/低速总线）
6: 4	TSEG2	R/W	0	Time Segment 2 的时钟周期数 $t_{TSEG2}= t_{SCLK} \times (4 \times TSEG2.2 + 2 \times TSEG2.1 + TSEG2.0 + 1)$
3: 0	TSEG1	R/W	0	Time Segment 1 的时钟周期数 $t_{TSEG1}= t_{SCLK} \times (8 \times TSEG1.3 + 4 \times TSEG1.2 + 2 \times TSEG1.1 + TSEG1.0 + 1)$

CAN 的位周期结构如图 15-1 所示。其中同步段（SYNC SEG）为 $1 \times t_{SCLK}$ ，相位缓冲段 1 和 2 长度由 TSEG1 和 TSEG2 决定。

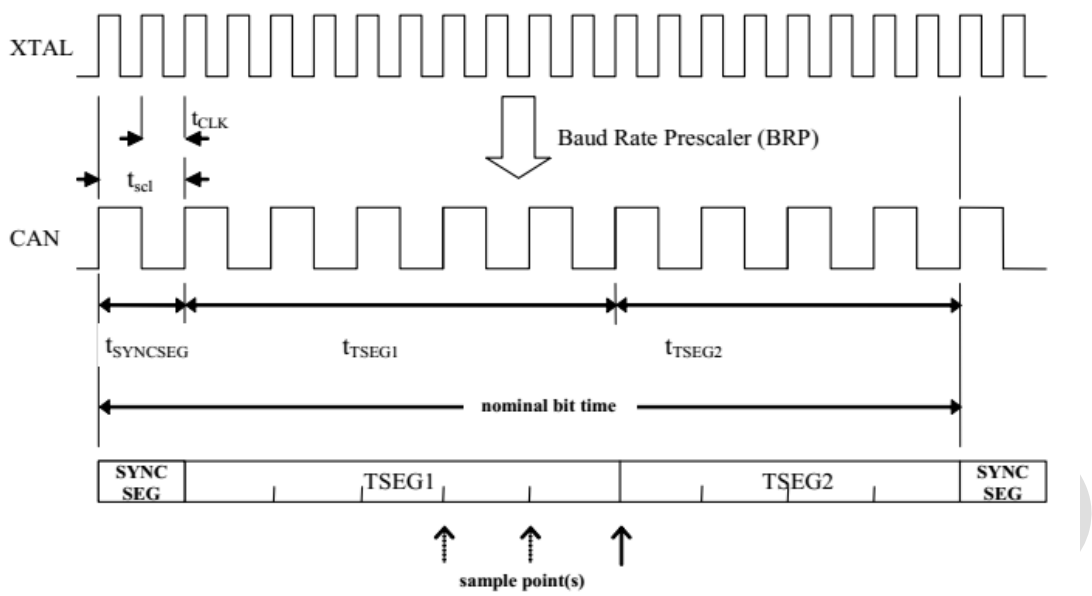


图 15-1: CAN 位周期结构

15.3.9 发送缓存寄存器 CAN_TXBUF (偏移: 20H)

比特	名称	属性	复位值	描述
31:0	TXBUF	W	0	发送缓存寄存器用于写入要通过 CAN 网络发送的 CAN 帧。 写入该寄存器执行内部写指针的自动递增，通过在 ISR 寄存器中写入 TI 位，可以将写指针复位到发送内存的地址 0h 处

15.3.10 接收缓存寄存器 CAN_RXBUF (偏移: 24H)

比特	名称	属性	复位值	描述
31:0	RXBUF	R	0	接收缓存寄存器用于读取从 CAN 网络接收的 CAN 帧。 读取该寄存器将自动递增内部 FIFO 的读取地址指针（读取后递增）

在收到一帧 CAN 数据后，RXBUF 寄存器读到的数据格式如图 15-2 所示（databuf 段长度由 DLC 段决定，数量为 0-8Bytes）:

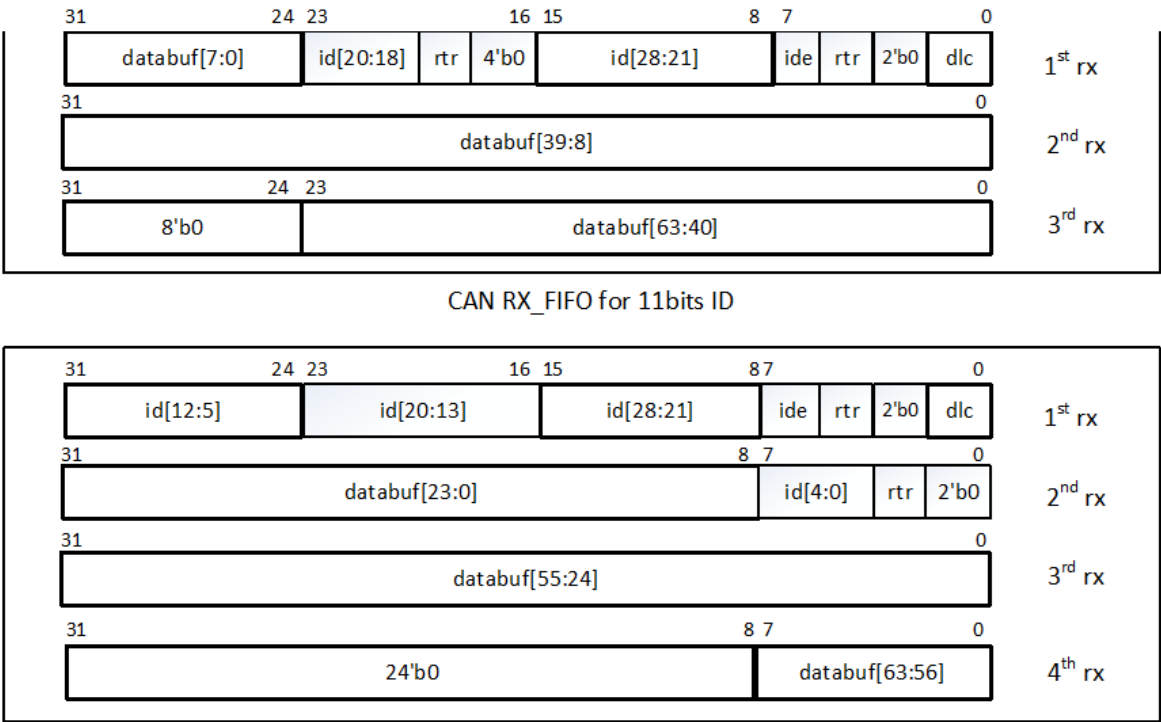


图 15-2: CAN RX FIFO 29 位 ID 数据格式

在接收完一帧 CAN 数据后，RMC 寄存器计数加 1，此时 CAN 控制器会往 RX FIFO 中逐个写入数据，当写入一个 32 位数据后，RBS 置位。在写入完一帧数据后，RI 标志位置位。

15.3.11 接收过滤匹配寄存器 CAN_ACR（偏移：28H）

只有当接收到的消息的标识符位等于接收过滤器中的预定义位时，CAN 控制器中的接收过滤器才有可能将接收到的消息传递给 RX FIFO。接收过滤器由接收过滤匹配寄存器（ACR3：ACR0）和接收过滤屏蔽寄存器（AMR3：AMR0）组成。模式寄存器的 AFM 位可设置单/双过滤器。在单过滤器配置中，过滤器为 4 字节长。若接收的数据为标准帧模式，可接收到包括仲裁位，RTR 位和数据位的前 2 个字节（数据字节不是必须接收的部分）。所有单个位的比较都必须发出信号，表示成功接收到数据；若接收的数据为拓展帧格式，可接收到仲裁位和 RTR 位数据。对于格式中没定义的位，过滤器将不进行对比。

双过滤器配置会定义两个长度更短的过滤器。接收到的数据将会跟两个过滤器对比，决定是否应该将数据存入 RX FIFO 中。如果至少一个接收过滤器对比成功，接收到的数据将被存储在 FIFO 中。如果接收到标准格式的帧，第一个过滤器将会对比标准格式的仲裁，RTR 位和第一个数据字节。

第二个过滤器只对比标准格式的仲裁和 RTR 位。如果过滤器 1 中不对数据字节过滤，需要把 AMR1 和 AMR3 的低四位设成逻辑 1（不对比此位）。

比特	名称	属性	复位值	描述
31: 0	ACR3-0	R/W	0	接收过滤匹配寄存器包含要接收的消息的仲裁位, 而相应的接收过滤屏蔽寄存器定义了将比较哪些位位置 and 无关位

15.3.12 接收过滤屏蔽寄存器 CAN_AMR (偏移: 2CH)

只有当接收到的消息的标识符位等于接收过滤器中的预定义位时，CAN 控制器中的接收过滤器才有可能将接收到的消息传递给 RX FIFO。接收过滤器由接收过滤匹配寄存器（ACR3: ACR0）和接收过滤屏蔽寄存器（AMR3: AMR0）定义。

比特	名称	属性	复位值	描述
31:0	AMR3-0	R/W	0	接收过滤屏蔽寄存器定义了将比较哪些位位置 and 无关位。将相应的位设为 1 表示不对比 ACR 寄存器中相应的位

不同过滤器设置以及不同仲裁长度（标准帧11位/拓展帧29位）对应的位格式如图 15-3所示：

- 当设置为单过滤器时：

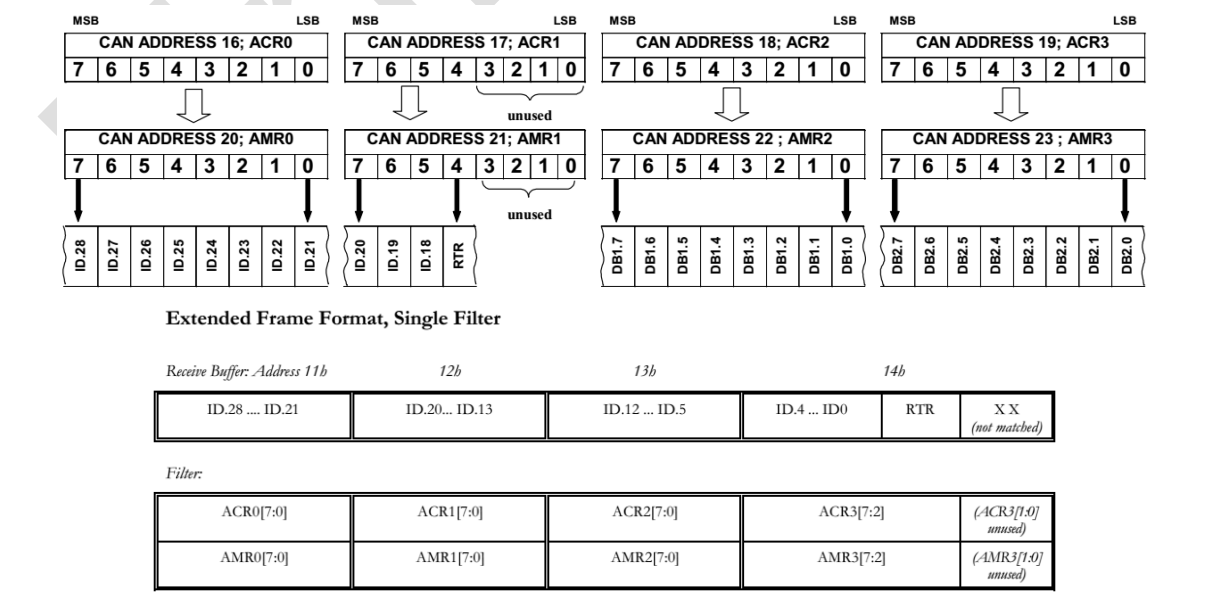
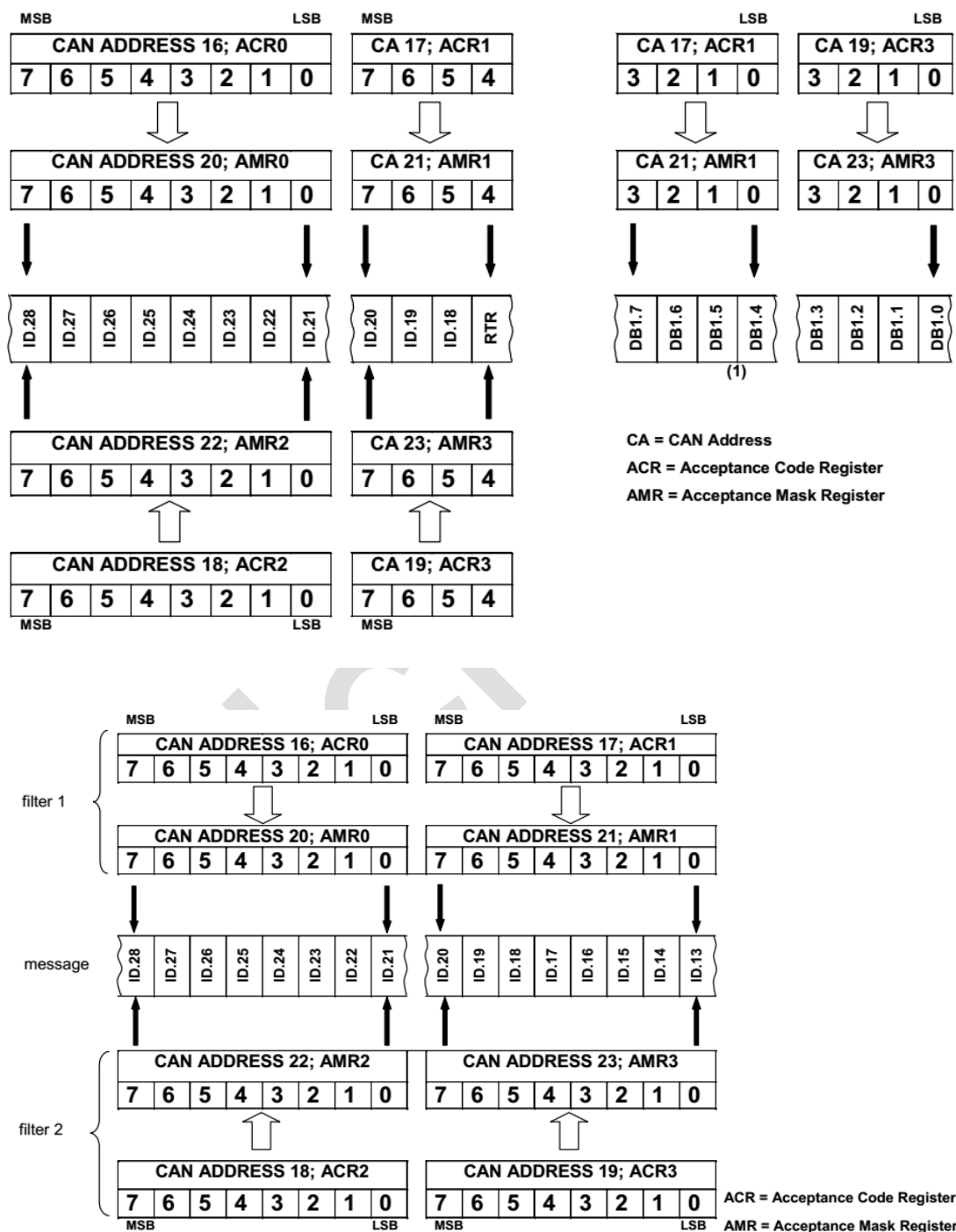


图 15-3：单过滤器

- 当设置为双过滤器时:

标准模式下, 当接收到数据后, 将会与第一个过滤器的 ID 包括 RTR 位, 以及第一个收到的数据字节进行对比, 或者与第二个过滤器的 ID 位包括 RTR 位进行对比。



15.3.13 错误码捕捉寄存器 CAN_ECC (偏移: 30H)

ECC 只读寄存器保存有关 CAN 网络上发生的最后总线错误的错误代码。该寄存器是只读的。

在确认先前的总线错误之前（通过确认总线错误中断），CAN 内核不会更新该寄存器。

比特	名称	属性	复位值	描述
31:8	RSV	R	-	保留
7	RXWRN	R	0	当 RXERR 计数器大于或等于 96 时置 1
6	TXWRN	R	0	当 TXERR 计数器大于或等于 96 时置 1
5	EDIR	R	0	表示错误发生时数据传输方向： 0: 发送 1: 接收
4	ACKER	R	0	发生 ACK 错误时置位
3	FRMER	R	0	发生帧格式错误时置位
2	CRCER	R	0	发生 CRC 错误时置位
1	STFER	R	0	发生填充错误时置位
0	BER	R	0	发生位错误时置位

15.3.14 接收错误计数寄存器 CAN_RXERR (偏移: 34H)

比特	名称	属性	复位值	描述
31:8	RSV	R	0	保留
7:0	RXERR	R	0	接收错误计数器的当前值。如果发生总线关闭事件，则 RX 错误计数器将初始化为 0

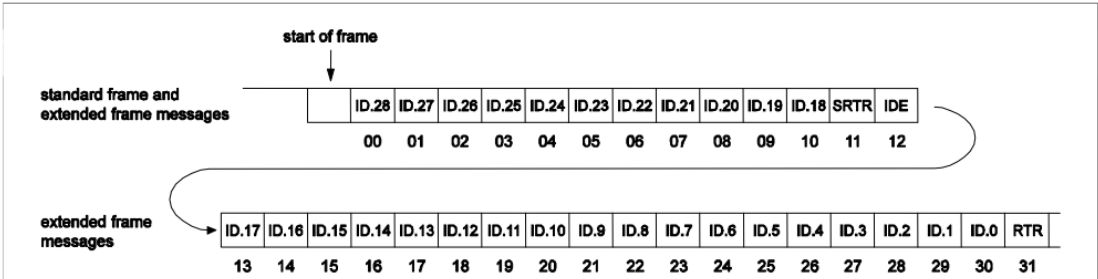
15.3.15 发送错误计数寄存器 CAN_TXERR (偏移: 38H)

比特	名称	属性	复位值	描述
31:8	RSV	R	0	保留
7:0	TXERR	R	0	发送错误计数器的当前值的低 8 位。如果发生总线关闭事件，则将传输错误计数器初始化为 127，以计算最小协议定义的时间（出现 128 次总线空闲信号）。在这段时间内读取 TXERR 可获得有关总线关闭恢复状态的信息

15.3.16 仲裁丢失捕获寄存器 CAN_ALC (偏移: 3CH)

CAN 控制器能够确定仲裁丢失的确切帧内位置。紧随其后，将产生“仲裁丢失中断”。此外，在仲裁丢失捕获寄存器中捕获位数。一旦主机控制器读取了该寄存器的内容，就会为下一个仲裁丢失情况激活捕获功能。此功能允许 CAN 监视每个 CAN 总线访问。对于诊断或在系统配置期间，可以确定仲裁不成功的每种情况。

比特	名称	属性	复位值	描述
31:5	RSV	R	0	保留
4:0	ALC	R	0	仲裁丢失位置



Bits					Decimal Value	Description
ALC4	ALC3	ALC2	ALC1	ALC0		
0	0	0	0	0	00	Arbitration lost in ID28 / 10
0	0	0	0	1	01	Arbitration lost in ID27 / 9
0	0	0	1	0	02	Arbitration lost in ID26 / 8
0	0	0	1	1	03	Arbitration lost in ID25 / 7
0	0	1	0	0	04	Arbitration lost in ID24 / 6
0	0	1	0	1	05	Arbitration lost in ID23 / 5
0	0	1	1	0	06	Arbitration lost in ID22 / 4
0	0	1	1	1	07	Arbitration lost in ID21 / 3
0	1	0	0	0	08	Arbitration lost in ID20 / 2
0	1	0	0	1	09	Arbitration lost in ID19 / 1
0	1	0	1	0	10	Arbitration lost in ID18 / 0
0	1	0	1	1	11	Arbitration lost in SRTR / RTR
0	1	1	0	0	12	Arbitration lost in IDE bit
0	1	1	0	1	13	Arbitration lost in ID17*
0	1	1	1	0	14	Arbitration lost in ID16*
0	1	1	1	1	15	Arbitration lost in ID15*
1	0	0	0	0	16	Arbitration lost in ID14*
1	0	0	0	1	17	Arbitration lost in ID13*
1	0	0	1	0	18	Arbitration lost in ID12*
1	0	0	1	1	19	Arbitration lost in ID11*
1	0	1	0	0	20	Arbitration lost in ID10*
1	0	1	0	1	21	Arbitration lost in ID9*
1	0	1	1	0	22	Arbitration lost in ID8*
1	0	1	1	1	23	Arbitration lost in ID7*
1	1	0	0	0	24	Arbitration lost in ID6*
1	1	0	0	1	25	Arbitration lost in ID5*
1	1	0	1	0	26	Arbitration lost in ID4*
1	1	0	1	1	27	Arbitration lost in ID3*
1	1	1	0	0	28	Arbitration lost in ID2*
1	1	1	0	1	29	Arbitration lost in ID1*
1	1	1	1	0	30	Arbitration lost in ID0*
1	1	1	1	1	31	Arbitration lost in RTR

15.3.17 接收缓存基地址设置寄存器 CAN_RXADDR (偏移: 40H)

比特	名称	属性	复位值	描述
31:11	RSV	R	0	保留
10:2	RX_ADDR_BASE	R/W	0	设置 rx fifo 使用 sram 中的地址的偏移地址 (rx fifo 是基地址为 0x20003000 (+偏移地址) 后的 64bytes 空间, 读取 fifo 数据时可从 0x20003000 (+偏移地址) 开始读取, 软件在编 译时需注意 SRAM 空间在此地址处的内存划分, 避免与其他数据存放有冲突。偏移地址为 {RX_ADDR_BASE[8:0], 2'b0})
1:0	RSV	R	0	保留

15.4 使用流程

15.4.1 发送 CAN 数据帧

1. 开启 CAN 时钟, 释放复位, can 功能引脚复用。
2. 配置总线时序寄存器 CAN_BTR0/CAN_BTR1。
3. CAN_ISR 寄存器清除错误标志位/中断标志位。
4. 配置中断使能 IMR 寄存器, 使能 TI 中断 (可选)。
5. 配置模式 MR 寄存器的 LOM 位, 进入正常模式。
6. 配置发送缓存寄存器 CAN_TXBUF, 根据定义的格式写入 CAN 数据帧内容, 按发送的先后顺序写入, 每次写入 32 位数据。
7. 配置指令 CMR 寄存器的 TR 位, 启动发送。
8. 等待状态寄存器的TBS位置 1 后 (若使能TI中断, 此处可选择等待TI中断触发), 数据发送完毕。

15.4.2 接收 CAN 数据帧

1. 开启 CAN 时钟, 释放复位, can 功能引脚复用。
2. 配置总线时序寄存器 CAN_BTR0/CAN_BTR1。

3. CAN_ISR 寄存器清除错误标志位/中断标志位。
4. 配置中断使能 IMR 寄存器，使能 RI 中断（可选）。
5. 设置接收过滤器配置，若使用单过滤器，CAN_MR 寄存器 AFM 位置 1。CAN_ACR 寄存器配置用户需要过滤筛选的内容，CAN_AMR 寄存器选择需要与 ACR 寄存器进行对比的位。若不需要进行对，AMR 寄存器所有位置 1。
6. 配置模式 MR 寄存器的 LOM 位，进入正常模式。
7. 等待状态寄存器的 RBS 位置 1 后（若使能 RI 中断，此处可选择等待 RI 中断触发），读取接收缓存寄存器 CAN_RXBUF 数据，多次读取直到取出所有数据。

15.4.3 CAN 速率计算

CAN 的波特率可以用以下四个变量可以算出：

- 最小时间段：Tscl
- 时间段 1：tesg1
- 时间段 2：tesg2
- 同步跳转宽度：SJW

其中最小时间段由 CAN 控制器的时钟频率以及分频决定。

tesg1=TS1+1, tesg2=TS2+1; prescaler=2(BRP+1)

$$BitRate = \frac{ClockFrequency}{prescaler \times (tesg1 + tesg2 + 1)}$$

例如：

速率可以选择为 1M/500k/250k/125k bps

APB 时钟=PCLK=32MHz,

CAN 波特率 BitRate = Fpclk/(2*((BRP+1)*(TS1+TS2+3))), 默认约定：TS1>=TS2

- 设波特率为 1M 的参数

设 BRP=1(4 分频), BitRate = 1M = 32M/(2*((1+1)*(TS1+TS2+3))), 所以可以设置 TS1=3,

TS2=2

- 设波特率为 500K 的参数

设 $BRP=3$ (8 分频), $BitRate = 0.5M = 32M / (2((3+1)*(TS1+TS2+3)))$, 所以可以设置 $TS1=3$, $TS2=2$

- 设波特率为 250K 的参数

设 $BRP=7$ (16 分频), $BitRate = 0.25M = 32M / (2((7+1)*(TS1+TS2+3)))$, 所以可以设置 $TS1=3$, $TS2=2$

- 设波特率为 125K 的参数

设 $BRP=15$ (32 分频), $BitRate = 0.125M = 32M / (2((15+1)*(TS1+TS2+3)))$, 所以可以设置 $TS1=3$, $TS2=2$

16 GTIMER

16.1 概述

有 3 个 16 位的通用定时/计数器 Timer(Gtimer0、Gtimer1、Gtimer2)，每个定时器都有自己独立的中断。这些 Timer 可以有多种用途，包括测量输入信号的脉冲宽度（输入捕获），产生输出波形（PWM、带死区时间的互补 PWM），计数器可以向上，向下，向上/下三种计数方向，且计数值可以随时由软件读取。每个 Timer 有 2 路 PWM 输出(可选是否互补)，有 1 路输入捕获。

16.2 主要特性

- 16 位向上、向下、向上/下计数自动重载计数器
- 16 位可编程预分频器，支持实时调整计数时钟分频
- 灵活的计数时钟源选择
- 通道可用于输入捕获、输出比较、PWM(边沿或中央对齐模式)、单脉冲输出
- 支持定时器间的级联
- 可编程的带死区时间互补输出
- 带刹车输入信号控制功能，可使 PWM 输出置于一个可设置的状态
- 中断在以下几种情况产生：
 - Update 中断：计数器向上/向下溢出
 - 输入捕获
 - 输出比较
 - 刹车信号输入
- 支持定时器同步使能

16.3 寄存器描述

GTimer0 寄存器基地址: 0x40000C00

GTimer1 寄存器基地址: 0x40003400

GTimer2 寄存器基地址: 0x40003800

表 16-1: GTIMER 寄存器列表

偏置	名称	描述
0x00	GTIM_CR	GTIM 控制寄存器
0x04	GTIM_IER	GTIM 中断使能寄存器
0x08	GTIM_SR	GTIM 状态寄存器
0x0C	GTIM_EGR	GTIM 事件产生寄存器
0x10	GTIM_CCMR	GTIM 捕捉/比较模式寄存器
0x14	GTIM_CCER	GTIM 捕获/比较使能寄存器
0x18	GTIM_CNT	GTIM 计数器寄存器
0x1C	GTIM_PSC	GTIM 预分频寄存器
0x20	GTIM_ARR	GTIM 自动重载寄存器
0x24	GTIM_CCR	GTIM 捕捉/比较寄存器
0x28	GTIM_CARS1	GTIM 硬件触发寄存器

16.3.1 GTIM 控制寄存器 GTIM_CR (偏移: 00H)

比特	名称	属性	复位值	描述
31:29	RSV	-	-	保留
28	ADC_HDT	R/W	0	ADC 硬件触发功能使能: 0: ADC硬件触发功能禁止 1: ADC 硬件触发功能使能
27	OPA_NKEN	R/W	0	OPA 作为刹车源使能 0: 关闭 OPA 作为刹车源 1: 使能 OPA 作为刹车源
26	COMP2_BKEN	R/W	0	COMP2 作为刹车源使能: 0: 关闭 COMP2 作为刹车源 1: 使能 COMP2 作为刹车源
25:24	RSV	-	-	保留

比特	名称	属性	复位值	描述
23	LVD_BKEN	R/W	0	LVD 作为刹车源使能： 0：关闭 LVD 作为刹车源 1：使能 LVD 作为刹车源
22	BREAK2_SEL	R/W	0	Gtimer0 IO 引脚刹车源选择： 0：不选择 Gtimer2 配置的 IO 作为刹车源 1：选择 Gtimer2 配置的 IO 作为刹车源 Gtimer1 IO 引脚刹车源选择： 0：不选择 Gtimer2 配置的 IO 作为刹车源 1：选择 Gtimer2 配置的 IO 作为刹车源 Gtimer2 IO 引脚刹车源选择： 0：不选择 Gtimer1 配置的 IO 作为刹车源 1：选择 Gtimer1 配置的 IO 作为刹车源
21	BREAK1_SEL	R/W	0	Gtimer0 IO 引脚刹车源选择： 0：不选择 Gtimer1 配置的 IO 作为刹车源 1：选择 Gtimer1 配置的 IO 作为刹车源 Gtimer1 IO 引脚刹车源选择： 0：不选择 Gtimer0 配置的 IO 作为刹车源 1：选择 Gtimer0 配置的 IO 作为刹车源

比特	名称	属性	复位值	描述
				Gtimer2 IO 引脚刹车源选择: 0: 不选择 Gtimer0 配置的 IO 作为刹车源 1: 选择 Gtimer0 配置的 IO 作为刹车源
20	PWMN_IDLE	R/W	0	PWM 输出负向电平 IDLE 状态: 0: 电平为低电平 1: 电平为高电平
19	PWMP_IDLE	R/W	0	PWM 输出正向电平 IDLE 状态: 0: 电平为低电平 1: 电平为高电平
18	MOE	R/W	0	输出使能: 0: 输出禁止 1: 输出总使能
17:16	PWMN_B_S	R/W	0	PWM 刹车触发后, PWM 互补电平状态设置位: 00: 低电平 01: 高电平 10/11: 高阻状态
15:14	PWMS_B_S	R/W	0	PWM 刹车触发后, PWM 正向电平状态设置位: 00: 低电平 01: 高电平 10/11: 高阻状态
13	SOFT_BK	R/W	0	软件触发刹车功能设置位: 0: 软件不触发刹车功能 1: 软件触发刹车功能
12	CEN_ALL	W	0	Gtimer0 0: 无操作, 读此位始终为 0 1: 同时使能 Gtimer0/1/2 和 LPTIM0/1/2, 写此位后, Gtimer0/1/2 和 LPTIM0/1/2 的 CEN 位同时为 1 Gtimer1: 无效位 Gtimer2: 无效位

比特	名称	属性	复位值	描述
11	BKE_POL	R/W	0	刹车信号极性配置： 0：刹车信号高电平有效 1：刹车信号低电平有效
10	BKE	R/W	0	刹车功能使能： 0：刹车功能禁止 1：刹车功能使能
9	PWM_DEAD	R/W	0	PWM 死区插入功能使能： 0：避免死区功能关闭 1：避免死区功能使能
8	PWM_INV	R/W	0	互补 PWM 与原 PWM 差分使能： 0：互补 PWM 和原 PWM 同相位 1：互补 PWM 和原 PWM 反相位
7	MMS	R/W	0	主机模式选择，用于配置主机模式下向从机发送的同步触发信号（TRGO）源： 0：OC1REF 用作 TRGO 1：UE（update event）信号被用作 TRGO
6	ARPE	R/W	0	Auto-reload 预装载使能： 0：ARR 寄存器不使能 preload 1：ARR 寄存器使能 preload
5:4	CMS	R/W	0	计数器对齐模式选择： 00：边沿对齐模式 01：中央对齐模式 1，输出比较中断标志仅在计数器向下计数的过程中置位 10：中央对齐模式 2，输出比较中断标志仅在计数器向上计数的过程中置位 11：中央对齐模式 3，输出比较中断标志在计数器向上向下计数的过程中都会置位
3	CEN_ALL_EN	R/W	0	CEN_ALL 使能： 0：当前 Gtimer 对 CEN_ALL 信号无效 1：当前 Gtimer 可以被 CEN_ALL 控制

比特	名称	属性	复位值	描述
2	DIR	R/W	0	计数方向寄存器: 0: 向上计数 1: 向下计数 注: 当定时器配置为中央计数模式时, 此寄存器只读
1	OPM	R/W	0	单脉冲输出模式: 0: Update Event发生时计数器不停止 1: Update Event 发生时计数器停止 (自动清零 CEN)
0	CEN	R/W	0	计数器使能: 0: 计数器关闭 1: 计数器使能

16.3.2 GTIM 中断使能寄存器 GTIM_IER (偏移: 04H)

比特	名称	属性	复位值	描述
31:3	RSV	-	-	保留
2	BKE_IE	R/W	0	刹车中断使能: 0: 刹车中断禁止 1: 刹车中断使能
1	CCIE	R/W	0	捕捉/比较通道中断使能: 0: 禁止捕捉/比较中断 1: 允许捕捉/比较中断
0	UIE	R/W	0	Update 事件中断使能: 0: 禁止 Update 事件中断 1: 允许 Update 事件中断

16.3.3 GTIM 状态寄存器 GTIM_SR (偏移: 08H)

比特	名称	属性	复位值	描述
31:3	RSV	-	-	保留
2	BKE_F	R/W1C	0	刹车中断标志: 0: 未处于刹车状态 1: 处于刹车状态 写 1 清 0

比特	名称	属性	复位值	描述
1	CCIF	R/W1C	0	捕捉/比较通道中断标志： 如果CC通道配置为输出：CCIF在计数值等于比较值时置位，软件写1清零 如果CC通道配置为输入：发生捕捉事件时置位，软件写1清零，或者软件读GTIM_CCR自动清零
0	UIF	R/W1C	0	Update 事件中断标志，硬件置位，软件写 1 清零 当发生更新事件时，UIF置位，并更新shadow寄存器

16.3.4 GTIM 事件产生寄存器 GTIM_EGR (偏移: 0CH)

比特	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	UG	W	0	软件 Update 事件, 软件置位此寄存器产生 Update 事件，硬件自动清零。 软件置位UG时会重新初始化计数器并更新shadow寄存器，预分频计数器被清零

16.3.5 GTIM 捕捉/比较模式寄存器 GTIM_CCMR (偏移: 10H)

比特	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15	CAPCLR	R/W	0	首次捕捉清零控制位： 0：计数器使能后立即计数 1：计数器使能后保持为 0，直到捕捉到第一个沿开始计数
14:13	ICPSC	R/W	0	捕捉源预分频位： 00：除1 01：除2 10：除4 11：除8

比特	名称	属性	复位值	描述
12	CAPFLT	R/W	0	输入捕捉信号滤波使能： 0：无输入滤波功能 1：有输入滤波功能
11:10	CAPEDGE	R/W	0	捕获沿触发控制位： 00：上升沿触发 01：下降沿触发 10/11：上升或下降沿触发
9:8	CAPSSEL	R/W	0	捕捉源选择位： Gtimer0 00：GTIMER0_CH 01：UART0_RX 10：LPTIM0_OUT 11：CLK32K_GTIMER0 Gtimer1 00：GTIMER1_CH 01：UART1_RX 10：LPTIM1_OUT 11：CLK32K_GTIMER1 Gtimer2 00：GTIMER2_CH 01：I2C_SCL 10：LPTIM2_OUT 11：CLK32K_GTIMER2
7	TFLT	R/W	0	外部计数源滤波使能： 0：无滤波功能 1：有滤波功能
6	TEDGE	R/W	0	计数源边沿选择： 0：上升沿计数 1：下降沿计数

比特	名称	属性	复位值	描述
5:4	TSSEL	R/W	0	计数源选择位： Gtimer0 00: APBCLK (PCLK) 01: NC 10: CLK32K_GTIMER0 (32k时钟) 11: GTIMER0_CH (GTIMER0捕获输入) Gtimer1 00: APBCLK (PCLK) 01: NC 10: CLK32K_GTIMER1 (32k时钟) 11: GTIMER1_CH (GTIMER1捕获输入) Gtimer2 00: APBCLK (PCLK) 01: NC 10: CLK32K_GTIMER1 (32k时钟) 11: GTIMER2_CH (GTIMER2捕获输入)
3:1	RSV	-	-	保留
0	CCS	R/W	0	捕捉/比较 1 通道选择： 0: CC 通道配置为输出 1: CC 通道配置为输入 注：CCS仅在通道关闭时（CCE=0）可以写

16.3.6 GTIM 捕捉/比较使能寄存器 GTIM_CCER (偏移：14H)

比特	名称	属性	复位值	描述
31:2	RSV	-	-	保留

比特	名称	属性	复位值	描述
1	CCP	R/W	0	CC 通道配置为输出时极性： 0: CNT<CCR 时输出高电平 1: CNT>CCR时输出高电平
0	CCE	R/W	0	捕捉/比较输出使能： CC 通道配置为输出时： 0: OC 无输出 1: OC 有输出 CC 通道配置为输入时： 0: 关闭捕捉功能 1: 使能捕捉功能

16.3.7 GTIM 计数寄存器 GTIM_CNT（偏移：18H）

比特	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	CNT	R	0	计数器值

16.3.8 GTIM 预分频寄存器 GTIM_PSC（偏移：1CH）

比特	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	PSC	R/W	0	计数器时钟（CK_CNT）预分频值： $f_{CK_CNT}=f_{CK_PSC}/(PSC[15:0]+1)$ 这是一个preload寄存器, 在update事件发生时其内容被载入shadow寄存器

注：不使能 preload，依旧要等到有 update 事件才能使 psc 值载入 shadow 寄存器

16.3.9 GTIM 自动重载寄存器 GTIM_ARR (偏移: 20H)

比特	名称	属性	复位值	描述
31:16	ARRN	R/W	0	计数溢出时的自动重载值，互补计数器 这是一个preload寄存器, 在update事件发生时其内容被载入shadow寄存器
15:0	ARR	R/W	0	计数溢出时的自动重载值 这是一个preload寄存器, 在update事件发生时其内容被载入shadow寄存器

16.3.10 GTIM 捕捉/比较寄存器 GTIM_CCR (偏移: 24H)

比特	名称	属性	复位值	描述
31:16	CCRN	R/W	0	捕捉/比较通道寄存器，互补计数器 如果通道配置为输出： 这是一个 preload 寄存器, 其内容被载入 shadow 寄存器后用于与计数器比较产生 OC 输出
15:0	CCR	R/W	0	捕捉/比较通道寄存器 如果通道配置为输出： 这是一个 preload 寄存器, 其内容被载入 shadow 寄存器后用于与计数器比较产生 OC 输出。 注：中央对齐模式中，当此值设置和 ARR 相等时，向上计数过程中无法产生 CCIF 中断，只有向下计数过程会产生中断。 如果通道配置为输入： CCR 保存最近一次输入捕捉事件发生时的计数器值，此时 CCR 为只读

16.3.11 GTIM 硬件触发寄存器 GTIM_CARS1（偏移：28H）

比特	名称	属性	复位值	描述
31:16	ARRS1	R/W	0	ADC 硬件触发时间设置： 此位用来设置 ADC 硬件触发点。 通过此组寄存器，设置一组PWM时间点，通过PWM的上升沿和下降沿（有效沿在ADC中可以设置），来硬件触发 ADC
15:0	CCRS1	R/W	0	ADC 硬件触发时间设置： 此位用来设置 ADC 硬件触发点。 通过此组寄存器，设置一组PWM时间点，通过PWM的上升沿和下降沿（有效沿在ADC中可以设置），来硬件触发 ADC

16.4 使用说明

16.4.1 计数器模式

- 往上计数
 - 在往上计数模式中，counter 从 0 计数到自动重载值，然后重新到 0 开始计数，并产生中断。而且在此时，UEV 事件发生。
 - 当 UEV 事件发生时，芯片内部加载寄存器才会被更新。
- 往下计数
 - 在往下计数模式中，counter 从自动重载值计数到 0，然后重新到自动重载值开始计数，并产生中断。而且在此时，UEV 事件发生。
 - 当 UEV 事件发生时，芯片内部加载寄存器才会被更新。
- 中央对齐模式（上下计数）
 - 在中央对齐模式中，counter 从 0 计数到自动重载值-1，产生中断；然后又从自动重载值

计数到 1，产生中断；然后又从 0 开始计数。当 counter 处于中央对齐模式时，DIR 寄存器无效。

- 每次向上溢出和向下溢出时，UEV 事件发生。
- 当 UEV 事件发生时，芯片内部加载寄存器才会被更新。

16.4.2 输入捕获模式

在输入捕获模式中，当在相应的 ICx 信号出现触发沿的时候，捕捉寄存器（CCR）会把当时的 counter 值保存下来。当一次捕获发生后，相应的中断标志被置位，同时产生一次捕获中断。CCIF 由软件清 0。触发变化沿可以由寄存器控制是上升沿或下降沿。捕捉源可以选择滤波或不滤波

16.4.3 PWM 模式

PWM 模式可以产生波形，其频率取决于 ARR 寄存器和 PSC，而占空比取决于 CCR 寄存器。

- PWM 边沿对齐模式

- 向上计数

向上计数的情况下，配置 GTIM_CCER.CCP 为 0 时，OCxREF 信号在 $CNT < CCR$ 时为高电平，否则为低电平。如果 CCR 值大于 ARR 值，则 OCxREF 被锁定为 1；如果 CRR 为 0，则 OCxREF 被固定为 0。下图为 ARR=7 时的 PWM 波形实例

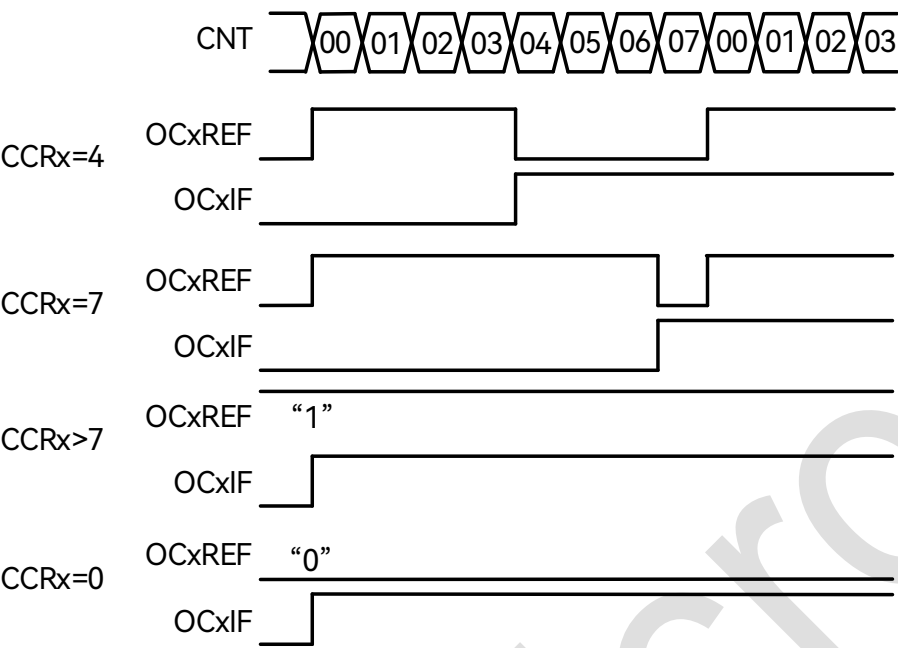


图 16-1: PWM 向上计数时序图

- 向下计数

向下计数的情况下，配置 GTIM_CCER.CCP 为 0 时，OCxREF 信号在 CNT>CCR 时为低电平，否则为高电平。如果 CCR 值大于 ARR 值，则 OCxREF 被锁定为 1；如果 CRR 为 0，则 OCxREF 被固定为 0
- PWM 中央对齐模式

根据 GTIM_CR.CMS 位的配置，比较标志可以在计数器向上计数时置 1、比较标志可以在计数器向下计数时置 1、比较标志可以在计数器向上和向下计数时置 1。ARR=7 时的 PWM 波形实例，如图 16-1 所示。

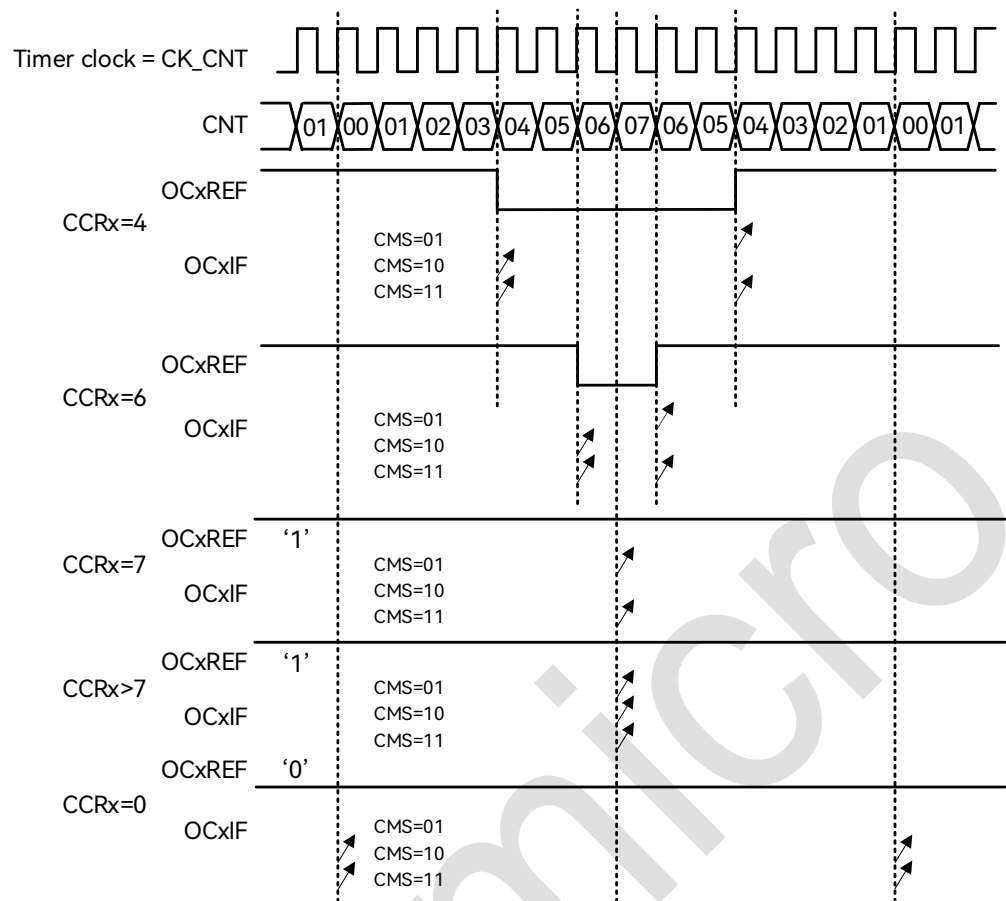


图 16-1: PWM 中央对齐时序图

16.4.4 互补输出和死区插入

互补输出: Gtimer0、1、2 均可输出两路互补 PWM, 配置 GTIM_CCR.CCR 和 GTIM_CCR.CCRN, 使能 GTIM_CR.PWM_DEAD 后, 可输出互补信号 OCx 和 OCxN。(注: 使能了 GTIM_CR.PWM_DEAD 后, 会以 CCRN 和 CCR 的最小值去做 CCR, 以 ARRn 和 ARR 的最大值做 ARR)。

死区插入: 死区时间由 GTIM_ARR.ARR 和 GTIM_ARR.ARRN 的差值、GTIM_CCR.CCR 和 GTIM_CCR.CCRN 的差值决定。(注: 当 ARR=2 CCR=1 这种情况下不能调节出死区)。

16.4.5 刹车功能

可使用软件刹车或硬件刹车功能, 且可选择多种片内外设触发刹车功能。刹车触发后 PWM 的极性状态由 GTIM_CR.PWMS_B_S 和 GTIM_CR.PWMN_B_S 决定。刹车生效后, GTIM_CR.MOE 会被置 0, 若需要重新输出 PWM, 则需要重新手动将 GTIM_CR.MOE 置 1。

16.5 使用流程

- 若想将 GTIM_ARR、GTIM_CCR、GTIM_PSC 的值立即载入到 shadow 寄存器中，则写入后手动将 GTIM_EGR 写 1 触发 Update 事件，并清除 GTIM_SR.UIF 状态。
- 使能了 GTIM_CR.ARPE 后，新的 GTIM_ARR、GTIM_CCR、GTIM_PSC 值会在触发 Update 事件后才会载入到 shadow 寄存器中。

16.5.1 普通定时器

1. 配置 GTIM_CCMR.TSSEL，选择计数时钟源。
2. 配置 GTIM_ARR.ARR，设置重载值。
3. 配置 GTIM_PSC.PSC，设置预分频值。
4. 配置 GTIM_EGR 为 1，手动产生 Update 事件将 ARR 和 PSC 的值立即载入到 shadow 寄存器，并清除 GTIM_SR.UIF。
5. 配置 GTIM_CR.DIR 和 GTIM_CR.CMS，设置计数方向。
6. 配置 GTIM_CR.CEN，启动 Gtimer 计数。

16.5.2 PWM 输出

1. 根据 IO 复用关系，将 IO 复用为 GTIM_CH 和 GTIM_CHN。
2. 配置 GTIM_CCMR.TSSEL，选择计数时钟源。
3. 配置 GTIM_ARR.ARR，设置重载值。
4. 配置 GTIM_CCR.CCR，设置比较值。
5. 配置 GTIM_PSC.PSC，设置预分频值。
6. 若想输出互补 PWM，则设置 GTIM_ARR.ARRN 和 GTIM_CCR.CCRN，并使能 GTIM_CR.PWM_DEAD。
7. 配置 GTIM_EGR 为 1，手动产生 Update 事件将 ARR 和 PSC 的值立即载入到 shadow 寄存器，

并清除 GTIM_SR.UIF。

8. 配置 GTIM_CR.DIR 和 GTIM_CR.CMS，设置计数方向。
9. 配置 GTIM_CCMR.CCS 为 0，设置通道为输出。
10. 配置 GTIM_CCER.CCP，设置通道输出极性。
11. 配置 GTIM_CCER.CCE 为 1，使能 OC 通道输出。
12. 配置 GTIM_CR.CEN，启动 Gtimer 计数。
13. 配置 GTIM_CR.MOE 为 1，使能总输出。

16.5.3 输入捕获

1. 根据 IO 复用关系，将 IO 复用为 GTIM_CH。
2. 配置 GTIM_CCMR.CAPSEL，设置捕捉源。
3. 配置 GTIM_CCMR.ICPSC，设置捕捉源分频。
4. 配置 GTIM_CCMR.CAPEDGE，设置捕捉源触发方式。
5. 配置 GTIM_CCMR.TSSEL，选择计数时钟源。
6. 配置 GTIM_ARR.ARR，设置重载值。
7. 配置 GTIM_PSC.PSC，设置预分频值。
8. 配置 GTIM_EGR 为 1，手动产生 Update 事件将 ARR 和 PSC 的值立即载入到 shadow 寄存器，并清除 GTIM_SR.UIF。
9. 配置 GTIM_CR.DIR 和 GTIM_CR.CMS，设置计数方向。
10. 配置 GTIM_CCMR.CCS 为 1，设置通道为输入。
11. 配置 GTIM_CCER.CCE 为 1，使能捕捉功能。
12. 若使用捕捉中断，配置 GTIM_IER.CCIE 为 1，使能捕捉中断。
13. 配置 GTIM_CR.CEN，启动 Gtimer 计数。

16.5.4 刹车功能

- 硬件刹车使用流程

1. 根据 IO 复用关系，将 IO 复用为 GTIM_BK。
2. 配置 GTIM_CR.BREAK1_SEL 和 GTIM_CR.BREAK2_SEL，选择刹车源。
3. 配置 GTIM_CR.BKE_POL，设置刹车信号极性。
4. 配置 GTIM_CR.PWMN_B_S 和 GTIM_CR.PWMS_B_S，刹车触发后 PWM 的极性状态。
5. 若使用刹车中断，配置 GTIM_IER.BKE_IE 为 1，使能刹车中断。
6. 配置 GTIM_CR.BKE 为 1，使能刹车功能。

- 软件刹车使用流程

1. 配置 GTIM_CR.BREAK1_SEL 和 GTIM_CR.BREAK2_SEL，选择刹车源。
2. 配置 GTIM_CR.BKE_POL，设置刹车信号极性。
3. 配置 GTIM_CR.PWMN_B_S 和 GTIM_CR.PWMS_B_S，刹车触发后 PWM 的极性状态。
4. 若使用刹车中断，配置 GTIM_IER.BKE_IE 为 1，使能刹车中断。
5. 配置 GTIM_CR.BKE 为 1，使能刹车功能。
6. 配置 GTIM_CR.SOFT_BK 为 1，软件触发刹车信号。

17 LPTIMER

17.1 概述

LPTIMER 是 16 位的低功耗定时/计数器模块。通过选择合适的工作时钟，在各种低功耗模式下保持运行，并且只消耗很低的功耗。可以在没有内部时钟的条件下工作，因此可实现休眠模式下的外部脉冲计数功能。此外，与外部输入的触发信号结合，可以实现低功耗超时唤醒功能。

17.2 主要特性

- 3 个独立的 16bit 向上计数器，LPTimer0, LPTimer1, LPTimer2
- 3bit 异步时钟预分频器，8 种分频系数（1、2、4、8、16、32、64、128）
- 可选工作时钟：
 - 内部时钟源：LSCLK（CLK32K）、RCLP（CLK1HZ）、PCLK
 - 外部时钟源：LPTIN（带有模拟滤波）
- 16bit 比较寄存器
- 16bit 目标值寄存器
- 软件/硬件触发
- 输入极性选择
- 无时钟外部脉冲计数
- 外部触发的休眠超时唤醒
- 支持 16bit PWM

17.3 寄存器描述

LPTIMER0 寄存器基地址：0x40001000

LPTIMER1 寄存器基地址：0x40002800

LPTIMER2 寄存器基地址：0x40002C00

表 17-1: LPTIMER 寄存器列表

偏置	名称	描述
0x00	LPTM_CFG	LPTIM 配置寄存器
0x04	LPTM_CNT	LPTIM 计数寄存器
0x08	LPTM_COMP	LPTIM 比较值寄存器
0x0C	LPTM_TARGET	LPTIM 目标值寄存器
0x10	LPTM_IE	LPTIM 中断使能寄存器
0x14	LPTM_IF	LPTIM 中断标志寄存器
0x18	LPTM_CTRL	LPTIM 控制寄存器

17.3.1 配置寄存器 LPTM_CFG（偏移：00H）

比特	名称	属性	默认值	功能描述
31:13	RSV	-	-	保留
12:10	DIVSEL	R/W	0	计数时钟分频选择： 000：1 分频 001：2 分频 010：4 分频 011：8 分频 100：16 分频 101：32 分频 110：64 分频 111：128 分频

比特	名称	属性	默认值	功能描述
9:8	CLKSEL	R/W	10	时钟源选择: 00: LSCLK (由 SYSREG->SYSCTRL0[12]选择的系统低速时钟 CLK32K, 即 RCL) 作为计数时钟 01: RCLP 作为计数时钟 (LPTIMER0 模块选择 RTC_1Hz 作为 RCLP; LPTIMER1 模块选择 LPTIMER0_OUT 输出作为 RCLP; LPTIMER2 模块选择 LPTIMER1_OUT 输出作为 RCLP) 10: PCLK 的门控时钟 (PCLK) 作为计数时钟 11: LPTIN (由 SYSREG->SYSCTRL0[21]选择的外部引脚或 CLK_1HZ) 作为计数时钟
7	EDGESEL	R/W	0	LPTIN 输入边沿选择: 0: LPTIN 的上升沿计数 1: LPTIN 的下降沿计数
6:5	TRIGCFG	R/W	0	外部触发边沿选择: 00: 外部输入信号上升沿 trigger 01: 外部输入信号下降沿 trigger 10/11: 外部输入信号上升下降沿 trigger
4	POLARITY	R/W	0	计数时钟分频选择: 0: 正极性波形, 即第一次计数值=比较值时产生输出波形上升沿 1: 负极性波形, 即第一次计数值=比较值时产生输出波形下降沿
3	PWM	R/W	0	脉宽调制模式: 0: 周期方波输出模式 1: PWM 输出模式

比特	名称	属性	默认值	功能描述
2	MODE	R/W	0	计数模式： 0：连续计数模式：计数器被触发后保持运行，直到被关闭为止。计数器达到目标值后回到 0 重新开始计数，并产生溢出中断。 1：单次计数模式：计数器被触发后计数到目标值后回到 0，并自动停止，产生溢出中断。
1:0	TMODE	R/W	0	工作模式选择： 00：带波形输出的普通定时器模式 01：Trigger 脉冲触发计数模式 10：外部异步脉冲计数模式 11：Timeout 模式

17.3.2 计数值寄存器 LPTM_CNT（偏移：04H）

比特	名称	属性	默认值	功能描述
31:16	RSV	-	-	保留
15:0	CNT16	R	0	计数器数值

17.3.3 比较寄存器 LPTM_COMP（偏移：08H）

比特	名称	属性	默认值	功能描述
31:16	RSV	-	-	保留
15:0	COMP_CCR_REG	R/W	0	比较值寄存器

17.3.4 目标寄存器 LPTM_TARGET（偏移：0CH）

比特	名称	属性	默认值	功能描述
31:16	RSV	-	-	保留
15:0	TARGET_REG	R/W	0	目标值寄存器

17.3.5 中断使能寄存器 LPTM_IE (偏移: 10H)

比特	名称	属性	默认值	功能描述
31:3	RSV	-	-	保留
2	TRIGIE	R/W	0	外部触发到来中断使能位: 0: 外部触发到来中断禁止 1: 外部触发到来中断使能
1	OVIE	R/W	0	计数器溢出中断使能位: 0: 计数器溢出中断禁止 1: 计数器溢出中断使能
0	COMPIE	R/W	0	比较匹配中断使能位: 0: 计数器值和比较值匹配中断禁止 1: 计数器值和比较值匹配中断使能

17.3.6 中断标志寄存器 LPTM_IF (偏移: 14H)

比特	名称	属性	默认值	功能描述
31:3	RSV	-	-	保留
2	TRIGIF	R/W	0	外部触发到来中断标志位, 写 1 清零: 0: 无中断产生 1: 外部触发到来中断产生 注: 使用 lptime2 时, 需要先将该标志位手动清 0
1	OVIF	R/W	0	计数器溢出中断使能位, 写 1 清零: 0: 无中断产生 1: 计数器溢出中断产生
0	COMPIF	R/W	0	比较匹配中断使能位, 写 1 清零: 0: 无中断产生 1: 计数器值和比较值匹配中断产生

17.3.7 控制寄存器 LPTM_CTRL (偏移: 18H)

比特	名称	属性	默认值	功能描述
31:2	RSV	-	-	保留
1	CEN_ALL_EN	R/W	0	GTIMER0 中 CEN_ALL 控制使能: 0: 当前 LPTIMER 计数不受 CEN_ALL 控制 1: 当前 LPTIMER 计数开始受 CEN_ALL 控制
0	LPTEN	R/W	0	LPTIM 使能位: 0: 禁止计数器计数 1: 使能计数器计数

17.4 使用流程

17.4.1 普通定时器

1. 配置 LPTM_CFG.CLKSEL, 选择时钟源。
2. 配置 LPTM_CFG.DIV, 设置分频值。
3. 配置 LPTM_CFG.MODE, 设置计数模式。
4. 配置 LPTM_CFG.TMODE, 选择普通定时器模式。
5. 配置 LPTM_TARGET 目标寄存器值。
6. 使能 LPTM_IE 中断寄存器, 选择溢出中断。
7. 使能 LPTM_CTRL.LPTEN 位, 启动计数器。

17.4.2 PWM 输出

1. 配置 LPTM_CFG.CLKSEL, 选择时钟源。
2. 配置 LPTM_CFG.DIV, 设置分频值。
3. 配置 LPTM_CFG.MODE, 设置计数模式。
4. 配置 LPTM_CFG.PWM, 选择 PWM 输出模式。

5. 配置 LPTM_CFG.POLARITY，选择波形极性。
6. 配置 LPTM_CFG.TMODE，选择普通定时器模式。
7. 配置 LPTM_COMP 比较寄存器值。
8. 配置 LPTM_TARGET 目标寄存器值。
9. 使能 LPTM_IE 中断寄存器，打开中断。
10. 使能 LPTM_CTRL.LPTEN 位，启动计数器。

17.4.3 Trigger 脉冲触发计数模式

1. 配置 LPTM_CFG.CLKSEL，选择时钟源。
2. 配置 LPTM_CFG.DIV，设置分频值。
3. 配置 LPTM_CFG.MODE，设置计数模式。
4. 配置 LPTM_CFG.TRIGCFG，设置外部触发边沿。
5. 配置 LPTM_CFG.TMODE，选择 Trigger 脉冲触发计数模式。
6. 使能 LPTM_IE.TRIGIE 中断寄存器，打开外部触发中断。
7. 使能 LPTM_CTRL.LPTEN 位，启动计数器。

17.4.4 外部异步脉冲计数模式

1. 配置 LPTM_CFG.CLKSEL，选择时钟源为 LPTIN。
2. 配置 LPTM_CFG.DIV，设置分频值。
3. 配置 LPTM_CFG.MODE，设置计数模式。
4. 配置 LPTM_CFG.EDGESEL，设置 LPTIN 输入边沿。
5. 配置 LPTM_CFG.TMODE，选择外部异步脉冲计数模式。
6. 配置 LPTM_TARGET 目标寄存器值。
7. 使能 LPTM_IE 中断寄存器，打开中断。
8. 使能 LPTM_CTRL.LPTEN 位，启动计数器。

17.4.5 Timeout 模式

1. 配置 LPTM_CFG.CLKSEL，选择时钟源 LSCLK。
2. 配置 LPTM_CFG.DIV，设置分频值。
3. 配置 LPTM_CFG.MODE，设置计数模式。
4. 配置 LPTM_CFG.TRIGCFG，设置外部触发边沿。
5. 配置 LPTM_CFG.TMODE，选择 Timeout 模式。
6. 配置 LPTM_TARGET 目标寄存器值。
7. 使能 LPTM_IE 中断寄存器，打开溢出中断。
8. 使能 LPTM_CTRL.LPTEN 位，启动计数器。

注：计数器溢出前没有出现新的 trigger，则产生溢出中断并停止计数，并清除使能，如果要重新使用，需要再次使能该中断。

18 OPA

18.1 概述

OPA 是一款具有轨到轨输入和 AB 类输出级的运算放大器。输入输出端可以根据需要配置成不同连接。

18.2 主要特性

- 一个运算放大器
- 电压范围：2.0 ~ 3.6V
- 工作电流：1mA

18.3 功能框图

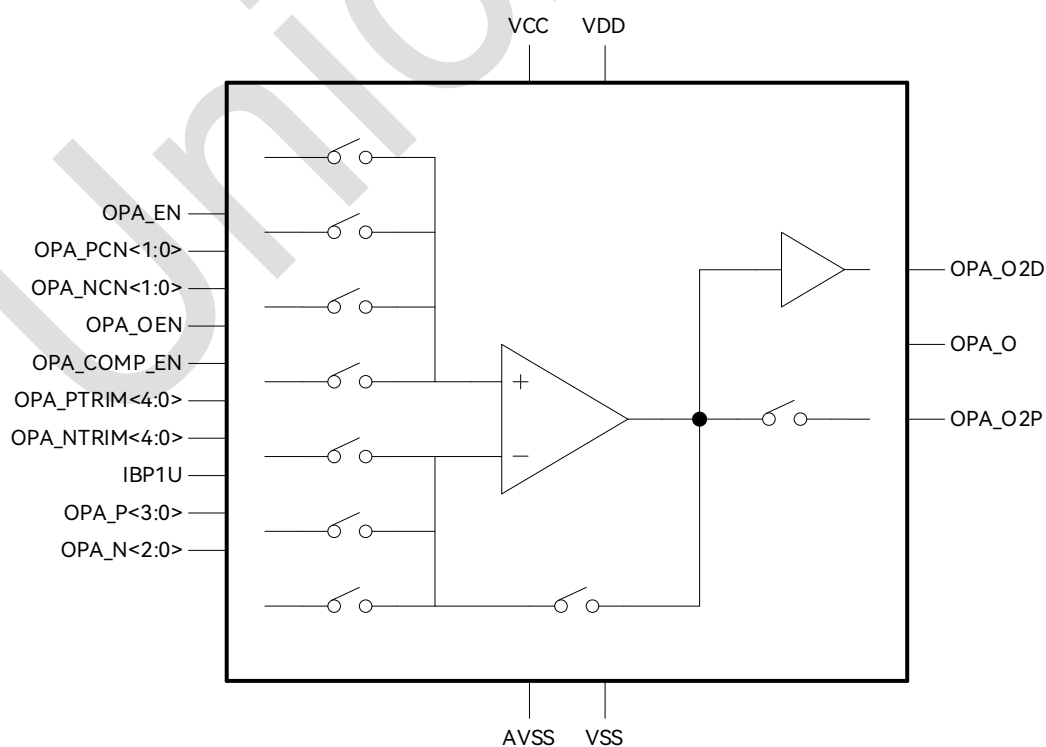


图 18-1: OPA 功能框图

18.4 寄存器描述

寄存器基地址：0x40002000

表 18-1：OPA 寄存器列表

偏置	名称	描述
0x088	OPA_CFG	OPA 控制寄存器
0x0C8	OPA_STATUS	放大器状态寄存器

18.4.1 OPA 控制寄存器 OPA_CFG（偏移：088H）

比特	名称	属性	复位值	描述
31:12	RSV	-	-	保留
11	OPA_LVEN	RW	0	OPA 作为比较器滤波使能信号： 0：滤波关闭 1：滤波使能
10:9	OPA_LVSET	RW	0	OPA 作为比较器滤波时间设置： 00：输出滤波 2 个 32K 系统低速时钟 01：输出滤波 4 个 32K 系统低速时钟 10：输出滤波 8 个 32K 系统低速时钟 11：输出滤波 16 个 32K 系统低速时钟
8	OPA_POL	RW	1	OPA 作为比较器使用时极性选择： 0：N 端电压高时输出 1 1：P 端电压高时输出 1
7	OPA_INTEN	RW	0	OPA 作为比较器使用时，中断使能： 0：OPA 中断不使能 1：OPA 中断使能
6:5	OPA_PCN	RW	0	OPA 正端信号选择： 00：选择 ADC 输入 01：OPA_P0 10：OPA_P1 11：OPA_P2
4	OPA_OEN	RW	0	OPA 输出使能： 0：OPA 输出关闭 1：OPA 输出使能

比特	名称	属性	复位值	描述
3:2	OPA_NCN	RW	0	OPA 负端信号选择: 00: OPA_N0 01: OPA_N1 10: OPA_N2 11: 内部连接 OPA 输出
1	OPA_EN	RW	0	OPA 使能: 0: 关闭 OPA 功能 1: 开启 OPA 功能
0	OPA_COMP_EN	RW	0	OPA 比较器功能使能: 0: 关闭 OPA 比较器功能 1: 开启 OPA 比较器功能

18.4.2 放大器状态寄存器 OPA_STATUS (偏移: 0C8H)

比特	名称	属性	复位值	描述
31:2	RSV	-	-	保留
1	OPA_CST	R	0	OPA 实时状态寄存器: 0: 当前 OPA 没有真事件 1: 当前 OPA 有真事件发生
0	OPA_INTR	RW	0	OPA 中断状态: 0: OPA 未发生中断 1: OPA 发生中断 写 1 清 0

18.5 OPA 使用流程

1. 配置 OPA_CFG 的 OPA_EN, 使能 OPA 功能。
2. 配置 PAD_ADS 寄存器, 配置 OPA 的输入引脚为模拟接口。
3. 配置 OPA_CFG 的 OPA_PCN, 配置 OPA 正端的信号选择 (如 ADC 使用 OPA 缓冲则选择内部接 ADC 输入)。
4. 配置 OPA_CFG 的 OPA_NCN, 配置 OPA 负端的信号选择 (如选择负端通过内部连接输出则 OPA 是作为信号跟随器, 不放大电压)。
5. 配置 OPA_CFG 的 OPA_ENO, 使能 OPA 输出。

18.6 OPA 作为 COMP 使用流程

1. 配置 OPA_CFG 的 OPA_EN，使能 OPA 功能。
2. 配置 PAD_ADS 寄存器，配置 OPA 的输入引脚为模拟接口。
3. 配置 OPA_CFG 的 OPA_PCN，配置 OPA 正端的信号选择。
4. 配置 OPA_CFG 的 OPA_NCN，配置 OPA 负端的信号选择。
5. 配置 OPA_CFG 的 OPA_ENO，使能 OPA 输出。
6. 配置 OPA_CFG 的 OPA_COMP_EN，使能 OPA 作为 COMP 功能。
7. 配置 OPA_CFG 的 OPA_POL，选择 OPA 作为 COMP 时的极性。
8. 配置 OPA_CFG 的 OPA_LVSET，设置滤波时间。
9. 配置 OPA_CFG 的 OPA_LVEN，使能滤波。
10. 配置 OPA_CFG 的 OPA_INTEN，使能 OPA 中断。
11. 根据极性选择，当 P 端或者 N 端的电压比另外一个高时，会触发中断。

19 COMP

19.1 概述

COMP 是一款具有轨到轨输入的迟滞比较器。输入端根据需要配置。

比较器用作电压比较，有两个输入端 IN+和 IN-，可选择其中一个输入端作为参考点来比较，当另一输入端电压小于参考电压时比较器输出低电平，反之输出高电平。

19.2 主要特性

- 1 个电压比较器
- 可产生比较中断
- 比较器精度可调
- 电压范围：2.0 ~ 3.6V
- 工作电流：4.6 μ A

19.3 功能框图

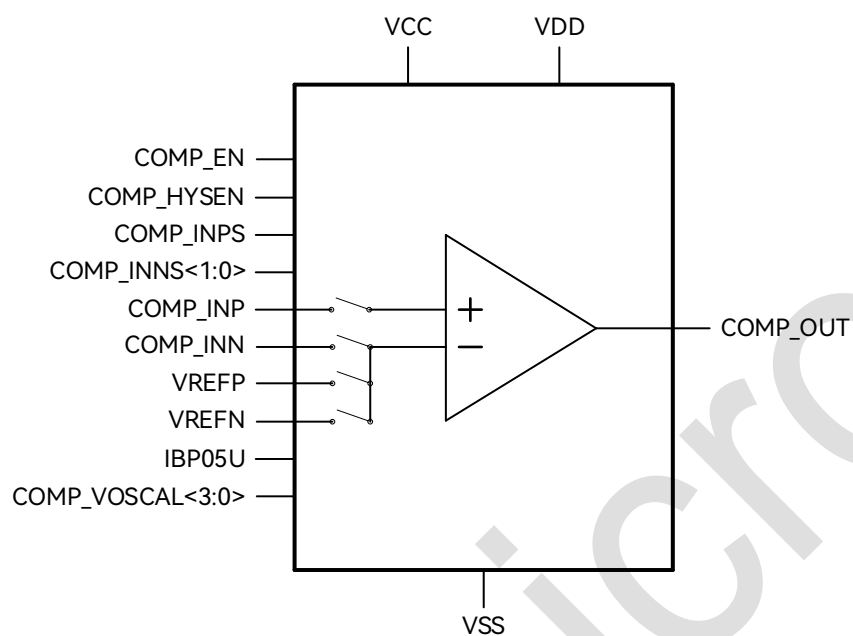


图 19-1: COMP 功能框图

19.4 寄存器描述

寄存器基地址：0x40002000

表 19-1：COMP 寄存器列表

偏置	名称	描述
0x08C	COMP_CFG	COMP 控制寄存器
0x0C4	COMP2_STATUS	比较器 2 状态寄存器

19.4.1 COMP 控制寄存器 COMP_CFG（偏移：08CH）

比特	名称	属性	复位值	描述
31:30	RSV	-	-	保留
29	COMP2_LVEN	RW	0	比较器 2 滤波使能信号： 0：滤波关闭 1：滤波使能
28:27	COMP2_LVSET	RW	0	比较器 2 滤波时间设置： 00：比较器 2 输出滤波 2 个 32K 时钟 01：比较器 2 输出滤波 4 个 32K 时钟 10：比较器 2 输出滤波 8 个 32K 时钟 11：比较器 2 输出滤波 16 个 32K 时钟
26	COMP2_POL	RW	1	COMP2 极性选择： 0：COMP2 N 端电压高时输出 1 1：COMP2 P 端电压高时输出 1
25	COMP2_INPS	RW	0	CM2 正端信号选择： 0：VREFN 1：COMP2_INP
24:23	COMP2_INNS	RW	0	COMP2 负端信号选择： 00：VREFN 01：VREFP 10/11：COMP2_INN
22	COMP2_HYSEN	RW	0	使能 COMP2 迟滞滤波： 0：关闭比较器迟滞滤波功能 1：使能比较器迟滞滤波功能；
21	COMP2_EN	RW	0	COMP2 使能： 0：关闭 COMP2 1：使能 COMP2

比特	名称	属性	复位值	描述
20	COMP2_CALEN	RW	0	COMP2 校准使能： 0：关闭 COMP2 校准功能 1：使能 COMP2 校准功能
19:0	RSV	-	-	保留

19.4.2 COMP2 寄存器 COMP2_STATUS（偏移：0C4H）

比特	名称	属性	复位值	描述
31:3	RSV	-	-	保留
2	COMP2_INTEN	RW	0	COMP2 中断使能 0：COMP2 中断关闭 1：COMP2 中断使能
1	COMP2_CST	R	0	COMP2 实时状态寄存器 0：当前 COMP2 没有真事件 1：当前 COMP2 有真事件发生
0	COMP2_INTR	RW	0	COMP2 中断状态 0：COMP2 未发生中断 1：COMP2 发生中断 写 1 清 0

19.5 使用流程

1. 配置 IO 复用为 COMP_OUT，作为输出端。
2. 配置 PAD_ADS 寄存器，将两个输入端 IO 配置为模拟接口。
3. 配置 COMP_CFG 寄存器，配置 COMP 极性、正端信号选择、负端信号选择。
4. 配置 COMP2_STATUS 寄存器，使能中断、清中断状态标志。
5. 配置 COMP_CFG 寄存器，使能 COMP。

20 ADC

20.1 概述

这是一个 12 位的 ADC 逐次接近型模数转换器。它具有多达 4 个输入通道，可测量来自 3 个外部源的信号和 1 个内部 LDO 输出。这些通道的 A/D 转换可在单次或连续扫描模式下进行。ADC 控制器实现 CPU 和 SAR ADC 之间的通信。ADC 转换的结果存储在数据寄存器的低 12 位。

20.2 主要特性

- 支持 DMA 传输模式
- 16 位的可编程分频器，用于产生 A/D 时钟
- 支持 12 位分辨率 A/D 输入数据，最大采样率为 1MSps，采样率可通过软件配置
- 支持 4 通道 ADC 输入:3 个引脚通道、1 个内部 LDO 输出
- 支持关闭模拟 ADC
- 支持轮询（poll）和中断（interrupt）传输模式
- 支持单次扫描或连续扫描模式
- 中断源：通道数据有效（4 个通道各有一个中断源）、FIFO 满（16 个数据）、FIFO 数据量达到设定值（1 或 8 个数据）
- 支持片内外设触发 ADC 转换
- ADC 电压输入范围：0 ~ Vref
- ADC 参考电压源可选择：芯片供电电压 VDDH、IO 引脚外接电压 VREFIO、内置 VREF

20.3 寄存器描述

ADC 寄存器基地址：0x40001C00

表 20-1: ADC 寄存器列表

偏置	名称	描述
0x00	ADC_GCR	ADC 通用控制寄存器
0x04	ADC0_DR	A/D 通道 0 数据寄存器
0x10	ADC3_DR	A/D 通道 3 数据寄存器
0x14	ADC4_DR	A/D 通道 4 数据寄存器
0x18	ADC5_DR	A/D 通道 5 数据寄存器
0x24	ADC_CDR	A/D 时钟分频寄存器
0x28	ADC_ISR	A/D中断状态寄存器
0x2C	ADC_IER	A/D 中断使能寄存器
0x30	ADC_ICR	A/D 中断清除寄存器
0x34	ADC_COUNT	A/D 切换间隔计数寄存器
0x38	ADC_RXREG	A/D 接收数据寄存器
0x3C	ADC_CSTAT	ADC 状态寄存器
0x40	ADC_SPW	ADC 采样脉宽寄存器
0x44	ADC_ICRL	模拟 ADC 配置寄存器
0x48	ADC_HDT	ADC 硬件触发使能配置寄存器

20.3.1 ADC 通用控制寄存器 ADC_GCR（偏移：000H）

比特	名称	属性	复位值	描述
31:25	RSV	-	-	保留
24	DATA_SAMP_NEG	R/W	0x0	ADC 数据在 EOC 信号的边沿采样选择： 0：ADC 数据在 EOC 的上升沿被采样 1：ADC 数据在 EOC 的下降沿被采样 注：在本芯片设计中此位只能设置为 0

比特	名称	属性	复位值	描述
23:16	CH_EN[7:0]	R/W	0x0	启用相关 ADC 通道进行模数转换。默认值：0。 0：通道禁用 1：通道启用 Bit[16]: ch_en[0] 通道 0 使能信号 Bit[19]: ch_en[3] 通道 3 使能信号. Bit[20]: ch_en[4] 通道 4 使能信号 Bit[21]: ch_en[5] 通道 5 使能信号 注： 1) 在单次扫描模式下，每个通道只进行 1 次模数转换。例如，需要选择通道 0、3、5 进行模数转换，则设置 ch_en[7:0] = 8'b0010 1001，通道 5 转换结束后操作完成。 2) 在连续扫描模式下，按通道编号从低到高的顺序，重复循环地进行模数转换。例如，ch_en[7:0] = 8'b0010 1001，则选择了通道 0、3、5 一共 3 个通道。通道 0 将首先执行模数转换，接着通道 3 执行，然后通道 5 执行，再循环回通道 0。
15:11	RSV	-	-	保留
10	ADC_START_EN	R/W	0x0	ADC 转换开始使能信号。当信号从低到高转换时，ADC 转换开始。当信号从高到低转换时，ADC 转换完成。此位在 ADC_EN=0 时清零。默认值为 0。
9	ADC_RST	R/W	1	ADC 内部模拟缓冲器复位信号： 0：SAR ADC 正常工作 1：SAR ADC 复位
8	ADC_PD_EN	R/W	1	SAR ADC 掉电使能信号： 0：SAR ADC 上电 1：SAR ADC 掉电
7	RSV	-	-	保留
6	ADC_CLK_SEL	R/W	0	A/D 时钟源选择信号： 0：ADC 时钟由内部时钟分频器产生 1：ADC 时钟由系统时钟产生
5	ADC_RCLR_EN	R/W	0	ADC 数据寄存器读取后清除使能： 0：禁止 ADC 数据寄存器读取后清除 1：使能 ADC 数据寄存器读取后清除

比特	名称	属性	复位值	描述
4	RXTLF	R/W	0	RX FIFO 中断触发条件位: 0: RX FIFO 里有 1 个或以上有效数据 1: RX FIFO 里有 8 个或以上有效数据
3	RXFIFO_EN	R/W	0	RX FIFO 使能位: 0: RX FIFO 禁用, 刷新 RX FIFO 中的数据 1: RX FIFO 使能
2	DMAMODE	R/W	0	DMA 访问模式位: 0: CPU 访问模式(只有 CPU 能访问 RX FIFO) 1: DMA 访问模式(只有 DMA 能访问 RX FIFO)
1	CONTINUOUS	R/W	0	ADC 工作模式选择位。默认值为 0。 0: 单次扫描模式 1: 连续扫描模式
0	ADC_EN	R/W	0	ADC 控制器使能信号。默认值为 0 0: 禁用 ADC 模块 1: 使能 ADC 模块

20.3.2 A/D 通道 0 数据寄存器 ADC0_DR (偏移: 004H)

比特	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15	DATA_VALID0	R	0	数据有效位。此位在 ADC_EN=0 时清除, 或者当 ADC_RCLR_EN=1 时的读取操作后清除。 0: 数据无效 1: 数据有效
14:12	RSV	-	-	保留
11:0	CH0_DATA	R	0x000	A/D 通道 0 接收数据寄存器。 当 ADC_RCLR_EN=1 时进行读取操作后, 此位将被清除

20.3.3 A/D 通道 3 数据寄存器 ADC3_DR (偏移: 010H)

比特	名称	属性	复位值	描述
31:16	RSV	-	-	保留

比特	名称	属性	复位值	描述
15	DATA_VALID 3	R	0	数据有效位。此位在ADC_EN=0时清除，或者当ADC_RCLR_EN=1时的读取操作后清除。 0：数据无效 1：数据有效
14:12	RSV	-	-	保留
11:0	CH3_DATA	R	0x000	A/D 通道 3 接收数据寄存器 当 ADC_RCLR_EN=1 时进行读取操作后，此位将被清除

20.3.4 A/D 通道 4 数据寄存器 ADC4_DR (偏移：014H)

比特	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15	DATA_VALID 4	R	0	数据有效位。此位在ADC_EN=0时清除，或者当ADC_RCLR_EN=1时的读取操作后清除 0：数据无效 1：数据有效
14:12	RSV	-	-	保留
11:0	CH4_DATA	R	0x000	A/D 通道 4 接收数据寄存器 当 ADC_RCLR_EN=1 时进行读取操作后，此位将被清除

20.3.5 A/D 通道 5 数据寄存器 ADC5_DR (偏移：018H)

比特	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15	DATA_VALID 5	R	0	数据有效位。此位在ADC_EN=0 时清除，或者当ADC_RCLR_EN=1 时的读取操作后清除 0：数据无效 1：数据有效
14:12	RSV	-	-	保留
11:0	CH5_DATA	R	0x000	A/D 通道 5 接收数据寄存器。 当 ADC_RCLR_EN=1 时进行读取操作后，此位将被清除

20.3.6 ADC 时钟分频寄存器 ADC_CDR (偏移: 024H)

比特	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	CLKDIV	R/W	0x00FF	ADC 内部时钟分频寄存器 分频公式: $adc_clk = fpclk / CLKDIV$ (fpclk : APB 总线时钟) 注: 建议 CLKDIV ≥ 1 。请勿把 clkdiv 设为 0 或 1, 若把 clkdiv 设为 0 或 1, 也当作 2 分频

20.3.7 ADC 中断状态寄存器 ADC_ISR (偏移: 028H)

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留
9	RXFIFO_FULL_INTF	R	0x0	RX FIFO 满中断标志位: 0: 接收 FIFO 未滿 1: 接收 FIFO 满
8	RX_INTF	R	0x0	接收器数据可用中断标志位 当接收器 FIFO 接收到足够数据时, 此标志位置 1 (根据 RXTLF 位设置) 0: 接收器 FIFO 无可用数据 1: 接收器 FIFO 有可用数据
7:6	RSV	-	-	保留
5	CH5_INTF	R	0x0	通道 5 数据中断, 高电平有效, 软件给 ch5_int_clr 写 1 后清除: 0: 没有中断 1: 中断激活
4	CH4_INTF	R	0x0	通道 4 数据中断, 高电平有效, 软件给 ch4_int_clr 写 1 后清除: 0: 没有中断 1: 中断激活
3	CH3_INTF	R	0x0	通道 3 数据中断, 高电平有效, 软件给 ch3_int_clr 写 1 后清除: 0: 没有中断 1: 中断激活
2:1	RSV	-	-	保留

比特	名称	属性	复位值	描述
0	CH0_INTF	R	0x0	通道 0 数据中断，高电平有效，软件给 ch0_int_clr 写 1 后清除： 0：没有中断 1：中断激活

20.3.8 ADC 中断使能寄存器 ADC_IER (偏移：02CH)

比特	名称	属性	复位值	描述
31:10	RSV	-	-	保留
9	RXFIFO_FULL_IEN	R/W	0x0	RX FIFO 满中断使能（有 16 个数据时触发中断）： 0：禁止中断 1：中断使能
8	RXIEN	R/W	0x0	接收器 FIFO 中断使能（有 1 个或 8 个数据时触发中断）： 0：禁止中断 1：中断使能
7:6	RSV	-	-	保留
5	CH5_INT_EN	R/W	0x0	通道 5 数据中断使能，高电平有效： 0：禁止中断 1：中断使能
4	CH4_INT_EN	R/W	0x0	通道 4 数据中断使能，高电平有效： 0：禁止中断 1：中断使能
3	CH3_INT_EN	R/W	0x0	通道 3 数据中断使能，高电平有效： 0：禁止中断 1：中断使能
2:1	RSV	-	-	保留
0	CH0_INT_EN	R/W	0x0	通道 0 数据中断使能，高电平有效： 0：禁止中断 1：中断使能

20.3.9 ADC 中断清除寄存器 ADC_ICR (偏移: 030H)

比特	名称	属性	复位值	描述
31:10	RSV	-	-	保留
9	RXFIFO_FULL_ICLR	W	0x0	RX FIFO 满中断清除: 0: 不清除中断 1: 清除中断
8	RXICLR	W	0x0	RX FIFO 中断清除: 0: 不清除中断 1: 清除中断
7:6	RSV	-	-	保留
5	CH5_INT_CLR	W	0x0	通道 5 数据中断清除, 高电平有效: 0: 不清除中断 1: 清除中断
4	CH4_INT_CLR	W	0x0	通道 4 数据中断清除, 高电平有效: 0: 不清除中断 1: 清除中断
3	CH3_INT_CLR	W	0x0	通道 3 数据中断清除, 高电平有效: 0: 不清除中断 1: 清除中断
2:1	RSV	-	-	保留
0	CH0_INT_CLR	W	0x0	通道 0 数据中断清除, 高电平有效: 0: 不清除中断 1: 清除中断

20.3.10 ADC 切换间隔计数寄存器 ADC_COUNT (偏移: 034H)

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	ADC_COUNT	R/W	0x01	通道切换间隔时间, 这个值的单位为 ADC 时钟周期。 实际通道切换时间=(adc_count+16) * ADC 时钟周期 注: 此寄存器只能在 ADC 控制器使能前配置, 否则无效

20.3.11 ADC 接收数据寄存器 ADC_RXREG (偏移: 038H)

比特	名称	属性	复位值	描述
31:12	RSV	-	-	保留
11:0	RXFIFO_OUT	R	0x0	接收器 FIFO 的输出, SAR ADC 的值。此寄存器只读

20.3.12 ADC 状态寄存器 ADC_CSTAT (偏移: 03CH)

比特	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	RXAVL	R	0x0	CPU 轮询模式下的接收数据可用标志位。 当接收器接收到有效数据时此位置 1。 0: 接收器 FIFO 为空 1: 接收器 FIFO 有有效数据

20.3.13 ADC 采样脉宽寄存器 ADC_SPW (偏移: 040H)

比特	名称	属性	复位值	描述
31:3	RSV	-	-	保留
2:0	SAMPCLK_WIDTH	R/W	0x3	采样脉宽配置。 3: SAMPCLK 为 4 个 ADC_CLK 脉冲信号 4: SAMPCLK 为 5 个 ADC_CLK 脉冲信号 5: SAMPCLK 为 6 个 ADC_CLK 脉冲信号 6: SAMPCLK 为 7 个 ADC_CLK 脉冲信号 7: SAMPCLK 为 8 个 ADC_CLK 脉冲信号 注: 在本芯片设计中, 此寄存器应该设置大于或等于 3 的值。 若测试精度不准确可适当增加此值。此寄存器值需与 ADC_COUNT 寄存器值同时更改。注: 请谨慎设置此值, 增加此数值是以牺牲转换效率为代价的

20.3.14 模拟 ADC 配置寄存器 ADC_TCRL (偏移: 044H)

比特	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4:3	SPEED	R/W	0x0	ADC 转换速度。建议设为默认值 0
2:1	VREF_SEL	R/W	0x0	选择 ADC 参考电压源: 00/01: 选择 VDDH 作为 ADC 的参考电压源 10: 选择 VREFIO 作为 ADC 的参考电压源 11: 选择内置 VREF 作为 ADC 的参考电压源
0	USE_OPA	R/W	0x0	选择 ADC 输入通道是否经 OPA 缓冲: 0: 关闭 OPA 缓冲使能 1: 开启 OPA 缓冲使能

20.3.15 ADC 硬件触发使能配置寄存器 ADC_HDT (偏移: 048H)

比特	名称	属性	复位值	描述
31:30	RSV	-	-	保留
29:28	OPA_SET	R/W	0x0	OPA 硬件触发极性选择 (作为比较器时有真事件发生): 00: 上升沿触发 01: 下降沿触发 10/11: 双边沿触发
27:26	COMP2_SET	R/W	0x0	COMP2 硬件触发极性选择 (有真事件发生): 00: 上升沿触发 01: 下降沿触发 10/11: 双边沿触发
25:22	RSV	-	-	保留
21:20	LPTIM2_SET	R/W	0x0	LPTIM2 硬件触发极性选择 (PWM 输出) 00: 上升沿触发 01: 下降沿触发 10/11: 双边沿触发
19:18	LPTIM1_SET	R/W	0x0	LPTIM1 硬件触发极性选择 (PWM 输出) 00: 上升沿触发 01: 下降沿触发 10/11: 双边沿触发

比特	名称	属性	复位值	描述
17:16	LPTIM0_SET	R/W	0x0	LPTIM0 硬件触发极性选择 (PWM 输出) 00: 上升沿触发 01: 下降沿触发 10/11: 双边沿触发
15:14	GTIMER2_SET	R/W	0x0	GTIMER2 硬件触发极性选择 (PWM 输出) 00: 上升沿触发 01: 下降沿触发 10/11: 双边沿触发
13:12	GTIMER1_SET	R/W	0x0	GTIMER1 硬件触发极性选择 (PWM 输出) 00: 上升沿触发 01: 下降沿触发 10/11: 双边沿触发
11:10	GTIMER0_SET	R/W	0x0	GTIMER0 硬件触发极性选择 (PWM 输出) 00: 上升沿触发 01: 下降沿触发 10/11: 双边沿触发
9	OPA_HT	R/W	0x0	OPA 硬件触发使能: 0: OPA 硬件触发关闭 1: OPA 硬件触发使能
8	COMP2_HT	R/W	0x0	COMP2 硬件触发使能: 0: COMP2 硬件触发关闭 1: COMP2 硬件触发使能
7:6	RSV	-	-	保留
5	LPTIM2_HT	R/W	0x0	LPTIMER2 硬件触发使能: 0: Lptimer2 硬件触发关闭 1: Lptimer2 硬件触发使能
4	LPTIM1_HT	RW	0x0	LPTIMER1 硬件触发使能: 0: Lptimer1 硬件触发关闭 1: Lptimer1 硬件触发使能
3	LPTIM0_HT	RW	0x0	LPTIMER0 硬件触发使能: 0: Lptimer0 硬件触发关闭 1: Lptimer0 硬件触发使能
2	GTIM2_HT	RW	0x0	GTIMER2 硬件触发使能: 0: Gtimer2 硬件触发关闭 1: Gtimer2 硬件触发使能

比特	名称	属性	复位值	描述
1	GTIM1_HT	RW	0x0	GTIMER1 硬件触发使能: 0: Gtimer1 硬件触发关闭 1: Gtimer1 硬件触发使能
0	GTIM0_HT	RW	0x0	GTIMER0 硬件触发使能: 0: Gtimer0 硬件触发关闭 1: Gtimer0 硬件触发使能

20.4 ADC 使用流程

20.4.1 单次扫描模式单通道 A/D 转换

单次扫描模式下，ADC 启动转换后只执行一次转换。

1. 配置 ADC_GCR.ADCRST 和 ADC_GCR.ADC_PD_EN 为 0，模拟 ADC 复位释放和上电。
2. 配置 ADC_GCR.ADC_CLK_SEL 为 0，选择 ADC 时钟源为内部时钟分频器产生时钟。
3. 配置 ADC_CDR，设置 ADC 时钟分频。
4. 配置 ADC_SPW 和 ADC_COUNT，设置 ADC 转换速度。
5. 配置 ADC_TCRL.VREF_SEL，设置 ADC 参考电压源。
6. 配置 ADC_GCR.CONTINUOUS 为 0，选择单次扫描模式。
7. 配置 ADC_GCR.ADC_EN 为 1，启用 ADC 模块。
8. 根据 ADC 输入通道对应的 GPIO 引脚，将待转换通道配置为模拟接口（PAD_ADS）。
9. 配置 ADC_GCR.CH_EN，启用待转换的通道。
10. 配置 ADC_GCR.ADC_START_EN 为 1，启动 ADC 转换。
11. 等待 REG_ADC_ISR 的相应通道标志位为 1，读取 ADC_DR 中的数据。
12. 如需进行多次单次转换，则重复执行以上 2 个步骤（注：启动 ADC 转换前需确保 ADC_GCR.ADC_START_EN 为 0）。
13. 如使能了 ADC_IER.CH_INT_EN 和 ADC_IER.RXIEN 中断，则等待中断触发后读取 ADC_DR。

20.4.2 单次扫描模式多通道 A/D 转换

单次扫描模式下，ADC 启动转换后只执行一次转换。当同时启用多个通道时，开启 ADC 转换后，依次按通道优先级 0~7 进行转换，当优先级最低的通道转换完成，则此次单次扫描转换完成。

1. 配置 ADC_GCR.ADCRST 和 ADC_GCR.ADC_PD_EN 为 0，模拟 ADC 复位释放和上电。
2. 配置 ADC_GCR.ADC_CLK_SEL 为 0，选择 ADC 时钟源为内部时钟分频器产生时钟。
3. 配置 ADC_CDR，设置 ADC 时钟分频。
4. 配置 ADC_SPW 和 ADC_COUNT，设置 ADC 转换速度。
5. 配置 ADC_TCRL.VREF_SEL，设置 ADC 参考电压源。
6. 配置 ADC_GCR.RXFIFO_EN 为 1，使能 RX FIFO。
7. 配置 ADC_GCR.CONTINUOUS 为 0，选择单次扫描模式。
8. 配置 ADC_GCR.ADC_EN 为 1，启用 ADC 模块。
9. 根据 ADC 输入通道对应的 GPIO 引脚，将待转换通道配置为模拟接口（PAD_ADS）。
10. 配置 ADC_GCR.CH_EN，启用待转换的通道。
11. 配置 ADC_GCR.ADC_START_EN 为 1，启动 ADC 转换。
12. 等待转换优先级最低的通道的 REG_ADC_ISR 的相应通道标志位为 1 后，再读取每个通道的 ADC_DR 中的数据。
13. 如需进行多次单次转换，则重复执行以上 2 个步骤（注：启动 ADC 转换前需确保 ADC_GCR.ADC_START_EN 为 0）。
14. 如使能了 ADC_IER.CH_INT_EN，则等待中断触发后读取 ADC_DR 中的数据。

注：

- 在使用多通道转换时，可以通过判断最低优先级通道数据有效后，再去读取每个通道的 ADC_DR 中的数据
- 在使用多通道转换时，若采样精度不足，建议适当修改 ADC_SPW 和 ADC_COUNT 的值

20.4.3 连续扫描模式单通道 A/D 转换

连续扫描模式下，启动一次 ADC 转换后会对所选通道进行不断地连续转换，将 ADC_GCR.ADC_START_EN 写 0 可停止转换。

1. 配置 ADC_GCR.ADCRST 和 ADC_GCR.ADC_PD_EN 为 0，模拟 ADC 复位释放和上电。
2. 配置 ADC_GCR.ADC_CLK_SEL 为 0，选择 ADC 时钟源为内部时钟分频器产生时钟。
3. 配置 ADC_CDR，设置 ADC 时钟分频。
4. 配置 ADC_SPW 和 ADC_COUNT，设置 ADC 转换速度。
5. 配置 ADC_TCRL.VREF_SEL，设置 ADC 参考电压源。
6. 配置 ADC_GCR.CONTINUOUS 为 1，选择连续扫描模式。
7. 配置 ADC_GCR.ADC_EN 为 1，启用 ADC 模块。
8. 根据 ADC 输入通道对应的 GPIO 引脚，将待转换通道配置为模拟接口（PAD_ADS）。
9. 配置 ADC_GCR.CH_EN，启用待转换的通道。
10. 配置 ADC_GCR.ADC_START_EN 为 1，启动 ADC 转换。
11. 等待 REG_ADC_ISR 的相应通道标志位为 1，读取 ADC_DR 中的数据。
12. 如需读取多个 ADC 数据，则重复以上 1 个步骤。
13. 如使能了 ADC_IER.CH_INT_EN，则等待中断触发后读取 ADC_DR 中的数据。

20.4.4 连续扫描模式多通道 A/D 转换

连续扫描模式下，启动一次 ADC 转换后会对所选通道进行不断地连续转换，将 ADC_GCR.ADC_START_EN 写 0 可停止转换。当同时启用多个通道时，开启 ADC 转换后，依次按通道优先级 0 ~ 7 进行转换，当优先级最低的通道转换完成，则此次连续扫描转换完成。

1. 配置 ADC_GCR.ADCRST 和 ADC_GCR.ADC_PD_EN 为 0，模拟 ADC 复位释放和上电。
2. 配置 ADC_GCR.ADC_CLK_SEL 为 0，选择 ADC 时钟源为内部时钟分频器产生时钟。
3. 配置 ADC_CDR，设置 ADC 时钟分频。
4. 配置 ADC_SPW 和 ADC_COUNT，设置 ADC 转换速度。

5. 配置 ADC_TCRL.VREF_SEL, 设置 ADC 参考电压源。
6. 配置 ADC_GCR.CONTINUOUS 为 1, 选择连续扫描模式。
7. 配置 ADC_GCR.ADC_EN 为 1, 启用 ADC 模块。
8. 根据 ADC 输入通道对应的 GPIO 引脚, 将待转换通道配置为模拟接口 (PAD_ADS)。
9. 配置 ADC_GCR.CH_EN, 启用待转换的通道。
10. 配置 ADC_GCR.ADC_START_EN 为 1, 启动 ADC 转换。
11. 等待 REG_ADC_ISR 的相应通道标志位为 1, 读取 ADC_DR 中的数据。
12. 如需读取多个 ADC 数据, 则重复以上 1 个步骤。
13. 如使能了 ADC_IER.CH_INT_EN 和 ADC_IER.RXIEN 中断, 则等待中断触发后读取 ADC_DR。

注:

- 在使用多通道转换时, 可以通过判断最低优先级通道数据有效后, 再去读取每个通道的 ADC_DR 中的数据
- 在使用多通道转换时, 若采样精度不足, 建议适当修改 ADC_SPW 和 ADC_COUNT 的值

20.4.5 注意事项

- 若在单次扫描模式中使用 RX_INTF 中断, 须将 ADC_GCR 寄存器的 RXTLF 位设为 0, 否则 RXFIFO 中的数据因达不到 8 个而无法触发中断
- 在单次扫描模式中, 若某通道的数据中断 CHx_INTF 被使能而且触发, 则 ADC 控制器将结束模数转换, 停止读取后面未采样通道的数据。已采样通道的 CHx_DATA 均保留在相应的 ADCx_DR 寄存器
- 在本芯片设计中, 为了与 ADC 模块配合, ADC_GCR 寄存器的 DATA_SAMP_NEG 位只能设为 0, ADC_SPW 寄存器的值 SAMPCLK_WIDTH 应该大于或等于 3
- 每产生一次硬件触发事件, ADC 只转换一次, 使能硬件触发时建议将 ADC 模式配置为单次扫描模式

20.5 ADC 经 OPA 缓冲采样使用流程

20.5.1 ADC 经 OPA 缓冲采样图

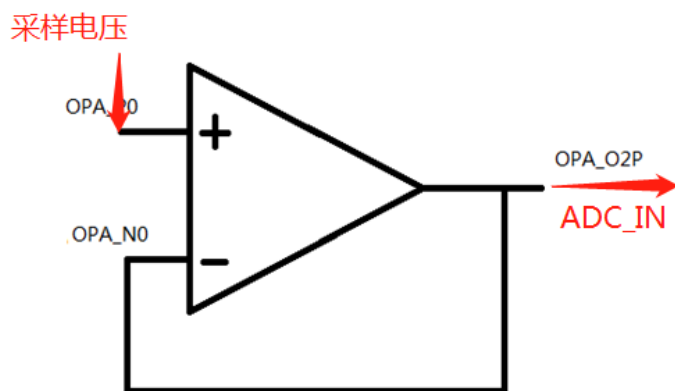


图 20-1: ADC 经 OPA 缓冲采样图

20.5.2 ADC 经 OPA 缓冲采样流程图

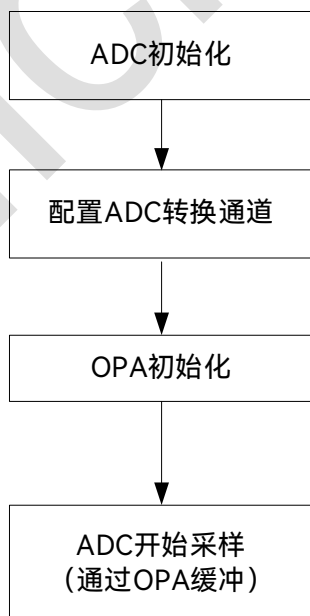


图 20-2: ADC 经 OPA 缓冲采样流程图

20.5.3 ADC 经 OPA 缓冲后采样使用流程

1. ADC 初始化。选择 ADC 参考电压源、ADCCLK 分频和采样模式。
2. 配置 ADC 转换通道。
3. OPA 初始化。配置成 OPA 正端内部接 ADC 输入，OPA 负端通过内部连接输出，使能 ADC 输入通道经 OPA 缓冲。
4. ADC 开始进行采样。

21 GPIO

21.1 概述

GPIO 包含通用数据输入输出接口，这些引脚可以与其他功能引脚共享，这取决于芯片的配置。通过这些数据接口，可以配置任意数目的引脚作为中断信号。UM2080F32 有五组 GPIO，分别是 GPIOA、GPIOB、GPIOC、GPIOD、GPIOE 分别简称为 PA、PB、PC、PD、PE。GPIO 的相关寄存器的功能需要设置对应的比特位，例如设置 PB1 方向为输出，GPIO_DIR 的 bit[1]控制位需要设置为 1，其他位的设置遵循此原则，也即是 PBx 对应寄存器 GPIO_DIR 的 bit[x]控制位。最大支持 17 个 GPIO。

21.2 主要特性

- 所有输入/输出引脚方向都可以通过软件进行配置
- 每个 GPIO_IN 引脚可配置成边沿或电平方式触发中断

21.3 寄存器描述

GPIOA 寄存器基地址: 0x40004000

GPIOB 寄存器基地址: 0x40004400

GPIOC 寄存器基地址: 0x40004800

GPIOD 寄存器基地址: 0x40004C00

GPIOE 寄存器基地址: 0x40005000

表 21-1: GPIO 寄存器列表

偏置	名称	描述
0x00	GPIO_DIR	GPIO 数据方向寄存器
0x08	GPIO_SET	GPIO 输出置位寄存器
0x0C	GPIO_CLR	GPIO 输出清零寄存器

偏置	名称	描述
0x10	GPIO_ODATA	GPIO 输出引脚映射寄存器
0x14	GPIO_IDATA	GPIO 输入引脚映射寄存器
0x18	GPIO_IEN	GPIO 中断使能寄存器
0x1C	GPIO_IS	GPIO 中断触发模式寄存器
0x20	GPIO_IBE	GPIO 中断边沿触发设置寄存器
0x24	GPIO_IEV	GPIO 中断高低电平触发设置寄存器
0x28	GPIO_IC	GPIO 中断状态清除寄存器
0x2C	GPIO_RIS	GPIO 原始中断状态寄存器
0x30	GPIO_MIS	GPIO 屏蔽后中断状态寄存器

21.3.1 数据方向寄存器 GPIO_DIR (偏移: 00H)

比特	名称	属性	复位值	描述
7:0	GPIO_DIR	R/W	0x00	8 位寄存器，GPIO 输入输出控制寄存器： 0：输入 1：输出

注：GPIOx_DIR[y] (x=A...E) 寄存器中的 8 位与该组 8 个引脚一一对应，例如 GPIOB_DIR[1]与 PB1 对应。

21.3.2 输出置位寄存器 GPIO_SET (偏移: 08H)

比特	名称	属性	复位值	描述
7:0	GPIO_SET	W	0x00	8 位寄存器，GPIO 输出置位寄存器： 0：无效操作 1：当 IO 为输出时，IO 置位

注：GPIOx_SET [y] (x=A...E) 寄存器中的 8 位与该组 8 个引脚一一对应，例如 GPIOB_SET [1]与 PB1 对应

21.3.3 输出清零寄存器 GPIO_CLR (偏移: 0CH)

比特	名称	属性	复位值	描述
7:0	GPIO_CLR	W	0x00	8 位寄存器，GPIO 输出清零寄存器： 0：无效操作 1：当 IO 为输出时，IO 清零

注：GPIOx_CLR [y] (x=A...E) 寄存器中的 8 位与该组 8 个引脚一一对应，例如 GPIOB_CLR [1]与 PB1 对应。

21.3.4 GPIO 输出引脚映射寄存器 GPIO_ODATA (偏移：10H)

比特	名称	属性	复位值	描述
7:0	GPIO_ODATA	R/W	0x00	8 位寄存器，GPIO 输出引脚映射寄存器： 当 GPIO 方向为输出有效，写操作直接写至外部引脚，读获得外部引脚值

注：GPIOx_ODATA [y] (x=A...E) 寄存器中的 8 位与该组 8 个引脚一一对应，例如 GPIOB_ODATA [1]与 PB1 对应

21.3.5 GPIO 输入引脚映射寄存器 GPIO_IDATA (偏移：14H)

比特	名称	属性	复位值	描述
7:0	GPIO_IDATA	R	0x00	8 位寄存器，GPIO 输入引脚映射寄存器： 当 GPIO 方向为输入有效，读获得外部引脚值；此寄存器为只读寄存器

注：GPIOx_IDATA [y] (x=A...E) 寄存器中的 8 位与该组 8 个引脚一一对应，例如 GPIOB_IDATA [1]与 PB1 对应

21.3.6 GPIO 中断使能寄存器 GPIO_IEN (偏移：18H)

比特	名称	属性	复位值	描述
7:0	GPIO_IEN	R/W	0x00	8 位寄存器，GPIO 中断使能寄存器： 0：禁止相应引脚中断 1：使能相应引脚中断

注：GPIOx_IEN[y] (x=A...E) 寄存器中的 8 位与该组 8 个引脚一一对应，例如 GPIOB_IEN[1]与 PB1 对应。

21.3.7 GPIO 中断触发模式寄存器 GPIO_IS (偏移: 1CH)

比特	名称	属性	复位值	描述
7:0	GPIO_IS	R/W	0x00	7 位寄存器, GPIO 中断触发模式: 0: 边沿检测 1: 电平检测

注: GPIOx_IS [y] (x=A...E) 寄存器中的 8 位与该组 8 个引脚一一对应, 例如 GPIOB_IS [1]与 PB1 对应。

21.3.8 GPIO 中断边沿触发设置寄存器 GPIO_IBE (偏移: 20H)

比特	名称	属性	复位值	描述
7:0	GPIO_IBE	R/W	0x00	8 位寄存器, GPIO 中断边沿触发设置: 0: 单边沿触发 1: 双边沿触发

注: GPIOx_IBE [y] (x=A...E) 寄存器中的 8 位与该组 8 个引脚一一对应, 例如 GPIOB_IBE[1]与 PB1 对应。

21.3.9 GPIO 中断高低电平触发设置寄存器 GPIO_IEV (偏移: 24H)

比特	名称	属性	复位值	描述
7:0	GPIO_IEV	R/W	0x00	8 位寄存器, GPIO 中断高低电平触发设置: 0: 下降沿/低电平触发 1: 上升沿/高电平触发

注: GPIOx_IEV [y] (x=A...E) 寄存器中的 8 位与该组 8 个引脚一一对应, 例如 GPIOB_IEV[1]与 PB1 对应。

21.3.10 GPIO 中断状态清除寄存器 GPIO_IC (偏移: 28H)

比特	名称	属性	复位值	描述
7:0	GPIO_IC	W	0x00	8 位寄存器, GPIO 中断清除寄存器: 0: 无效操作 1: 清除对应引脚中断

注：GPIOx_IC[y] (x=A...E) 寄存器中的 8 位与该组 8 个引脚一一对应，例如 GPIOB_IC[1]与 PB1 对应。

21.3.11 GPIO 原始中断状态寄存器 GPIO_RIS（偏移：2CH）

比特	名称	属性	复位值	描述
7:0	GPIO_RIS	R	0x00	8 位寄存器，GPIO 原始中断寄存器： 0：对应引脚无中断挂起 1：对应引脚有中断挂起

注：GPIOx_RIS[y] (x=A...E) 寄存器中的 8 位与该组 8 个引脚一一对应，例如 GPIOB_RIS[1]与 PB1 对应。

21.3.12 GPIO 屏蔽后中断状态寄存器 GPIO_MIS（偏移：30H）

比特	名称	属性	复位值	描述
31:0	GPIO_MIS	R	0x00	32 位寄存器，GPIO 屏蔽后中断状态寄存器：反映对应引脚屏蔽后的中断状态。

注：GPIOx_MIS[y] (x=A...E) 寄存器中的 8 位与该组 8 个引脚一一对应，例如 GPIOB_MIS[1]与 PB1 对应。

21.4 使用流程

21.4.1 输入输出 IO

- 配置 GPIO_DIR 寄存器，选择 GPIO 方向
- 可使用 GPIO_SET/GPIO_CLR 或 GPIO_ODATA 来设置输出电平
- 使用 GPIO_IDATA 来获取输入引脚电平

21.4.2 中断触发模式

中断初始化过程:

1. 设置 GPIO_DIR 为输入。
2. 清除 GPIO_IE 以避免异常。
3. 配置寄存器 GPIO_IS, 选择是边沿/电平触发类型。
4. 在单边沿触发方式下, 配置寄存器 GPIO_IBE, 确定是单边触发还是双边触发。
5. 在单边沿触发方式下, 配置寄存器 GPIO_IIEV, 确定是哪种边沿触发类型。
6. 在电平触发方式下, 配置寄存器 GPIO_IIEV, 确定是哪种电平触发类型。
7. 配置寄存器 GPIO_IC 来清除中断。
8. 配置寄存器 GPIO_IE 使能相应位中断。

21.4.3 清除中断

ISR 写 GPIO_IC 来清除中断状态。如果在清除寄存器的同时有新的边沿触发中断产生, 这个新的中断将会保持有效直到下一次清除。读取中断状态操作应该在清 GPIO_IE 之前进行, 清 GPIO_IE 操作将清除相应中断状态。

22 CRC16

22.1 概述

CRC16 是一个以多项式 $G(x) = x^{16}+x^{12}+x^5+1$ 为计算式的硬件 16 位 CRC 循环冗余校验计算电路。可以根据用户预设的 CRC 初值，通讯数据计算出合适的 CRC 结果，并且支持设置输入数据与结果的正反向。

22.2 寄存器描述

CRC 寄存器基地址：0x40001800

表 22-1：CRC 寄存器列表

偏置	名称	描述
0x00	CRC16_DATA	写入需校验的数据与读出 CRC 结果
0x04	CRC16_INIT	写入 16 位 CRC 初值
0x08	CRC16_CTRL	CRC 控制寄存器

22.2.1 数据寄存器 CRC16_DATA（偏移：00H）

比特	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:8	RSLT2	R	0x0	读出 16 位 CRC 计算结果的高 8 位
7:0	DATA_RSLT1	R/W	0x0	写：写入需要进行 CRC 校验计算的数据，如需要校验的数据不止 8 位需按顺序逐次写入 读：读出 16 位 CRC 计算结果的低 8 位

低 8 位对于需校验数据来说为只写，写入后无法再次读出。

读操作返回 16 位 CRC 计算结果，其中低 8 位为与数据寄存器复用。

读操作将会对 CRC 计算清零，即读操作后将会重新载入初始值，次输入数据时会进行与读之前结果无关的新一轮计算。

22.2.2 初始值寄存器 CRC16_INIT (偏移: 04H)

比特	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	INIT	R/W	0x0	写入 16 位 CRC 初始值

22.2.3 控制寄存器 CRC16_CTRL (偏移: 08H)

比特	名称	属性	复位值	描述
31:3	RSV	-	-	保留
2	RSLT_REV	R/W	0	CRC 计算结果是否进行高低位倒序 0: 不倒序 1: 倒序
1	DATA_REV	R/W	0	CRC 计算数据是否进行高低位倒序 0: 不倒序 1: 倒序
0	INITIAL_REV	R/W	0	CRC 初始值是否进行高低位倒序 0: 不倒序 1: 倒序

22.3 使用流程

1. 设置 16 位初始值 CRC16_INIT。
2. 设置 CRC16_CTRL，选择数据是否倒序。
3. 向 CRC16_DATA 中写入 8 位 CRC 计算数据，如没有完成 CRC 数据输入顺次输入之后 8 位数据直至输入完成。
4. 读 CRC16_DATA，将一次返回 CRC 计算结果。

注：读取结果后 CRC 计算模块将结束当前计算并重新载入初始值以备下次计算使用。

23 WDT

23.1 概述

看门狗定时器在到达超时的值时可以产生不可屏蔽中断或者是复位。当系统由于软件错误或是由于外部设备故障而无法按照预期的方式响应时，使用看门狗定时器可以重新获得控制权。

23.2 主要特性

- 32 位递减并且可编程装载的寄存器
- 独立的看门狗时钟使能
- 带中断屏蔽的中断生成逻辑
- 软件跑飞保护锁定寄存器
- 复位使能/禁止产生逻辑
- 调试期间，微处理器的 CPU 暂停时，用户可使能的停滞

23.3 寄存器描述

WDT 寄存器基地址：0x40002400

表 23-1: WDT 寄存器列表

偏置	名称	描述
0x00	WDT_LOAD	装载寄存器
0x04	WDT_CNT	计数寄存器
0x08	WDT_CTRL	控制寄存器
0x0C	WDT_CLR	清除寄存器
0x10	WDT_INTRAW	RAW 中断状态寄存器
0x14	WDT_MINTS	MASK 中断状态寄存器
0x18	WDT_STALL	STALL 寄存器
0x1C	WDT_LOCK	LOCK 寄存器

23.3.1 装载寄存器 WDT_LOAD (偏移: 00H)

比特	名称	属性	复位值	描述
31:0	LOAD	R/W	0xFFFFFFFF	WDOG初始装载值

23.3.2 计数寄存器 WDT_CNT (偏移: 04H)

比特	名称	属性	复位值	描述
31:0	CNT	R	0xFFFFFFFF	WDOG内部CNT计数值

23.3.3 控制寄存器 WDT_CTRL (偏移: 08H)

比特	名称	属性	复位值	描述
31	WRC	R	1	WDT加载值设置或写WDT_CTRL寄存器生效。向WDT_LOAD或者WDT_CTRL寄存器进行写操作时，设置位生效会有一定时间的延时。 0: 设置为仍未生效 1: 设置位生效
30:2	RSV	-	-	保留
1	RSTEN	R/W	0	WDT溢出复位使能: 0: 不使能溢出复位功能 1: 使能溢出复位功能
0	INTEN	R/W	0	WDT中断使能: 0: 不使能中断 1: 使能中断

23.3.4 清除寄存器 WDT_CLR (偏移: 0CH)

比特	名称	属性	复位值	描述
31:0	CLR_CARRY	W	0	向此寄存器写入任何值，将清除WDT溢出状态，从而清除掉中断和复位

23.3.5 RAW 中断状态寄存器 WDT_INTRAW (偏移: 10H)

比特	名称	属性	复位值	描述
31:0	INTRAW	R	0	原始中断寄存器，未经中断使能屏蔽： 0: WDT内部未发生溢出 1: WDT内部发生溢出

23.3.6 MASK 中断状态寄存器 WDT_MINTS (偏移: 14H)

比特	名称	属性	复位值	描述
31:0	INTMS	R	0	0: WDT未产生中断 1: WDT产生中断

23.3.7 STALL 控制寄存器 WDT_STALL (偏移: 18H)

比特	名称	属性	复位值	描述
31:16	CLKDIV	R/W	0	WDT计数时钟分频值： 0x0: 不分频 0x1: 2分频 0x2: 3分频 0xFFFE: 0xFFFF分频 0xFFFF: 保留
15:9	RSV	-	-	保留
8	STALL	R/W	0	WDT在芯片处于HALT状态时不计数功能的使能位： 0: 不使能HALT状态计数器停止工作功能 1: 使能HALT状态计数器停止工作功能
7:0	RSV	-	-	保留

23.3.8 LOCK 寄存器 WDT_LOCK (偏移: 1CH)

比特	名称	属性	复位值	描述
31:0	LOCK	W	0	WDT LOCK功能使能，当使能LOCK功能时，除此寄存器外的所有WDT寄存器均不可写。向此寄存器写任意值，使能WDT LOCK功能，向此寄存器写0x1ACCE551清除LOCK功能

23.4 使用流程

- WDT 定时器配置:
 1. 向 WDT_LOCK 寄存器写入 0x1ACCE551 解锁寄存器。
 2. 在 WDT_STALL 寄存器设置分频值。
 3. 在 WDT_CTRL 寄存器自行选择 RSTEN 复位功能和 INTEN 中断功能。
 4. 等待 WDT_CTRL 寄存器的 WRC 位被置位。
 5. 在 WDT_LOAD 寄存器里装载所需要的加载值。
 6. 等待 WDT_CTRL 寄存器的 WRC 位被置位。
 7. 向 WDT_LOCK 寄存器写入任意值锁定寄存器。
- WDT 喂狗流程配置
 1. 向 WDT_LOCK 寄存器写入 0x1ACCE551 解锁寄存器。
 2. 在 WDT_LOAD 寄存器里装载所需要的重载值。
 3. 等待 WDT_CTRL 寄存器的 WRC 位被置位。
 4. 向 WDT_LOCK 寄存器写入任意值锁定寄存器。

24 WWDT

24.1 概述

窗口看门狗是一个与 CPU 同步运行的看门狗，目的是实时监控 CPU 运行状态，在 CPU 运行异常的情况下复位 CPU，避免不可预计的后果。

24.2 主要特性

- 10 位递减并且可编程装载的寄存器
- 系统内部的故障探测器
- 时钟与系统时钟相同
- 用于监视软件错误
- 窗口前喂狗或超时不喂狗都会触发复位（喂狗有效窗口为 50 ~ 100%时间内）
- 计数器达到溢出时间的 75%时触发预警中断

24.3 寄存器描述

WWDT 寄存器基地址：0x40003C00

表 24-1: WWDT 寄存器列表

偏置	名称	描述
0x00	WWDT_CON	控制寄存器
0x04	WWDT_CFG	配置寄存器
0x08	WWDT_CNT	计数寄存器
0x0C	WWDT_IE	中断使能寄存器
0x10	WWDT_IF	中断标志寄存器

24.3.1 控制寄存器 WWDT_CON (偏移: 00H)

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	CON	W	0	当CPU向此地址写入0x5A时启动WWDT定时器, 在启动WWDT后, 当CPU向此地址写入0xAC时清零计数器

24.3.2 配置寄存器 WWDT_CFG (偏移: 04H)

比特	名称	属性	复位值	描述
31:3	RSV	-	-	保留
2:0	CFG	-	0	配置看门狗溢出时间: 000: TPCLK * 4096 * 1 001: TPCLK * 4096 * 4 010: TPCLK * 4096 * 16 011: TPCLK * 4096 * 64 100: TPCLK * 4096 * 128 101: TPCLK * 4096 * 256 110: TPCLK * 4096 * 512 111: TPCLK * 4096 * 1024

24.3.3 计数寄存器 WWDT_CNT (偏移: 08H)

比特	名称	属性	复位值	描述
31:10	RSV	-	-	保留
9:0	CNT	R	0	WWDT计数寄存器值, 软件可通过查询此寄存器了解WWDT计时进度

24.3.4 中断使能寄存器 WWDT_IE (偏移: 0CH)

比特	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	IE	R/W	0	WWDT中断使能: 0: 中断使能禁止 1: 中断使能打开

24.3.5 中断标志寄存器 WWDT_IF (偏移: 10H)

比特	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	IF	R/W	0	WWDT 75%计时中断标志，写1清零： 0: 无中断产生 1: 中断标志置位

24.4 使用流程

- WWDT 定时器配置
 1. 在 WWDT_CFG 寄存器设置溢出时间长度。
 2. 在 WWDT_IE 寄存器里打开中断使能。
 3. 在 WWDT_CON 寄存器中写入 0x5A 启动 WWDT 定时器。
 4. 等待发生中断（计数到 75%时间产生中断）。
 5. 等待发生复位（溢出后产生复位）。
- WWDT 喂狗流程配置

在计数时间 50 ~ 100%之内，在 WWDT_CON 寄存器中写入 0xAC 清零计数器。

25 RTC

25.1 概述

实时时钟 (RTC) 是一个独立的定时器/计数器, 可提供基本的闹钟中断或者长时间的计数服务。

闹钟中断通过可配置的实时时钟计数周期实现。

25.2 主要特性

- 内部或者外部 32kHz 时钟源
- 使用 BCD 时间实现可编程的完整万年历
- 周期唤醒中断功能
- 可编程的闹钟功能
- 可从 PAD 输出 XTALF 时钟信号供用户校准
- 数字调校, 精度 $\pm 0.119\text{ppm}$
- RTC 计时器部分不复位
- 2 路输入上下沿时间戳功能

25.3 低功耗时基分频器 (LTBC)

25.3.1 LTBC 功能

低功耗时基计数器(LTBC)模块用于产生系统所需的低速工作时钟, 功能包括:

- 通过对 RCL 的预分频得到 64Hz 的 RTC 工作时钟
- 可通过调整计数周期实现 RTC 时钟的数字调校, 每 128s 调校一次可实现最小步长为 0.23842ppm 调校后理论精度 $\pm 0.1192\text{ppm}$
- 16.384MHz 时钟虚拟调校可得到精确秒时标

- 可产生 1kHz、256Hz、64Hz、16Hz、4Hz、1Hz 周期中断，其中 1k 和 256Hz 是未经调校的，其他是经过数字调校的（如果使能了数字调校）
- 64Hz 预分频电路不受芯片复位影响
- 1/256s 精度授时

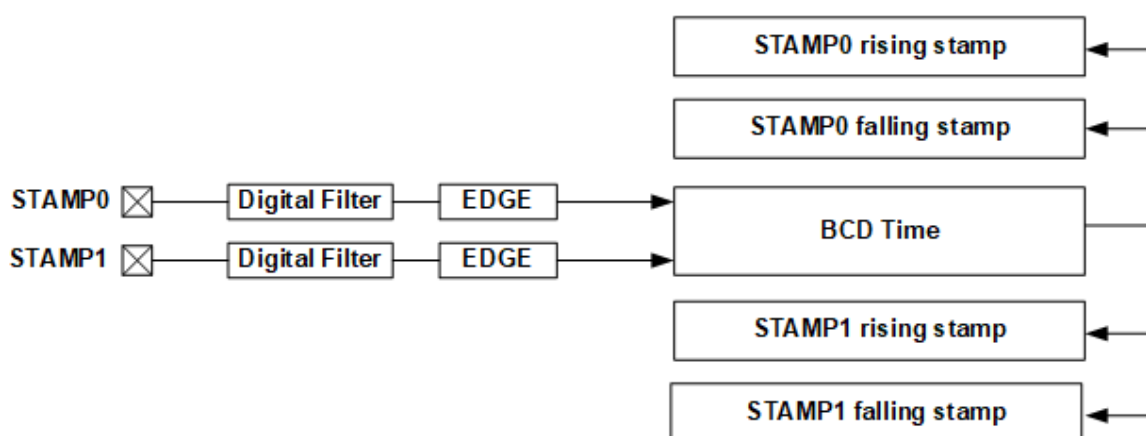
25.3.2 LTBC 数字调校

数字调校用来弥补外部时钟源带来的误差。使用 ADJUST、ADSIGN、PR1SEN 寄存器对 RTC 进行正向或者负向的数字调校，具体使用方法见寄存器的描述。

25.4 时间戳功能

为了支持开盖合盖检测，RTC 支持外部 IO 事件触发的时间戳功能。外部 IO 触发源为 STAMP0 和 STAMP1 的输入电平变化。使用此功能时，将相应的 IO 复用为 STAMP0 和 STAMP1 功能，打开 RTCSTAMPEN 寄存器使能相应通道，当 STAMP0 或 STAMP1 上出现任何上升沿或下降沿时，RTC 会自动记录当前时间到 STAMP 寄存器组中，同时产生相应的标志，可用于产生中断或者供软件查询。

注意时间戳功能仅在 Sleep 和 Deepsleep 休眠模式下有效，ACTIVE 模式下时间戳功能不起作用，开盖合盖检测由软件中断来处理。



时间戳仅在相应标志寄存器为 0 的情况下记录事件发生时间，如果对应标志已经为 1，则忽略相应事件。因此如果有多次事件发生，时间戳仅记录第一次事件发生的时间，除非软件在事件发生后清除了标志寄存器。

25.5 寄存器描述

RTC 寄存器基地址：0x40001400

表 25-1：RTC 寄存器列表

偏置	名称	描述
0x00	RTCWE	RTC 写使能寄存器
0x04	RTCIE	RTC 中断使能寄存器
0x08	RTCIF	RTC 中断标志寄存器
0x0C	BCDSEC	BCD 秒寄存器
0x10	BCDMIN	BCD 分寄存器
0x14	BCD HOUR	BCD 时寄存器
0x18	BCDDATE	BCD 天寄存器
0x1C	BCDWEEK	BCD 星期几寄存器
0x20	BCDMONTH	BCD 月寄存器
0x24	BCDYEAR	BCD 年寄存器
0x28	ALARM	闹铃设置寄存器
0x2C	FSEL	FOUT 输出选择寄存器
0x30	ADJUST	LTBC 计时调校寄存器
0x34	ADSIGN	LTBC 调校符号寄存器
0x38	PR1SEN	精确秒时标虚拟调校使能
0x3C	SECCNT	毫秒计数值寄存器
0x40	STAMPEN	RTC 时间戳使能寄存器
0x44	CLKSTAMP0R	RTC 上升沿时间戳 0
0x48	CALSTAMP0R	RTC 上升沿日历戳 0
0x4C	CLKSTAMP0F	RTC 下降沿时间戳 0
0x50	CALSTAMP0F	RTC 下降沿日历戳 0
0x54	CLKSTAMP1R	RTC 上升沿时间戳 1
0x58	CALSTAMP1R	RTC 上升沿日历戳 1
0x5C	CLKSTAMP1F	RTC 下降沿时间戳 1
0x60	CALSTAMP1F	RTC 下降沿日历戳 1

本小结对 RTC 寄存器进行了详细介绍。

25.5.1 写使能寄存器（RTCWE）（偏移：00H）

比特	名称	属性	复位值	描述
31:0	RTCWE	R/W	0	RTC 写使能寄存器： 当 CPU 向 RTCWE 写入 0xACACACAC 时，允许 CPU 向 RTC 的 BCD 时间寄存器写入初值，这时 RTCWE 置 1；当 CPU 向 RTCWE 写入不为 0xACACACAC 的任意值时恢复写保护,这时 RTCWE 清 0

25.5.2 中断使能寄存器（RTCIE）（偏移：04H）

比特	名称	属性	复位值	描述
31:17	RSV	R	0	保留
16	STPR1IE	R/W	0	RTC STAMP1 上升沿事件中断使能： 0：禁止中断 1：使能中断
15	STPF1IE	R/W	0	RTC STAMP1 下降沿事件中断使能： 0：禁止中断 1：使能中断
14	STPR0IE	R/W	0	RTC STAMP0 上升沿事件中断使能： 0：禁止中断 1：使能中断
13	STPF0IE	R/W	0	RTC STAMP0 下降沿事件中断使能： 0：禁止中断 1：使能中断
12	ADJ128_IE	R/W	0	128 秒中断使能： 0：中断使能禁止 1：中断使能打开
11	ALARM_IE	R/W	0	闹钟中断使能： 0：中断使能禁止 1：中断使能打开
10	1KHZ_IE	R/W	0	1kHz 中断使能： 0：中断使能禁止 1：中断使能打开

比特	名称	属性	复位值	描述
9	256HZ_IE	R/W	0	256hz 中断使能: 0: 中断使能禁止 1: 中断使能打开
8	64HZ_IE	R/W	0	64Hz 中断使能: 0: 中断使能禁止 1: 中断使能打开
7	16HZ_IE	R/W	0	16Hz 中断使能: 0: 中断使能禁止 1: 中断使能打开
6	8HZ_IE	R/W	0	8Hz 中断使能: 0: 中断使能禁止 1: 中断使能打开
5	4HZ_IE	R/W	0	4Hz 中断使能: 0: 中断使能禁止 1: 中断使能打开
4	2HZ_IE	R/W	0	2Hz 中断使能: 0: 中断使能禁止 1: 中断使能打开
3	SEC_IE	R/W	0	秒中断使能: 0: 中断使能禁止 1: 中断使能打开
2	MIN_IE	R/W	0	分中断使能: 0: 中断使能禁止 1: 中断使能打开
1	HOUR_IE	R/W	0	小时中断使能: 0: 中断使能禁止 1: 中断使能打开
0	DATE_IE	R/W	0	天中断使能: 0: 中断使能禁止 1: 中断使能打开

25.5.3 中断标志寄存器（RTCIF）（偏移：08H）

比特	名称	属性	复位值	描述
31:17	RSV	R	0	保留

比特	名称	属性	复位值	描述
16	STPR1IF	R/W	0	RTC STAMP1 上升沿事件中断标志： 0：无中断产生 1：中断置位 此寄存器为 1 的情况下时间戳 1 不再记录新的上升沿事件
15	STPF1IF	R/W	0	RTC STAMP1 下降沿事件中断标志： 0：无中断产生 1：中断置位 此寄存器为 1 的情况下时间戳 1 不再记录新的下降沿事件
14	STPR0IF	R/W	0	RTC STAMP0 上升沿事件中断标志： 0：无中断产生 1：中断置位 此寄存器为 1 的情况下时间戳 0 不再记录新的上升沿事件
13	STPF0IF	R/W	0	RTC STAMP0 下降沿事件中断标志： 0：无中断产生 1：中断置位 此寄存器为 1 的情况下时间戳 0 不再记录新的下降沿事件
12	ADJ128_IF	R/W	0	128 秒中断标志。写 1 清零： 0：无中断产生 1：中断置位
11	ALARM_IF	R/W	0	闹钟中断标志。写 1 清零： 0：无中断产生 1：中断置位
10	1KHZ_IF	R/W	0	1kHz 中断标志。写 1 清零： 0：无中断产生 1：中断置位
9	256HZ_IF	R/W	0	256Hz 中断标志。写 1 清零： 0：无中断产生 1：中断置位
8	64HZ_IF	R/W	0	64Hz 中断标志。写 1 清零： 0：无中断产生 1：中断置位

比特	名称	属性	复位值	描述
7	16HZ_IF	R/W	0	16Hz 中断标志。写 1 清零： 0：无中断产生 1：中断置位
6	8HZ_IF	R/W	0	8Hz 中断标志。写 1 清零： 0：无中断产生 1：中断置位
5	4HZ_IF	R/W	0	4Hz 中断标志。写 1 清零： 0：无中断产生 1：中断置位
4	2HZ_IF	R/W	0	2Hz 中断标志。写 1 清零： 0：无中断产生 1：中断置位
3	SEC_IF	R/W	0	秒中断标志。写 1 清零： 0：无中断产生 1：中断置位
2	MIN_IF	R/W	0	分中断标志。写 1 清零： 0：无中断产生 1：中断置位
1	HOUR_IF	R/W	0	小时中断标志。写 1 清零： 0：无中断产生 1：中断置位
0	DATE_IF	R/W	0	天中断标志。写 1 清零： 0：无中断产生 1：中断置位

25.5.4 BCD 时间秒寄存器（RTC_BCDSEC）（偏移：0CH）

比特	名称	属性	复位值	描述
31:7	RSV	R	0	保留
6:0	BCDSEC	R/W	不会被复位	秒时间数值，BCD 格式

25.5.5 BCD 时间分钟寄存器（RTC_BCDMIN）（偏移：10H）

比特	名称	属性	复位值	描述
31:7	RSV	R	0	保留
6:0	BCDMIN	R/W	不会被复位	分钟时间数值，BCD 格式

25.5.6 BCD 时间小时寄存器 (RTC_BCDHOUR) (偏移: 14H)

比特	名称	属性	复位值	描述
31:6	RSV	R	0	保留
5:0	BCDHOUR	R/W	不会被复位	小时数值, BCD 格式

25.5.7 BCD 时间天寄存器 (RTC_BCDDATE) (偏移: 18H)

比特	名称	属性	复位值	描述
31:6	RSV	R	0	保留
5:0	BCDDATE	R/W	不会被复位	天数数值, BCD 格式

25.5.8 BCD 时间星期寄存器 (RTC_BCDWEEK) (偏移: 1CH)

比特	名称	属性	复位值	描述
31:3	RSV	R	0	保留
2:0	BCDWEEK	R/W	不会被复位	周数值, BCD 格式

25.5.9 BCD 时间月寄存器 (RTC_BCDMONTH) (偏移: 20H)

比特	名称	属性	复位值	描述
31:5	RSV	R	0	保留
4:0	BCDMONTH	R/W	不会被复位	月数值, BCD 格式

25.5.10 BCD 时间年寄存器 (RTC_BCDYEAR) (偏移: 24H)

比特	名称	属性	复位值	描述
31:8	RSV	R	0	保留
7:0	BCDYEAR	R/W	不会被复位	年数值, BCD 格式

25.5.11 闹钟寄存器 (RTC_ALARM) (偏移: 28H)

比特	名称	属性	复位值	描述
31:22	RSV	R	0	保留
21:16	ALARMHOUR	R/W	0	闹钟的小时数值

比特	名称	属性	复位值	描述
15	RSV	R	0	保留
14:8	ALARMMIN	R/W	-	闹钟的分数值
7	RSV	R	-	保留
6:0	ALARMSEC	R/W	-	闹钟的秒数值

25.5.12 时钟信号输出控制寄存器（RTC_FSEL）（偏移：2CH）

比特	名称	属性	复位值	描述
31:4	RSV	R	0	保留
3:0	FSEL	R/W	4'b0000	频率输出选择信号： 4'b0000：输出 16.384M 时钟分频得到的精确 1 秒方波 4'b0001：输出 16.384M 时钟分频的高电平宽度 80ms 的秒时标 4'b0010：输出秒计数器进位信号，高电平宽度 1s 4'b0011：输出分计数器进位信号，高电平宽度 1s 4'b0100：输出小时计数器进位信号，高电平宽度 1s 4'b0101：输出天计数器进位信号，高电平宽度 1s 4'b0110：输出闹钟匹配信号 4'b0111：输出 128 秒方波信号 4'b1000：反向输出 16.384M 时钟分频的高电平宽度 80ms 的秒时标 4'b1001：反向输出秒计数器进位信号 4'b1010：反向输出分计数器进位信号 4'b1011：反向输出小时计数器进位信号 4'b1100：反向输出天计数器进位信号 4'b1101：反向输出闹钟匹配信号 4'b1110：反向输出 16.384M 时钟分频的精确 1s 方波信号 4'b1111：输出 RTC 内部秒时标方波

25.5.13 LTBC 数值调整寄存器 (RTC_ADJUST) (偏移: 30H)

比特	名称	属性	复位值	描述
31:11	RSV	R	0	保留
10:0	ADJUST	R/W	不会被复位	LTBC 补偿调整数值 --在进行数字调校时, ADJUST[10:7]为数字调校的公共值, 在每秒的计数里调整 ADJUST[10:7]个 32768Hz 时钟周期; ADJUST[6:0]为数字调校的私有值, 在 128s 计数里的第 0 秒到第 ADJUST[6:0]-1 秒各调整 1 个 32768Hz 时钟周期。 --在进行虚拟调校时, RTC 在 128s 计数里调整 ADJUST[10:0]个 30.5μs。

25.5.14 LTBC 数值调整方向寄存器 (RTC_ADSIGN) (偏移: 34H)

比特	名称	属性	复位值	描述
31:1	RSV	R	0	保留
0	ADSIGN	R/W	不会被复位	LTBC 补偿方向: 0: 增加计数初值 1: 减少计数初值

25.5.15 LTBC 虚拟调校使能寄存器 (RTC_PR1SEN) (偏移: 38H)

比特	名称	属性	复位值	描述
31:1	RSV	R	0	保留
0	PR1SEN	R/W	0	虚拟调校使能信号: 0: 使能虚拟调校功能, 使用 16.384MHz 时钟分频对 RTC 进行调校 1: 禁止虚拟调校功能, 使用 32768Hz 时钟分频对 RTC 进行调校

25.5.16 毫秒计数寄存器（RTC_MSCNT）（偏移：3CH）

比特	名称	属性	复位值	描述
31:8	RSV	R	0	保留
7:0	MSCNT	R	不会被复位	毫秒计数器值。以 256Hz 为周期计数，精度 3.9ms

25.5.17 RTC 时间戳使能寄存器（RTC_STORREN）（偏移：40H）

比特	名称	属性	复位值	描述
31:2	RSV	R	0	保留
1	STAMP1EN	R/W	不会被复位	STAMP1 触发的时间戳功能使能位。无复位值，建议软件上电后进行初始化 0：关闭时间戳 1：打开时间戳
0	STAMP0EN	R/W	不会被复位	STAMP0 触发的时间戳功能使能位。无复位值，建议软件上电后进行初始化 0：关闭时间戳 1：打开时间戳

25.5.18 CLKSTAMP0R 寄存器（RTC_CLKS0R）（偏移：44H）

比特	名称	属性	复位值	描述
31:22	RSV	R	0	保留
21:16	HRSTP0R	R/W	不会被复位	检测到 STAMP0 输入上升沿后存储 BCD 小时寄存器的值
15	RSV	R	-	-
14:8	MINSTP0R	R/W	不会被复位	检测到 STAMP0 输入上升沿后存储 BCD 分寄存器的值
7	RSV	R	0	-
6:0	SECSTP0R	R/W	不会被复位	检测到 STAMP0 输入上升沿后存储 BCD 秒寄存器的值

25.5.19 CALSTAMP0R 寄存器 (RTC_CALS0R) (偏移: 48H)

比特	名称	属性	复位值	描述
31:24	YRSTP0R	R/W	不会被复位	检测到 STAMP0 输入上升沿后存储 BCD 年寄存器的值
23:21	RSV	R	0	保留
20:16	MONSTP0R	R/W	不会被复位	检测到 STAMP0 输入上升沿后存储 BCD 月寄存器的值
15:11	RSV	R	0	保留
10:8	WKSTP0R	R/W	不会被复位	检测到 STAMP0 输入上升沿后存储 BCD 周寄存器的值
7:6	RSV	R	0	保留
5:0	DAYSTP0R	R/W	不会被复位	检测到 STAMP0 输入上升沿后存储 BCD 天寄存器的值

25.5.20 CLKSTAMP0F 寄存器 (RTC_CLKS0F) (偏移: 4CH)

比特	名称	属性	复位值	描述
31:22	RSV	R	0	保留
21:16	HRSTP0F	R/W	不会被复位	检测到 STAMP0 输入下降沿后存储 BCD 小时寄存器的值
15	RSV	R	0	保留
14:8	MINSTP0F	R/W	不会被复位	检测到 STAMP0 输入下降沿后存储 BCD 分寄存器的值
7	RSV	R	0	保留
6:0	SECSTP0F	R/W	不会被复位	检测到 STAMP0 输入下降沿后存储 BCD 秒寄存器的值

25.5.21 CALSTAMP0F 寄存器 (RTC_CALS0F) (偏移: 50H)

比特	名称	属性	复位值	描述
31:24	YRSTP0F	R/W	不会被复位	检测到 STAMP0 输入下降沿后存储 BCD 年寄存器的值
23:21	RSV	R	0	保留
20:16	MONSTP0F	R/W	不会被复位	检测到 STAMP0 输入下降沿后存储 BCD 月寄存器的值

比特	名称	属性	复位值	描述
15:11	RSV	R	0	保留
10:8	WKSTP0F	R/W	不会被复位	检测到 STAMP0 输入下降沿后存储 BCD 周寄存器的值
7:6	RSV	R	0	保留
5:0	DAYSTP0F	R/W	不会被复位	检测到 STAMP0 输入下降沿后存储 BCD 天寄存器的值

25.5.22 CLKSTAMP1R 寄存器 (RTC_CLKS1R) (偏移: 54H)

比特	名称	属性	复位值	描述
31:22	RSV	R	0	保留
21:16	HRSTP1R	R/W	不会被复位	检测到 STAMP1 输入上升沿后存储 BCD 小时寄存器的值
15	RSV	R	0	保留
14:8	MINSTP1R	R/W	不会被复位	检测到 STAMP1 输入上升沿后存储 BCD 分寄存器的值
7	RSV	R	0	保留
6:0	SECSTP1R	R/W	不会被复位	检测到 STAMP1 输入上升沿后存储 BCD 秒寄存器的值

25.5.23 CALSTAMP1R 寄存器 (RTC_CALS1R) (偏移: 58H)

比特	名称	属性	复位值	描述
31:24	YRSTP1R	R/W	不会被复位	检测到 STAMP1 输入上升沿后存储 BCD 年寄存器的值
23:21	RSV	R	0	保留
20:16	MONSTP1R	R/W	不会被复位	检测到 STAMP1 输入上升沿后存储 BCD 月寄存器的值
15:11	RSV	R	0	保留
10:8	WKSTP1R	R/W	不会被复位	检测到 STAMP1 输入上升沿后存储 BCD 周寄存器的值
7:6	RSV	R	0	保留
5:0	DAYSTP1R	R/W	不会被复位	检测到 STAMP1 输入上升沿后存储 BCD 天寄存器的值

25.5.24 CLKSTAMP1F 寄存器 (RTC_CLKS1F) (偏移: 5CH)

比特	名称	属性	复位值	描述
31:22	RSV	R	0	保留
21:16	HRSTP1F	R/W	不会被复位	检测到 STAMP1 输入下降沿后存储 BCD 小时寄存器的值
15	RSV	R	0	保留
14:8	MINSTP1F	R/W	不会被复位	检测到 STAMP1 输入下降沿后存储 BCD 分寄存器的值
7	RSV	R	0	保留
6:0	SECSTP1F	R/W	不会被复位	检测到 STAMP1 输入下降沿后存储 BCD 秒寄存器的值

25.5.25 CALSTAMP1F 寄存器 (RTC_CALS1F) (偏移: 60H)

比特	名称	属性	复位值	描述
31:24	YRSTP1F	R/W	不会被复位	检测到 STAMP1 输入下降沿后存储 BCD 年寄存器的值
23:21	RSV	R	0	保留
20:16	MONSTP1F	R/W	不会被复位	检测到 STAMP1 输入下降沿后存储 BCD 月寄存器的值
15:11	RSV	R	0	保留
10:8	WKSTP1F	R/W	不会被复位	检测到 STAMP1 输入下降沿后存储 BCD 周寄存器的值
7:6	RSV	R	0	保留
5:0	DAYSTP1F	R/W	不会被复位	检测到 STAMP1 输入下降沿后存储 BCD 天寄存器的值

25.6 使用流程

25.6.1 RTC 时间设置

由于 RTC 走时时钟较慢，为了提高抗 EMC 干扰能力，提供时间写保护功能，必须先对写保护寄存器写入 0xACACACAC，才能改写时间寄存器，软件可以通过写入除 0xACACACAC 外的任意值

来禁止时间寄存器的写入，恢复写保护。

软件应在秒中断发生后再去置时，降低异步风险。同时支持 ms 级授时，即可以设置时间到 3.9ms 级别精度（1/256s）。此外，当软件写入秒时间时，硬件自动清零 64Hz->1Hz 的秒内计数器，以便实现秒对齐。

推荐的 RTC 时间设置流程如下：

1. 等待秒中断时间发生。
2. 连续写入年月日时分秒寄存器。
3. 如需 ms 级授时，再写入毫秒计数器。
4. 读出时间寄存器进行校验。

经过以上操作后最大授时误差在 4ms 以内。

25.6.2 RTC 时间读取

- 时间读取方式 1

1. 读 BCDTIME 值
2. 再次读 BCDTIME 值

如果两次读取内容一致，则为正确的当前时间；如果两次读取内容不一致，则重复前两个步骤。

- 时间读取方式 2

软件在 1s 中断发生后去读取时间，在 1s 内当前时间不会变化，因此能保证读到正确的当前时间值。

25.6.3 时间戳使用

1. 配置 SCU_PxSEL 寄存器，触发使用的 GPIO 为 RTC_STAMP 模式。
2. 配置 SCU_PADIE0 寄存器，触发使用的 GPIO 为输入使能模式。
3. 配置 GPIO_DIR(x)寄存器，触发使用的 GPIO 为输入模式。
4. 配置 RTCIE 寄存器，选择 GPIO 边沿触发中断的模式。

5. 配置 STAMPEN 寄存器，使能时间戳功能。
6. 使能 RTC 相关的中断。
7. 当 GPIO 检测到匹配信号时，触发中断，并记录时间戳。

25.6.4 RTC 设置闹钟

1. 配置 SCU_CTRL0 寄存器，32kHz RCL 时钟，RTC 模块内部自动分频成 1Hz 时钟。
2. 配置 RTCWE 寄存器，使能写功能。
3. 设定 BCD 码的时分秒初始值。
4. 配置 RTCWE 寄存器，关闭写功能。
5. 配置 ALARM 寄存器，设定 RTC 闹钟匹配值。
6. 写 RTCIF 寄存器清除 RTC 中断。
7. 配置 RTCIE 寄存器，使能 RTC 中断。
8. RTC 中断触发后，配置 RTCIF 寄存器清除中断。

26 DMA

26.1 概述

直接存储器访问(DMA)，支持 4 通道数据传输。

26.2 主要特性

- 支持单 MASTER 口
- 可以控制 FLASH、SRAM、SPI0、SPI1、UART1、ADC 模块之间的数据传输，其中 FLASH 仅可以作为源地址
- 支持 Memory to Memory 模式、Memory to Peripheral 模式、Peripheral to Memory 模式、Peripheral to Peripheral 模式
- 内部含有 4 个 DMA 通道
- 数据传输的位宽可设、传输的 Block 长度可设
- 内部含有深度为 16 的 FIFO
- Block 最大长度可设为 32767
- 支持源地址不变传输、递增传输，支持目的地址的不变传输、递增传输

26.3 寄存器描述

DMA 寄存器基地址：0x40020000

表 26-1: DMA 寄存器列表

偏置	名称	描述
0x00	DMA_SRC_ADDR_C0	通道 0 源传送地址寄存器
0x04	DMA_DST_ADDR_C0	通道 0 目的传送地址寄存器
0x08	DMA_CH_CTRL_C0	通道 0 控制信息寄存器
0x0C	DMA_CH_STS_C0	通道 0 传送状态寄存器

偏置	名称	描述
0x10	DMA_SRC_ADDR_C1	通道 1 源传送地址寄存器
0x14	DMA_DST_ADDR_C1	通道 1 目的传送地址寄存器
0x18	DMA_CH_CTRL_C1	通道 1 控制信息寄存器
0x1c	DMA_CH_STS_C1	通道 1 传送状态寄存器
0x20	DMA_SRC_ADDR_C2	通道 2 源传送地址寄存器
0x24	DMA_DST_ADDR_C2	通道 2 目的传送地址寄存器
0x28	DMA_CH_CTRL_C2	通道 2 控制信息寄存器
0x2c	DMA_CH_STS_C2	通道 2 传送状态寄存器
0x30	DMA_SRC_ADDR_C3	通道 3 源传送地址寄存器
0x34	DMA_DST_ADDR_C3	通道 3 目的传送地址寄存器
0x38	DMA_CH_CTRL_C3	通道 3 控制信息寄存器
0x3C	DMA_CH_STS_C3	通道 3 传送状态寄存器
0x40	DMAC_EN	DMA 控制器使能寄存器
0x44	DMA_SOFT_RESET	DMA 软复位寄存器
0x48	DMA_INT_STATUS	DMA 中断指示寄存器
0x4c	DMA_INT_MASK	DMA 中断屏蔽寄存器
0x54	DMA_PER_REQ	DMA 外设请求寄存器

26.3.1 通道源传送地址寄存器 DMA_SRC_ADDR_Cx（偏移：10xH）

(x=0,1,2,3)

比特	名称	属性	复位值	描述
31:21	HI_SRC_ADDR	R/W	0	源高位地址，主要用于 decoder 选通
20:0	LOW_SRC_ADDR	R/W	0	源低位地址，主要用于具体外设的存储访问

26.3.2 通道目的传送地址寄存器 DMA_DST_ADDR_Cx（偏移：10x+04H）（x=0,1,2,3）

比特	名称	属性	复位值	描述
31:21	HI_DST_ADDR	R/W	0	目的高位地址，主要用于 decoder 选通
20:0	LOW_DST_ADDR	R/W	0	目的低位地址，主要用于具体外设的存储访问

26.3.3 通道控制信息寄存器 DMA_CH_CTRL_Cx (偏移: 10x+08H)

(x=0,1,2,3)

比特	名称	属性	复位值	描述																				
31:30	WIDTH	R/W	0	数据位宽： 0：8 位数据位宽 1：16 位数据位宽 2：32 位数据位宽 3：非法，但模块表现为 32 位的读写 源和目的数据位宽一样																				
29:15	XFER_SIZE	R/W	0	传输块大小，对 8 位模式支持 32767 byte 的块，对 32 位模式支持 32767words 的块																				
14:12	FLOW_CTRL	R/W	0	流控模式 <table><tr><th>数据值</th><th>源</th><th>目的</th><th>流控</th></tr><tr><td>0</td><td>Memory</td><td>Memory</td><td>DMA</td></tr><tr><td>1</td><td>Memory</td><td>Peripheral</td><td>DMA</td></tr><tr><td>2</td><td>Peripheral</td><td>Memory</td><td>DMA</td></tr><tr><td>3</td><td>Peripheral</td><td>Peripheral</td><td>DMA</td></tr></table>	数据值	源	目的	流控	0	Memory	Memory	DMA	1	Memory	Peripheral	DMA	2	Peripheral	Memory	DMA	3	Peripheral	Peripheral	DMA
数据值	源	目的	流控																					
0	Memory	Memory	DMA																					
1	Memory	Peripheral	DMA																					
2	Peripheral	Memory	DMA																					
3	Peripheral	Peripheral	DMA																					
11	RSV	-	-	保留																				
10:8	DST_PER	R/W	0	目的外设，主要用于目的外设的请求选取 具体外设分配为： 0：SPI0 发送 2：SPI1 发送 4：UART1 发送 7：GPIO0_Gtimer																				
7:5	SRC_PER	R/W	0	源外设，主要用于源外设的请求选取 具体外设分配为： 1：SPI0 接收 3：SPI1 接收 5：UART1 接收 6：ADC 接收 7：GPIO0_Gtimer																				
4:3	DST_INC	R/W	0	目的地址递增指示位，如果有效，则目的地址 将随读取的数据递增，否则保持不变 01：地址递增 10：地址递减																				

比特	名称	属性	复位值	描述
2:1	SRC_INC	R/W	0	源地址递增指示位，如果有效，则源地址将随读取的数据递增，否则保持不变 01：地址递增 10：地址递减
0	CH_EN	R/W	0	通道使能标志，对于 DMAC 流控时，块传送结束后自动清 0

26.3.4 通道传送状态寄存器 DMA_CH_STS_Cx (偏移：10x+0CH)

(x=0,1,2,3)

比特	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:1	LENGTH	R	0	在 DMA 传输时，表示此通道已经传输的数据长度
0	CH_BUSY	R	0	通道工作状态信息： 0: Idle 1: Busy

26.3.5 DMA 控制器使能寄存器 DMAC_EN (偏移：40H)

比特	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	DMAC_EN	R/W	0	0: 关闭 DMA 控制寄存器 1: 使能 DMA 控制器；

26.3.6 DMA 软复位寄存器 DMA_SOFT_RESET (偏移：44H)

比特	名称	属性	复位值	描述
31:0	DMA_SOFT_RESET	W	-	本寄存器为写操作虚拟寄存器，当 DMAC 模块采样到对此寄存器有写操作时，DMAC 将复位状态机以及需要复位的寄存器。没有实际的比特存在

26.3.7 DMA 中断指示寄存器 DMA_INT_STATUS (偏移: 48H)

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7	INT_TC_C3	R/W	0	通道 3 块传输结束中断指示, 写 1 清 0
6	INT_TC_C2	R/W	0	通道 2 块传输结束中断指示, 写 1 清 0
5	INT_ERR_C3	R/W	0	通道 3 总线出错中断指示, 写 1 清 0
4	INT_ERR_C2	R/W	0	通道 2 总线出错中断指示, 写 1 清 0
3	INT_TC_C1	R/W	0	通道 1 块传输结束中断指示, 写 1 清 0
2	INT_TC_C0	R/W	0	通道 0 块传输结束中断指示, 写 1 清 0
1	INT_ERR_C1	R/W	0	通道 1 总线出错中断指示, 写 1 清 0
0	INT_ERR_C0	R/W	0	通道 0 总线出错中断指示, 写 1 清 0

26.3.8 DMA 中断屏蔽寄存器 DMA_INT_MASK (偏移: 4CH)

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7	MASK_TC_C3	R/W	0	通道 3 块传输结束中断屏蔽, 如果为低, 将不输出 IntTc 中断, 即 IntTc=0
6	MASK_TC_C2	R/W	0	通道 2 块传输结束中断屏蔽, 如果为低, 将不输出 IntTc 中断, 即 IntTc=0
5	MASK_ERR_C3	R/W	0	通道 3 总线出错中断屏蔽; 如果为低, 将不输出 IntErr 中断, 即 IntErr=0
4	MASK_ERR_C2	R/W	0	通道 2 总线出错中断屏蔽; 如果为低, 将不输出 IntErr 中断, 即 IntErr=0
3	MASK_TC_C1	R/W	0	通道 1 块传输结束中断屏蔽, 如果为低, 将不输出 IntTc 中断, 即 IntTc=0
2	MASK_TC_C0	R/W	0	通道 0 块传输结束中断屏蔽, 如果为低, 将不输出 IntTc 中断, 即 IntTc=0
1	MASK_ERR_C1	R/W	0	通道 1 总线出错中断屏蔽; 如果为低, 将不输出 IntErr 中断, 即 IntErr=0
0	MASK_ERR_C0	R/W	0	通道 0 总线出错中断屏蔽; 如果为低, 将不输出 IntErr 中断, 即 IntErr=0

26.3.9 DMA 外设请求寄存器 DMA_PER_REQ DMA（偏移：54H）

直接把 8 个外设请求连过来，没有实际的寄存器。

比特	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7	GPIO0_GTIM_REQ	R	0	GPIO0_Gtimer 请求
6	ADC_REQ	R	0	ADC 接收请求
5	UART1_RX_REQ	R	0	UART1 接收请求
4	UART1_TX_REQ	R	0	UART1 发送请求
3	SPI1_RX_REQ	R	0	SPI1 接收请求
2	SPI1_TX_REQ	R	0	SPI1 发送请求
1	SPI0_RX_REQ	R	0	SPI0 接收请求
0	SPI0_TX_REQ	R	0	SPI0 发送请求

26.4 使用流程

软件配置步骤：

1. 配置 DMA_CH_CTRL_Cx.FLOW_CTRL，选择 DMA 传输模式。
2. 配置 DMA_CH_CTRL_Cx.SRC_PER 和 DST_PER，选择外设握手信号(传输地址为外设时才需要设置)。
3. 配置 DMA_CH_CTRL_Cx.SRC_INC 和 DST_INC，选择源地址和目的地址是否递增或不变。
4. 配置 DMA_CH_CTRL_Cx.WIDTH，选择传输数据的位宽。
5. 配置 DMAC_EN 为 1，使能 DMA 控制器。
6. 配置 DMA_SRC_ADDR_Cx，配置通道源地址。
7. 配置 DMA_DST_ADDR_Cx，配置通道目的地址。
8. 配置 DMA_CH_CTRL_Cx.XFER_SIZE，配置传输块数量。
9. 配置 DMA_CH_CTRL_Cx.CH_EN，使能 DMA 通道传输。
10. 等待 DMA_CH_CTRL_Cx.CH_EN 为 0，传输完成。

若使能了传输结束中断，则等待传输结束中断后再处理。

27 SysTick

27.1 概述

OS 要想支持多任务，就需要周期执行上下文切换，这样就需要有定时器之类的硬件资源打断程序执行。当定时器中断产生时，处理器就会在异常处理中进行 OS 任务调度，同时还会进行 OS 维护的工作。Cortex-M0+处理器中有一个称为 SysTick 的简单定时器，用于产生周期性的中断请求。

SysTick 为 24 位的定时器，并且向下计数。定时器的计数减到 0 后，就会重新装载一个可编程的数值，并且同时产生 SysTick 异常(异常编号为 15)，该异常事件会引起 SysTick 异常处理的执行，这个过程是 OS 的一部分。

对于不需要 OS 的系统，SysTick 定时器也可以用作其他用途，比如定时、计时或者为需要周期执行的任务提供中断源。SysTick 异常的产生是可控的，如果异常被禁止，仍然可以用轮询的方法使用 SysTick 定时器，比如检查当前的计数值或者轮询溢出标志。

27.2 寄存器描述

SysTick 寄存器基地址：0xE000E010

表 27-1: SysTick 寄存器列表

偏置	名称	描述
0x00	SYS_CSR (SysTick_CTRL)	SysTick 控制和状态寄存器
0x04	SYS_RVR (SysTick_LOAD)	SysTick 重载值寄存器
0x08	SYS_CVR (SysTick_VAL)	SysTick 当前值寄存器

27.2.1 控制和状态寄存器 SYS_CSR (偏移: 00H)

比特	名称	属性	复位值	描述
31:17	RSV	-	-	保留

比特	名称	属性	复位值	描述
16	COUNTFLAG	R	0	Systick定时器溢出标志 0: Systick定时器未发生溢出 1: Systick定时器发生下溢出 读该寄存器，可清除COUNTFLAG标志
15:3	RSV	-	-	保留
2	CLKSOURCE	R/W	1	SysTick时钟源选择 0: 外部参考时钟 1: HCLK
1	TICKINT	R/W	0	SysTick中断使能 0: 禁止中断 1: 使能中断
0	ENABLE	R/W	0	SysTick定时器使能 0: 禁止SysTick 1: 使能SysTick

27.2.2 重载值寄存器 SYS_RVR (偏移: 04H)

比特	名称	属性	复位值	描述
31:24	RSV	-	-	保留
23:0	RELOAD	R/W	0xFFFFFFFF	SysTick 定时器重载值

27.2.3 当前值寄存器 SYS_CVR (SysTick->VAL) (偏移: 08H)

比特	名称	属性	复位值	描述
31:24	RSV	-	-	保留
23:0	CURRENT	R/W	0xFFFFFFFF	读该寄存器, 获取 SysTick 定时器的当前计数值; 写任意值到该寄存器, 清零该寄存器及 COUNTFLAG。

27.3 使用流程

由于 SysTick 定时器的重载值和当前值在复位时都是未定义的，为了防止产生异常结果，对 SysTick 的配置需要遵循一定的流程：

- 1. 配置 SysTick->CTRL. ENABLE 为 0，禁止 SysTick。

2. 配置 SysTick->CTRL.CLKSOURCE, 选择 SysTick 的时钟源。
3. 配置 SysTick->LOAD, 选择 SysTick 的溢出周期。
4. 向 SysTick->VAL 写入任意值, 清零 SysTick->VAL 及 SysTick->CTRL。
5. COUNTFLAG。
6. 配置 SysTick->CTRL.TICKINT 为 1, 使能 SysTick 中断。
7. 配置 SysTick->CTRL.ENABLE 为 1, 使能 SysTick。
8. 查询等待定时器溢出标志到来之后关闭和清空计数器, 或者在中断服务程序中读取 SysTick->CTRL 以清除溢出标志。

28 典型应用

28.1 参考原理图

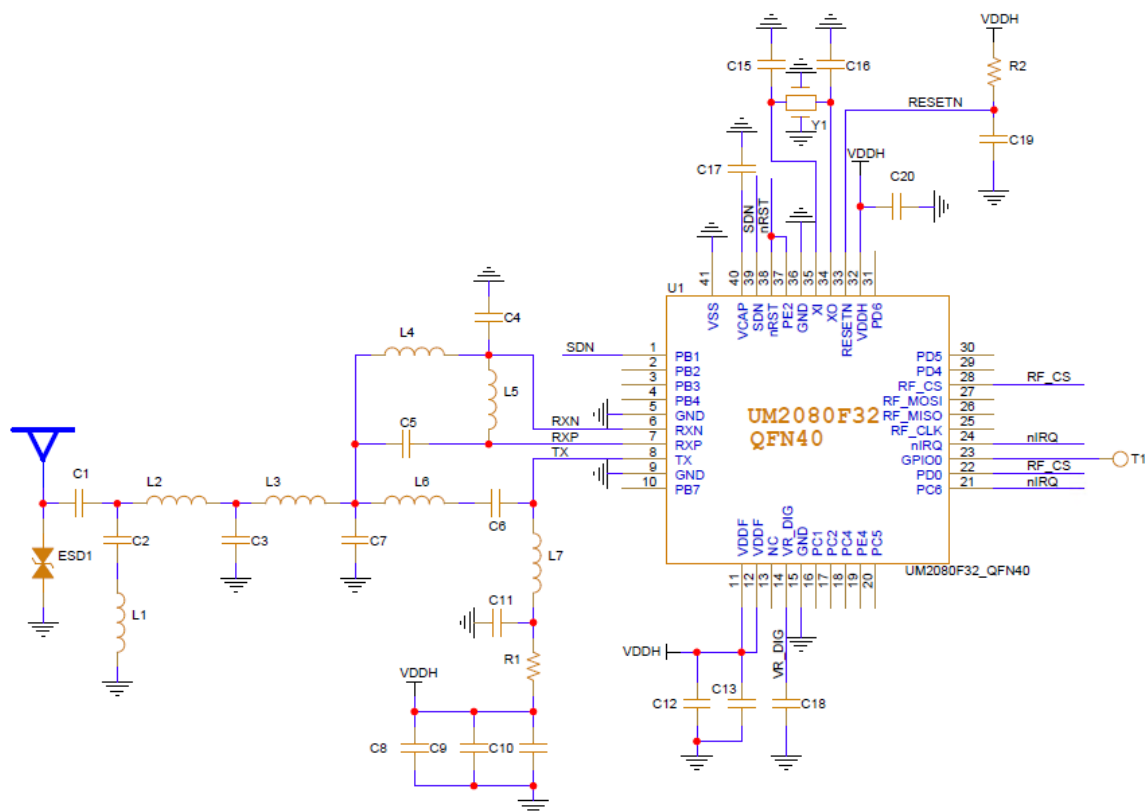


图 28-1: 参考原理图

- SDN 为 RF 的关断引脚，高电平为关断。如应用中需要控制 RF 关断/唤醒，必须给固定电平。
如应用不需要独立关断控制 RF，可接固定低电平
- nRST 为 RF 的复位引脚，低电平为复位状态。如应用需要控制 RF 复位，可连接 MCU GPIO 控制电平进行 RF 复位；如应用不需要独立控制复位，可接固定高电平或者悬空（内部默认上拉）
- 芯片关断时，SDN、GPIO0、Nrst（GPIO2）、RF_CS、RF_CLK、RF_MOSI、RF_MISO 处于高阻状态
- 芯片睡眠时，SDN、RF_CS、RF_CLK、RF_MOSI 为输入，RF_MISO 为输出低电平。GPIO0、GPIO2 方向保持，如为输出，则输出低电平
- ESD1 为 ESD 保护管，可根据应用要求来决定是否保留

28.2 +13dBm 匹配参数

表 28-1: +13dBm 参数表

位号	描述	元件值				单位
		315	433	868	915	MHz
C1	±5%, 0402, NPO, 50V, YAGEO	-	20	-	-	pF
	±5%, 0402, 贴片电阻, YAGEO	0	-	-	-	Ω
	±5%, 0402, 贴片高频电感, Sunlord	-	-	5.6	5.6	nH
C2	±5%, 0402, NPO, 50V, YAGEO	5	2.7	2.7	2.7	pF
C3	±5%, 0402, NPO, 50V, YAGEO	8.2	8.2	5	5	pF
C4	±5%, 0402, NPO, 50V, YAGEO	4.7	8.2	2	2	pF
C5	±5%, 0402, NPO, 50V, YAGEO	5.6	3	2	2	pF
C6	±5%, 0402, NPO, 50V, YAGEO	8.2	5.6	4.7	4.7	pF
C7	±5%, 0402, NPO, 50V, YAGEO	3.3	3	3	3	pF
C8	±10%, 0402, X7R, 50V, YAGEO	470	470	470	470	pF
C9	±10%, 0402, X7R, 50V, YAGEO	100	100	100	100	nF
C10	±10%, 0402, X7R, 50V, YAGEO	2.2	2.2	2.2	2.2	μF
C11	±10%, 0402, X7R, 50V, YAGEO	100	100	100	100	nF
C12	±10%, 0402, X7R, 50V, YAGEO	100	100	100	100	nF
C13	±10%, 0603, X7R, 50V, YAGEO	10	10	10	10	μF
C15	±5%, 0402, NPO, 50 V, YAGEO	20	20	20	20	pF
C16	±5%, 0402, NPO, 50 V, YAGEO	20	20	20	20	pF
C17	±10%, 0603, X7R, 50V, YAGEO	4.7	4.7	4.7	4.7	μF
C18	±10%, 0402, X7R, 50V, YAGEO	10	10	10	10	nF
C19	±10%, 0402, X7R, 50V, YAGEO	1	1	1	1	μF
C20	±10%, 0402, X7R, 50V, YAGEO	1	1	1	1	μF
L1	±5%, 0402, 贴片高频电感, Sunlord	12	10	2.7	2.7	nH
L2	±5%, 0603, 贴片高频电感, Sunlord	47	33	10	10	nH
L3	±5%, 0603, 贴片高频电感, Sunlord	68	39	8.2	8.2	nH
L4	±5%, 0402, 贴片高频电感, Sunlord	68	33	12	12	nH
L5	±5%, 0402, 贴片高频电感, Sunlord	82	33	12	12	nH
L6	±5%, 0603, 贴片高频电感, Sunlord	56	33	12	12	nH
L7	±5%, 0402, 贴片高频电感, Sunlord	220	180	100	100	nH
R1	±5%, 0402, 贴片电阻, YAGEO	4.7	4.7	4.7	4.7	Ω
R2	±10%, 0402, 贴片电阻, YAGEO	4.7	4.7	4.7	4.7	kΩ
Y1	±10ppm, 12pF, SMD3225-4P	26	26	26	26	MHz

位号	描述	元件值				单位
		315	433	868	915	MHz
ESD1	ESD 保护管	-	-	-	-	-

28.3 +18dBm 匹配参数

表 28-2: +18dBm 参数表

位号	描述	元件值				单位
		315	433	868	915	MHz
C1	±5%, 0402, NPO, 50V, YAGEO	-	20	-	-	pF
	±5%, 0402, 贴片高频电感, Sunlord	27	-	5.6	5.6	nH
C2	±5%, 0402, NPO, 50V, YAGEO	5	2.7	2.7	2.7	pF
C3	±5%, 0402, NPO, 50V, YAGEO	8.2	8.2	5	5	pF
C4	±5%, 0402, NPO, 50V, YAGEO	4.7	8.2	2	2	pF
C5	±5%, 0402, NPO, 50V, YAGEO	5.6	3	2	2	pF
C6	±5%, 0402, NPO, 50V, YAGEO	8.2	5.6	3.9	3.9	pF
C7	±5%, 0402, NPO, 50V, YAGEO	3.3	3	5	5	pF
C8	±10%, 0402, X7R, 50V, YAGEO	470	470	470	470	pF
C9	±10%, 0402, X7R, 50V, YAGEO	100	100	100	100	nF
C10	±10%, 0402, X7R, 50V, YAGEO	2.2	2.2	2.2	2.2	μF
C11	±10%, 0402, X7R, 50V, YAGEO	100	100	100	100	nF
C12	±10%, 0402, X7R, 50V, YAGEO	100	100	100	100	nF
C13	±10%, 0603, X7R, 50V, YAGEO	10	10	10	10	μF
C15	±5%, 0402, NPO, 50 V, YAGEO	20	20	20	20	pF
C16	±5%, 0402, NPO, 50 V, AGEO	20	20	20	20	pF
C17	±10%, 0603, X7R, 50V, YAGEO	4.7	4.7	4.7	4.7	μF
C18	±10%, 0402, X7R, 50V, YAGEO	10	10	10	10	nF
C19	±10%, 0402, X7R, 50V, YAGEO	1	1	1	1	μF
C20	±10%, 0402, X7R, 50V, YAGEO	1	1	1	1	μF
L1	±5%, 0402, 贴片高频电感, Sunlord	12	10	2.7	2.7	nH
L2	±5%, 0603, 贴片高频电感, Sunlord	47	33	10	10	nH
L3	±5%, 0603, 贴片高频电感, Sunlord	22	22	8.2	8.2	nH
L4	±5%, 0402, 贴片高频电感, Sunlord	68	33	12	12	nH
L5	±5%, 0402, 贴片高频电感, Sunlord	82	33	12	12	nH
L6	±5%, 0603, 贴片高频电感, Sunlord	33	22	8.2	8.2	nH
L7	±5%, 0402, 贴片高频电感, Sunlord	220	180	100	100	nH
R1	±5%, 0402, 贴片电阻, YAGEO	4.7	4.7	4.7	4.7	Ω

位号	描述	元件值				单位
		315	433	868	915	MHz
R2	±10%, 0402, 贴片电阻, YAGEO	4.7	4.7	4.7	4.7	kΩ
Y1	±10ppm, 12pF, SMD3225-4P	26	26	26	26	MHz
ESD1	ESD 保护管	-	-	-	-	-

29 封装尺寸

29.1 QFN40 (5*5 mm)

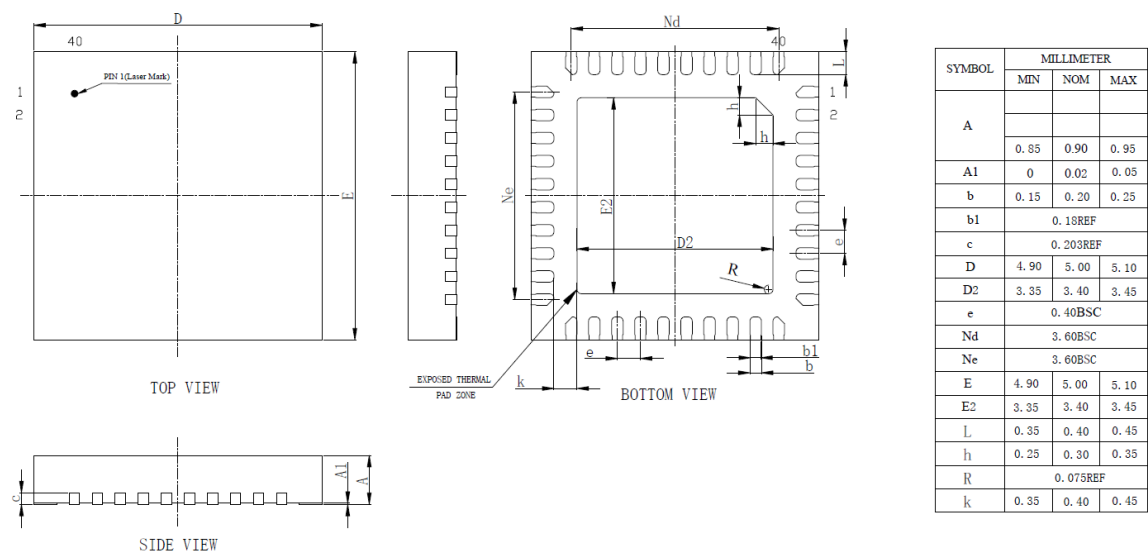


图 29-1: QFN40 封装图

30 版本维护

版本	日期	描述
V1.0	2022.07.05	初始版
V1.1	2023.10.16	1. 修改部分寄存器描述; 2. 更新电气参数值, 且表格重新排版; 3. 更新应用参考原理图及参数清单表格; 4. 调整功能描述、芯片运行、数据处理机制内容及排版。
V1.2	2026.07.06	1. 调整排版; 2. 删除 BUCK DCDC 模式; 3. 调整功能框图; 4. 调整 RF 的描述。