

UM2010 用户手册

版本: V1.4



UNICMICRO
广芯微电子

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，其著作产权归广芯微电子（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特点，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
2	功能框图	2
3	封装及管脚描述	3
3.1	封装管脚分布	3
3.2	引脚功能描述	3
4	寄存器定义	5
4.1	Reg00 Address: 0x00 Default: 0x1D	5
4.2	Reg01 Address: 0x01 Default: 0x78	5
4.3	Reg02 Address: 0x02 Default: 0x00	5
4.4	Reg03 Address: 0x03 Default: 0x00	6
4.5	Reg04 Address: 0x04 Default: 0x00	6
4.6	Reg05 Address: 0x05 Default: 0x03	6
4.7	Reg06 Address: 0x06 Default: 0x33	6
4.8	Reg07 Address: 0x07 Default: 0x33	6
4.9	Reg08 Address: 0x08 Default: 0x10	7
4.10	Reg09 Address: 0x09 Default: 0x00	7
4.11	Reg0A Address: 0x0A Default: 0x00	7
4.12	Reg0B Address: 0x0B Default: 0x00	7
4.13	Reg0C Address: 0x0C Default: 0x50	7
4.14	Reg0D Address: 0x0D Default: 0x30	8
4.15	Reg0E Address: 0x0E Default: 0x32	8
4.16	Reg10 Address: 0x10 Default: 0x40	9
4.17	Reg11 Address: 0x11 Default: 0x00	10
4.18	Reg1D Address: 0x1D Default: 0x80	10
4.19	Reg20 Address: 0x20 Default: 0x50	11
4.20	Reg21 Address: 0x21 Default: 0x60	11
4.21	Reg27 Address: 0x27 Default: 0x60	11
4.22	Reg28 Address: 0x28 Default: 0x19	12
4.23	Reg2B Address: 0x2B Default: 0xFF	12
4.24	Reg2C Address: 0x2C Default: 0xFF	12
4.25	Reg2D Address: 0x2D Default: 0x10	13
4.26	Reg30 Address: 0x30 Default: 0x80	13
4.27	Reg31 Address: 0x31 Default: 0x66	13
4.28	Reg37 Address: 0x37 Default: 0x06	14
4.29	Reg38 Address: 0x38 Default: 0x91	14

4.30	Reg4D Address: 0x4D Default: 0x10	14
4.31	Reg4E Address: 0x4E Default: 0x45	15
4.32	Reg4F Address: 0x4F Default: 0x67	15
4.33	Reg50 Address: 0x50 Default: 0x00	16
4.34	Reg51 Address: 0x51 Default: 0x00	16
4.35	Reg52 Address: 0x52 Default: 0x00	16
4.36	Reg53 Address: 0x53 Default: 0x00	16
4.37	Reg54 Address: 0x54 Default: 0x00	16
4.38	Reg55 Address: 0x55 Default: 0x00	17
4.39	Reg56 Address: 0x56 Default: 0xB7	17
4.40	Reg57 Address: 0x57 Default: 0x00	17
4.41	Reg58 Address: 0x58 Default: 0x00	17
4.42	Reg5D Address: 0x5D Default: 0x00	18
4.43	Reg60 Address: 0x60 Default: 0x80	18
4.44	Reg61 Address: 0x61 Default: 0x00	18
4.45	Reg64 Address: 0x64 Default: 0x00	19
4.46	Reg67 Address: 0x67 Default: 0x80	19
4.47	Reg68 Address: 0x68 Default: 0x00	19
4.48	Reg69 Address: 0x69 Default: 0x00	19
4.49	Reg6A Address: 0x6A Default: 0x00	20
4.50	Reg6B Address: 0x6B Default: 0x00	20
4.51	Reg6C Address: 0x6C Default: 0x00	20
4.52	Reg6D Address: 0x6D Default: 0x00	20
4.53	Reg6E Address: 0x6E Default: 0x00	20
4.54	Reg6F Address: 0x6F Default: 0x00	21
4.55	Reg70 Address: 0x70 Default: 0xFF	21
4.56	Reg71 Address: 0x71 Default: 0x09	21
4.57	Reg72 Address: 0x72 Default: 0x01	21
4.58	Reg73 Address: 0x73 Default: 0x55	22
4.59	Reg74 Address: 0x74 Default: 0x03	22
4.60	Reg75 Address: 0x75 Default: 0xA7	23
4.61	Reg76 Address: 0x76 Default: 0xA7	23
4.62	Reg77 Address: 0x77 Default: 0x98	23
4.63	Reg78 Address: 0x78 Default: 0xF3	23
4.64	Reg79 Address: 0x79 Default: 0x98	23
4.65	Reg7A Address: 0x7A Default: 0xF3	23
4.66	Reg7B Address: 0x7B Default: 0x98	24
4.67	Reg7C Address: 0x7C Default: 0xF3	24

4.68	Reg7D Address: 0x7D Default: 0x0E	24
4.69	Reg7E Address: 0x7E Default: 0x00	24
4.70	Reg7F Address: 0x7F Default: 0x00	24
4.71	Reg80 Address: 0x80 Default: 0x1D	25
4.72	Reg81 Address: 0x81 Default: 0x78	25
4.73	Reg82 Address: 0x82 Default: 0x00	26
4.74	Reg83 Address: 0x83 Default: 0x00	26
4.75	Reg84 Address: 0x84 Default: 0x00	26
4.76	Reg85 Address: 0x85 Default: 0x03	26
4.77	Reg86 Address: 0x86 Default: 0x33	26
4.78	Reg87 Address: 0x87 Default: 0x33	27
4.79	Reg88 Address: 0x88 Default: 0x10	27
4.80	Reg89 Address: 0x89 Default: 0x00	27
4.81	Reg8A Address: 0x8A Default: 0x00	27
4.82	Reg8B Address: 0x8B Default: 0x00	28
4.83	Reg8C Address: 0x8C Default: 0x50	28
4.84	Reg8D Address: 0x8D Default: 0x30	28
4.85	Reg8E Address: 0x8E Default: 0x32	28
4.86	Reg94 Address: 0x94 Default: 0x08	28
4.87	Reg95 Address: 0x95 Default: 0x20	29
4.88	Reg96 Address: 0x96 Default: 0x26	29
4.89	Reg97 Address: 0x97 Default: 0x1F	29
4.90	Reg98 Address: 0x98 Default: 0x68	29
4.91	Reg99 Address: 0x99 Default: 0x04	30
4.92	Reg9A Address: 0x9A Default: 0xAE	30
4.93	Reg9B Address: 0x9B Default: 0x89	30
4.94	Reg9C Address: 0x9C Default: 0x84	30
4.95	Reg9D Address: 0x9D Default: 0x03	30
4.96	Reg9E Address: 0x9E Default: 0x80	31
4.97	Reg9F Address: 0x9F Default: 0x26	31
4.98	RegA0 Address: 0xA0 Default: 0x50	31
4.99	RegA3 Address: 0xA3 Default: 0x42	31
4.100	RegE0 Address: 0xE0 Default: 0x80	32
4.101	RegE1 Address: 0xE1 Default: 0x00	32
4.102	RegE2 Address: 0xE2 Default: 0x01	32
4.103	RegE3 Address: 0xE3 Default: 0x40	32
5	功能描述	33
5.1	接收机	33

5.2	发射机	33
5.3	系统复位	34
5.3.1	POR复位	34
5.3.2	外部复位	34
5.3.3	软件复位	35
5.4	频率综合器	35
5.5	晶体振荡器	37
5.6	AGC	38
5.7	RSSI	38
5.7.1	自动计算RSSI	39
5.7.2	手动调节增益	40
5.7.3	RSSI自动过滤功能	41
5.8	AFC	41
6	芯片运行	42
6.1	SPI接口	42
6.2	FIFO	43
6.2.1	FIFO的工作模式	44
6.2.2	FIFO的中断时序	44
6.2.3	FIFO的应用场景	44
6.3	GPIO和中断	46
6.4	启动时序和状态机	49
6.4.1	状态机控制图	49
6.4.2	工作模式	50
6.4.3	状态机说明	50
7	数据包及包处理机制	52
7.1	直通模式 (Direct)	52
7.1.1	直通模式操作流程	53
7.2	数据包 (Packet)	53
7.2.1	数据包模式	54
7.2.2	数据包操作流程	56
7.2.3	数据包字段	57
8	典型应用	75
8.1	参考原理图	75
8.2	+13dBm匹配参数	76
8.3	+18dBm匹配参数	77
9	封装尺寸	78
9.1	QFN20 (4*4mm)	78

10 版本维护	79
---------------	----

unicmicro

表目录

表 5-1: 外部复位相关寄存器.....	35
表 5-2: 软复位相关寄存器.....	35
表 5-3: 频率综合器相关寄存器.....	36
表 5-4: 晶振相关寄存器	37
表 5-5: AGC相关寄存器	38
表 5-6: RSSI相关寄存器列表.....	38
表 5-7: Reg2B增益控制表.....	40
表 5-8: AFC相关寄存器.....	41
表 6-1: SPI时序参数	43
表 6-2: SPI相关寄存器.....	43
表 6-3: FIFO相关寄存器列表.....	44
表 6-4: GPIO配置相关寄存器列表	47
表 6-5: GPIO输出功能描述	48
表 6-6: 工作模式表	50
表 7-1: Direct相关寄存器列表.....	52
表 7-2: Mode相关寄存器列表.....	54
表 7-3: Preamble配置相关寄存器列表.....	57
表 7-4: Sync配置相关寄存器列表	58
表 7-5: Payload相关寄存器	60
表 7-6: Length相关的寄存器列表.....	63
表 7-7: Address配置相关寄存器列表	68
表 7-8: seqnum相关的寄存器列表	69
表 7-9: FCS2相关的寄存器列表.....	70
表 7-10: CRC相关的寄存器列表.....	71
表 7-11: 常用CRC标准和多项式列表	73
表 8-1: +13dBm匹配参考参数表.....	76
表 8-2: 天线匹配参考参数表.....	77

图目录

图 2-1: 功能框图	2
图 5-1: POR复位时序图	34
图 5-2: RSSI线性图	40
图 6-1: SPI读写寄存器时序图	42
图 6-2: SPI读写FIFO时序图	43
图 6-3: 状态机控制图	49
图 7-1: Mode 0帧格式	54
图 7-2: Mode 1帧格式	55
图 7-3: Mode 2帧格式	55
图 7-4: Mode 3帧格式	56
图 7-5: PAYLOAD_BIT_ORDER操作	61
图 7-6: PN7抽出模式移位顺序	62
图 7-7: PN9插入模式移位顺序	62
图 7-8: m3_length_en=0时MODE 3帧格式	64
图 7-9: 可变长度包格式, Address不存在	65
图 7-10: Address存在, Address在Length Byte之前	66
图 7-11: Address存在, Address在Length Byte之后	67
图 7-12: ACK的包格式	71
图 7-13: CRC编码范围	72
图 7-14: CRC_INV	73
图 7-15: CRC_BIT_ORDER	73
图 7-16: CRC8的配置示例	74
图 7-17: CRC16的配置示例	74
图 8-1: 应用参考原理图	75
图 9-1: QFN20封装图	78

1 系统概述

UM2010 是一款工作于 200MHz~960MHz 范围内的低功耗、高性能、单片集成的(G)FSK/OOK 无线收发机芯片。内部集成完整的射频接收机、射频发射机、频率综合器、调制解调器，只需配备简单、低成本的外围器件就可以获得良好的收发性能。

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

典型应用：

- 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

2 功能框图

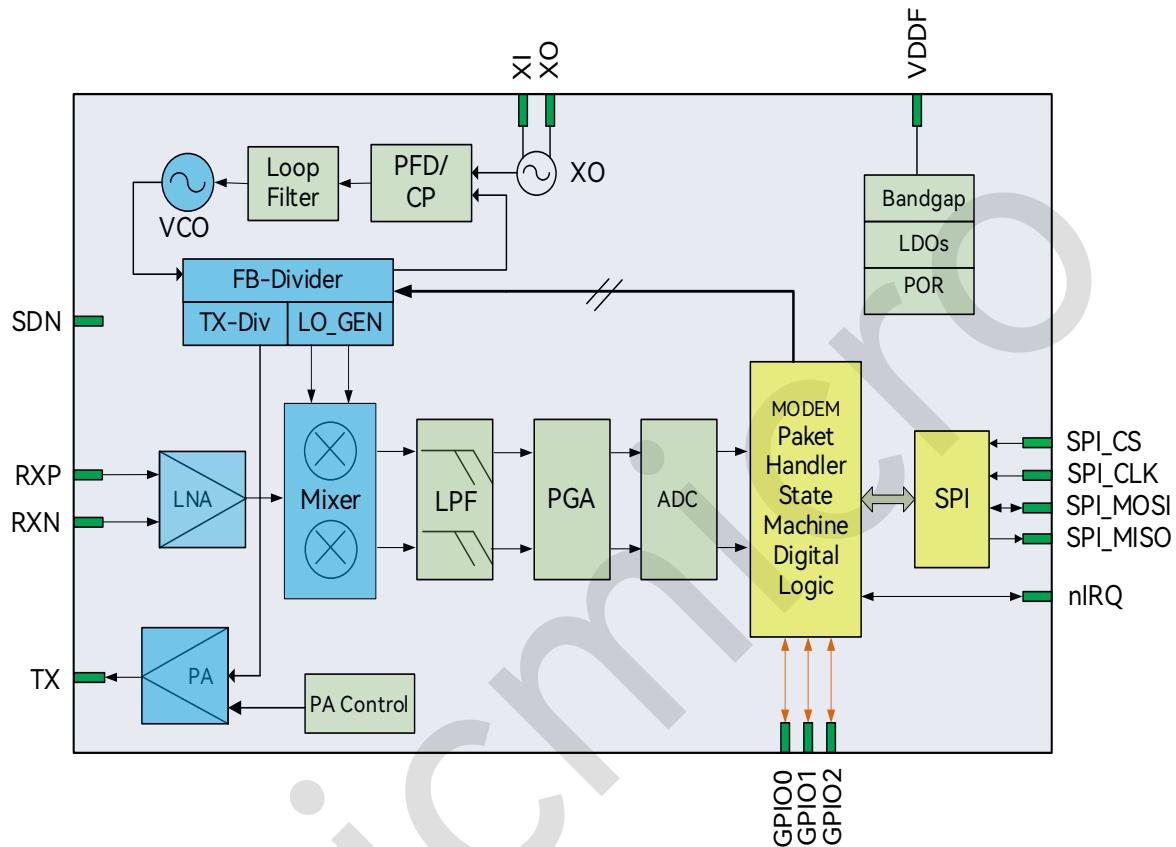


图 2-1: 功能框图

3 封装及管脚描述

3.1 封装管脚分布

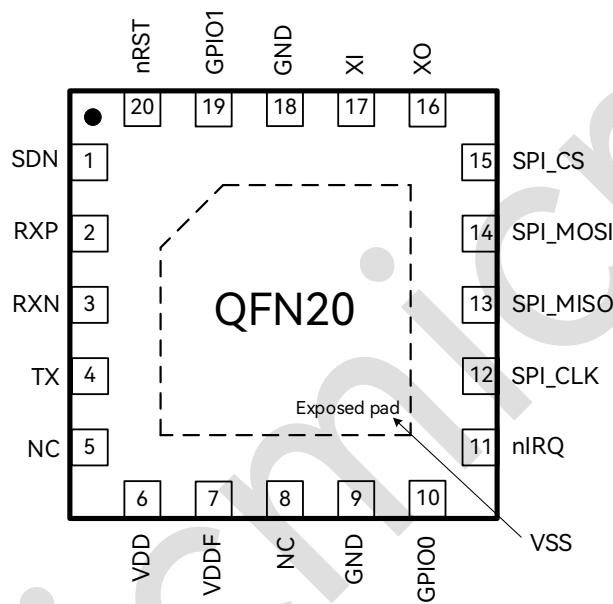


图 3-1: QFN20 封装管脚分布图

3.2 引脚功能描述

表 3-1: 引脚功能说明

引脚编号	管脚名称	IO 类型	关断状态	功能描述
0	VSS	G	-	芯片地(LF PAD)公共地
1	SDN	DI	HZ	芯片关断使能, SDN 高电平时芯片处于关断模式
2	RXP	RFI	-	射频正端输入
3	RXN	RFI	-	射频负端输入
4	TX	RFO	-	射频 PA 输出
5	NC	NC	-	空脚, 未连接任何内部电路
6	VDD	P	-	1.8V~3.6V 电源输入
7	VDDF	P	-	RF LDO 供电电源

引脚编号	管脚名称	IO 类型	关断状态	功能描述
8	NC	NC	-	空脚, 未连接任何内部电路
9	GND	G	-	芯片地
10	GPIO0	DIO	HZ	可配置 GPIO0
11	nIRQ	DIO	HZ	可配置 GPIO, 默认为中断输出
12	SPI_CLK	DI	HZ	SPI 时钟
13	SPI_MISO	DO	HZ	SPI 数据输出
14	SPI_MOSI	DIO	HZ	SPI 数据输入 (或三线制输入输出)
15	SPI_CS	DI	HZ	片选信号
16	XO	AO	-	晶振输出
17	XI	AI	-	晶振输入
18	GND	G	-	芯片地
19	GPIO1	DIO	HZ	可配置 GPIO1
20	nRST	DIO	HZ	可配置 GPIO2, 默认为外部复位引脚

说明: RF——射频信号; A——模拟信号; D——数字信号; I——Input; O——Output; G——Ground;

P——Power; PU——pull up 上拉; PD——pull down 下拉; HZ——高阻状态。

4 寄存器定义

4.1 Reg00 Address: 0x00 Default: 0x1D

Bit	Name	Type	Description	Default
7	reserved	R/W	-	1'b0
6:0	rf_f0[30:24]	R/W	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。	7'h1D

4.2 Reg01 Address: 0x01 Default: 0x78

Bit	Name	Type	Description	Default
7:0	rf_f0[23:16]	R/W	信道频率 f0[23:16]	8'h78

4.3 Reg02 Address: 0x02 Default: 0x00

Bit	Name	Type	Description	Default
7:0	rf_f0[15:8]	R/W	信道频率 f0[15:8]	8'h00

4.4 Reg03 Address: 0x03 Default: 0x00

Bit	Name	Type	Description	Default
7:0	rf_f0[7:0]	R/W	信道频率 f0[7:0]	8'h00

4.5 Reg04 Address: 0x04 Default: 0x00

Bit	Name	Type	Description	Default
7:0	ch_num	R/W	信道号 ch_num, 参考 Reg00 描述	8'h00

4.6 Reg05 Address: 0x05 Default: 0x03

Bit	Name	Type	Description	Default
7:0	f_step[23:16]	R/W	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	8'h03

4.7 Reg06 Address: 0x06 Default: 0x33

Bit	Name	Type	Description	Default
7:0	f_step[15:8]	R/W	信道步进频率 f_step[15:8], 参考 Reg00 描述	8'h33

4.8 Reg07 Address: 0x07 Default: 0x33

Bit	Name	Type	Description	Default
7:0	f_step[7:0]	R/W	信道步进频率 f_step[7:0], 参考 Reg00 描述	8'h33

4.9 Reg08 Address: 0x08 Default: 0x10

Bit	Name	Type	Description	Default
7	Reserved	R/W	-	1'b0
6:0	ref_freq[30:24]	R/W	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。	7'h10

4.10 Reg09 Address: 0x09 Default: 0x00

Bit	Name	Type	Description	Default
7:0	ref_freq[23:16]	R/W	参考 Reg08 描述	8'h00

4.11 Reg0A Address: 0x0A Default: 0x00

Bit	Name	Type	Description	Default
7:0	ref_freq[15:8]	R/W	参考 Reg08 描述	8'h00

4.12 Reg0B Address: 0x0B Default: 0x00

Bit	Name	Type	Description	Default
7:0	ref_freq[7:0]	R/W	参考 Reg08 描述	8'h00

4.13 Reg0C Address: 0x0C Default: 0x50

Bit	Name	Type	Description	Default
7:0	tx_preamble_len	R/W	发射 Preamble 长度, 单位字节	8'h50

4.14 Reg0D Address: 0x0D Default: 0x30

Bit	Name	Type	Description	Default
7	payload_bit_order	R/W	Payload bit 顺序: 0: LSB 低位在前 1: MSB 高位在前	1'b0
6	manchester_inv	R/W	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	1'b0
5	syncword_en	R/W	Syncword 同步字使能: 0: Disable 1: Enable	1'b1
4	preamble_en	R/W	设置 Preamble 前导码使能: 0: Disable 1: Enable	1'b1
3:2	pkt_enc_type	R/W	数据包编码: 00: NRZ 01: 曼彻斯特编码 10: 无效 11: 交织编码	2'b00
1:0	fec_type	R/W	数据包 FEC 类型: 00: NO FEC 01: 1/3 FEC 10: 2/3 FEC 11: 1/2 FEC	2'b00

4.15 Reg0E Address: 0x0E Default: 0x32

Bit	Name	Type	Description	Default
7	length_byte_swap	R/W	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	1'b0
6	length_sel	R/W	包长度控制域字节配置: 0: 1 字节 1: 2 字节	1'b0

Bit	Name	Type	Description	Default
5	crc_en	R/W	CRC 使能: 0: Disable 1: Enable	1'b1
4	scramble_en	R/W	数据白化 (扰码) 使能: 0: Disable 1: Enable	1'b1
3	fifo_share_en	R/W	FIFO 共享设置: 0: RX 和 TX 各 128 字节 FIFO 1: RX 和 TX 共用 256 字节 FIFO	1'b0
2	direct_mode	R/W	直通模式使能: 0: Disable 1: Enable	1'b0
1:0	packet_mode	R/W	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 中的第一个或 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	2'b10

4.16 Reg10 Address: 0x10 Default: 0x40

Bit	Name	Type	Description	Default
7:4	retx_times	R/W	Auto ACK 使能时, 发射端未收到 ACK 包时自动重发的次数	4'h4
3	rssi_val_sel	R/W	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	1'b0
2	preamble_int_en	R/W	接收 Preamble (前导码) 匹配中断使能: 0: Disable 1: Enable	1'b0
1	syncword_int_en	R/W	接收 Syncword (同步字) 匹配中断使能: 0: Disable 1: Enable	1'b0
0	reserved	R/W	-	1'b0

4.17 Reg11 Address: 0x11 Default: 0x00

Bit	Name	Type	Description	Default
7	reserved	R/W	-	1'b0
6	int_flag_clr	R/W	写‘1’清除中断标志	1'b0
5	miso_tri_opt	R/W	SPI 在非工作状态时, spi_miso 状态: 0: 高阻 1: 输出	1'b0
4:3	reserved	R/W	-	2'b00
2:0	brclk_sel	R/W	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	3'b000

4.18 Reg1D Address: 0x1D Default: 0x80

Bit	Name	Type	Description	Default
7:6	vdddig	R/W	数字 LDO 输出电压设置: 00: 0.8V 01: 1.0V 10: 1.2V 11: 1.4V	2'b10
5:0	reserved	R/W	-	6'h00

4.19 Reg20 Address: 0x20 Default: 0x50

Bit	Name	Type	Description	Default
7:6	frequency_band	R/W	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	2'b01
5:0	Reserved	R/W	-	6'h10

4.20 Reg21 Address: 0x21 Default: 0x60

Bit	Name	Type	Description	Default
7	reserved	R/W	-	1'b0
6	pa_ramp_en	R/W	PA RAMP 使能: 0: Disable 1: Enable	1'b1
5:0	pa_bias	R/W	PA_bias 控制: 000000: 最小 111111: 最大	6'h20

4.21 Reg27 Address: 0x27 Default: 0x60

Bit	Name	Type	Description	Default
7	reserved	R/W	-	1'b0
6	pa_drvh	R/W	PA 前级驱动设置: 0: Low 1: High	1'b1
5:0	pa_gain	R/W	PA 增益控制	6'h20

4.22 Reg28 Address: 0x28 Default: 0x19

Bit	Name	Type	Description	Default
7	rssi_Reg_clr	R/W	写‘1’清除被锁定的 RSSI	1'b0
6	rssi_sel	R/W	收到同步字后锁定 RSSI 值: 0: Disable 1: Enable	1'b0
5	rx_rssi_thr_en	R/W	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: Disable 1: Enable	1'b0
4	agc_en	R/W	AGC 使能: 0: Disable 1: Enable	1'b1
3:0	reserved	R/W	-	4'h9

4.23 Reg2B Address: 0x2B Default: 0xFF

Bit	Name	Type	Description	Default
7:6	reserved	R/W	-	2'b11
5:4	rxfe_gn	R/W	rxfe 增益控制	2'b11
3:2	filter_gn	R/W	Filter 增益控制	2'b11
1:0	pga_gn	R/W	PGA 增益控制	2'b11

4.24 Reg2C Address: 0x2C Default: 0xFF

Bit	Name	Type	Description	Default
7:0	rx_rssi_thr	R/W	接收 RSSI 门限值, 参考 Reg28[5]	8'hFF

4.25 Reg2D Address: 0x2D Default: 0x10

Bit	Name	Type	Description	Default
7	cw_mode	R/W	发射载波模式: 0: Disable 1: Enable	1'b0
6:4	reserved	R/W	-	3'b001
3	ook_en	R/W	调制解调模式选择: 0: FSK 1: OOK	1'b0
2:0	reserved	R/W	-	3'b000

4.26 Reg30 Address: 0x30 Default: 0x80

Bit	Name	Type	Description	Default
7	sel_gau_o	R/W	发射高斯滤波器使能: 0: Disable 1: Enable	1'b1
6:4	reserved	R/W	-	3'b000
3:0	dev_set[11:8]	R/W	发射调制频偏 dev_set 的高 4 位	4'h0

4.27 Reg31 Address: 0x31 Default: 0x66

Bit	Name	Type	Description	Default
7:0	dev_set[7:0]	R/W	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	8'h66

4.28 Reg37 Address: 0x37 Default: 0x06

Bit	Name	Type	Description	Default
7:0	csrst_len	R/W	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	8'h06

4.29 Reg38 Address: 0x38 Default: 0x91

Bit	Name	Type	Description	Default
7:5	reserved	R/W	-	3'b100
4	demod_comp_en	R/W	解调频偏补偿: 0: Disable 1: Enable	1'b1
3:0	reserved	R/W	-	4'h1

4.30 Reg4D Address: 0x4D Default: 0x10

Bit	Name	Type	Description	Default
7	nirq_dir	R/W	nIRQ_flag IO 口方向: 0: Output 1: Input	1'b0
6	gpio_dir[0]	R/W	gpio[0] 方向: 0: Output 1: Input	1'b0
5	gpio_dir[1]	R/W	gpio[1] 方向: 0: Output 1: Input	1'b0
4	gpio_dir[2]	R/W	gpio[2] 方向, 默认复位输入功能: 0: Output 1: Input	1'b1

Bit	Name	Type	Description	Default
3:0	nirq_sel	R/W	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	4'h0

4.31 Reg4E Address: 0x4E Default: 0x45

Bit	Name	Type	Description	Default
7:4	gpio_0_sel	R/W	gpio[0]输出功能选择, 参考 Reg4D[3:0]描述	4'h4
3:0	gpio_1_sel	R/W	gpio[1]输出功能选择, 参考 Reg4D[3:0]描述	4'h5

4.32 Reg4F Address: 0x4F Default: 0x67

Bit	Name	Type	Description	Default
7:4	gpio_2_sel	R/W	gpio[2]输出功能选择, 参考 Reg4D[3:0]描述	4'h6
3:0	reserved	R/W	-	4'h7

4.33 Reg50 Address: 0x50 Default: 0x00

Bit	Name	Type	Description	Default
7:0	rx_fifo_wr_ptr	R/W	RX FIFO 写指针, 写 0x80 清除指针	8'h00

4.34 Reg51 Address: 0x51 Default: 0x00

Bit	Name	Type	Description	Default
7:0	rx_fifo_rd_ptr	R/W	RX FIFO 读指针, 写 0x80 清除指针	8'h00

4.35 Reg52 Address: 0x52 Default: 0x00

Bit	Name	Type	Description	Default
7:0	rx_fifo_data	R/W	RX FIFO	8'h00

4.36 Reg53 Address: 0x53 Default: 0x00

Bit	Name	Type	Description	Default
7:0	tx_fifo_wr_ptr	R/W	TX FIFO 写指针, 写 0x80 清除指针	8'h00

4.37 Reg54 Address: 0x54 Default: 0x00

Bit	Name	Type	Description	Default
7:0	tx_fifo_rd_ptr	R/W	TX FIFO 读指针, 写 0x80 清除指针	8'h00

4.38 Reg55 Address: 0x55 Default: 0x00

Bit	Name	Type	Description	Default
7:0	tx_fifo_data	R/W	TX FIFO	8'h00

4.39 Reg56 Address: 0x56 Default: 0xB7

Bit	Name	Type	Description	Default
7:0	delta_rssi	R/W	计算 RSSI 时的预偏移量	8'hB7

4.40 Reg57 Address: 0x57 Default: 0x00

Bit	Name	Type	Description	Default
7:3	reserved	R/W	-	5'h00
2	spi_3w_sel	R/W	三线 SPI 选择: 0: 4 线标准 SPI 1: 3 线 SPI, 数据线只使用 SPI_MOSI	1'b0
1	crc_err_st	R/W	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	1'b0
0	reserved	R/W	-	1'b0

4.41 Reg58 Address: 0x58 Default: 0x00

Bit	Name	Type	Description	Default
7:5	reserved	R/W	-	3'b000
4:3	direct_tx_data_sel	R/W	Direct mode 时, TX 数据源选择: 00: nIRQ 引脚 01: gpio[0] 1x: gpio[1]	2'b00
2:0	reserved	R/W	-	3'b000

4.42 Reg5D Address: 0x5D Default: 0x00

Bit	Name	Type	Description	Default
7	nrst_sel	R/W	nRST 引脚功能设置: 0: nRST 1: GPIO2	1'b0
6	pull_up_en	R/W	nRST 引脚 Pull-UP 使能: 0: Enable 1: Disable	1'b0
5:0	reseved	R/W	-	5'h00

4.43 Reg60 Address: 0x60 Default: 0x80

Bit	Name	Type	Description	Default
7:0	command	R/W	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	8'h80

4.44 Reg61 Address: 0x61 Default: 0x00

Bit	Name	Type	Description	Default
7	syncword_rec	R	接收到正确的 Syncword	1'b0
6	preamble_rec	R	接收到有效的 Preamble	1'b0
5	crc_error	R	高电平时表示 CRC 校验错误	1'b0
4	pkt_flag	R	pkt_flag	1'b0
3	fifo_flag	R	fifo_flag	1'b0
2:0	reserved	R	-	3'b000

4.45 Reg64 Address: 0x64 Default: 0x00

Bit	Name	Type	Description	Default
7:0	rssi	R	读出的值为实际 RSSI 的绝对值	8'h00

4.46 Reg67 Address: 0x67 Default: 0x80

Bit	Name	Type	Description	Default
7	st_idle	R	IDLE 状态	1'b1
6	st_tx	R	发射状态	1'b0
5	st_rx	R	接收状态	1'b0
4	st_fson	R	频综开启状态	1'b0
3:0	reseved	R	-	4'h0

4.47 Reg68 Address: 0x68 Default: 0x00

Bit	Name	Type	Description	Default
7:2	reseved	R	-	6'h0
1	m3_seqnum_ok	R	数据包模式 3 时, 接收 seqnum 匹配正确	1'b0
0	m3_addr_comp_ok	R	数据包模式 3 时, 接收 address 匹配正确	1'b0

4.48 Reg69 Address: 0x69 Default: 0x00

Bit	Name	Type	Description	Default
7:0	m3_rx_fcs2_value	R	数据包模式 3 时, 接收 FCS2 数据	8'h00

4.49 Reg6A Address: 0x6A Default: 0x00

Bit	Name	Type	Description	Default
7:0	m3_rx_payload_len[7:0]	R	数据包模式 3 时, 接收 length 数据	8'h00

4.50 Reg6B Address: 0x6B Default: 0x00

Bit	Name	Type	Description	Default
7:0	m3_rx_payload_len[15:8]	R	数据包模式 3 时, 接收 length 数据	8'h00

4.51 Reg6C Address: 0x6C Default: 0x00

Bit	Name	Type	Description	Default
7:0	m3_rx_addr[7:0]	R	数据包模式 3 时, 接收 address	8'h00

4.52 Reg6D Address: 0x6D Default: 0x00

Bit	Name	Type	Description	Default
7:0	m3_rx_addr[15:8]	R	数据包模式 3 时, 接收 address	8'h00

4.53 Reg6E Address: 0x6E Default: 0x00

Bit	Name	Type	Description	Default
7:0	m3_rx_addr[23:16]	R	数据包模式 3 时, 接收 address	8'h00

4.54 Reg6F Address: 0x6F Default: 0x00

Bit	Name	Type	Description	Default
7:0	m3_rx_addr[31:24]	R	数据包模式 3 时, 接收地址域数据	8'h00

4.55 Reg70 Address: 0x70 Default: 0xFF

Bit	Name	Type	Description	Default
7:0	scramble_init_val[7:0]	R/W	scramble 初始化值	8'hFF

4.56 Reg71 Address: 0x71 Default: 0x09

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

4.57 Reg72 Address: 0x72 Default: 0x01

Bit	Name	Type	Description	Default
7:5	reserved	R/W	-	3'b000
4:0	sync_thres	R/W	同步字检测允许的错误位数	5'h01

4.58 Reg73 Address: 0x73 Default: 0x55

Bit	Name	Type	Description	Default
7:0	preamble_val	R/W	发射前导码时的使用的数据	8'h55

4.59 Reg74 Address: 0x74 Default: 0x03

Bit	Name	Type	Description	Default
7	auto_ack	R/W	auto_ack 使能: 0: Disable 1: Enable	1'b0
6	ack_payload_en	R/W	发射 ACK 时带 payload 数据返回: 0: Disable 1: Enable	1'b0
5	cont_rx_en	R/W	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	1'b0
4	sync_man_en	R/W	同步字曼彻斯特编码使能: 0: Disable 1: Enable	1'b0
3	sync_bit_order	R/W	同步字 bit 顺序: 0: 字节低位 1: 字节高位 MSB	1'b0
2:0	sync_len	R/W	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, 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}	3'b011

4.60 Reg75 Address: 0x75 Default: 0xA7

Bit	Name	Type	Description	Default
7:0	sync_id_1	R/W	同步字字节 1	8'hA7

4.61 Reg76 Address: 0x76 Default: 0xA7

Bit	Name	Type	Description	Default
7:0	sync_id_2	R/W	同步字字节 2	8'hA7

4.62 Reg77 Address: 0x77 Default: 0x98

Bit	Name	Type	Description	Default
7:0	sync_id_3	R/W	同步字字节 3	8'h98

4.63 Reg78 Address: 0x78 Default: 0xF3

Bit	Name	Type	Description	Default
7:0	sync_id_4	R/W	同步字字节 4	8'hF3

4.64 Reg79 Address: 0x79 Default: 0x98

Bit	Name	Type	Description	Default
7:0	sync_id_5	R/W	同步字字节 5	8'h98

4.65 Reg7A Address: 0x7A Default: 0xF3

Bit	Name	Type	Description	Default
7:0	sync_id_6	R/W	同步字字节 6	8'hF3

4.66 Reg7B Address: 0x7B Default: 0x98

Bit	Name	Type	Description	Default
7:0	sync_id_7	R/W	同步字字节 7	8'h98

4.67 Reg7C Address: 0x7C Default: 0xF3

Bit	Name	Type	Description	Default
7:0	sync_id_8	R/W	同步字字节 8	8'hF3

4.68 Reg7D Address: 0x7D Default: 0x0E

Bit	Name	Type	Description	Default
7:0	payload_len[7:0]	R/W	payload 长度, 在数据包模式 1 和数据包模式 3 有效	8'h0E

4.69 Reg7E Address: 0x7E Default: 0x00

Bit	Name	Type	Description	Default
7:0	payload_len[15:8]	R/W	payload 长度, 在数据包模式 1 和数据包模式 3 有效	8'h00

4.70 Reg7F Address: 0x7F Default: 0x00

Bit	Name	Type	Description	Default
7:1	reserved	R/W	-	7'h0
0	page_sel	R/W	寄存器页地址: 0: Reg00~Reg7E 1: Reg80~RegFE	1'b0

4.71 Reg80 Address: 0x80 Default: 0x1D

Bit	Name	Type	Description	Default
7:6	reserved	R/W	-	2'b00
5	m3_length_en	R/W	数据包模式 3 时, 数据长度域控制: 0: Disable 1: Enable	1'b0
4	m3_crc_sel	R/W	数据包模式 3 时, CRC 的校验范围: 0: All payload 1: 仅 FIFO 中的数据	1'b1
3:0	m3_ack_node_ctrl	R/W	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	4'hD

4.72 Reg81 Address: 0x81 Default: 0x78

Bit	Name	Type	Description	Default
7:6	reserved	R/W	-	2'b01
5:4	m3_addr_size	R/W	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'b11
3	reserved	R/W	-	1'b1
2	m3_addr_split_mode	R/W	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	1'b0
1	m3_addr_pos_sel	R/W	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	1'b0

Bit	Name	Type	Description	Default
0	m3_addr_en	R/W	数据包模式 3 时, 地址域控制: 0: Disable 1: Enable	1'b0

4.73 Reg82 Address: 0x82 Default: 0x00

Bit	Name	Type	Description	Default
7:0	m3_addr_val[7:0]	R/W	数据包模式 3 时, 地址值	8'h00

4.74 Reg83 Address: 0x83 Default: 0x00

Bit	Name	Type	Description	Default
7:0	m3_addr_val[15:8]	R/W	数据包模式 3 时, 地址值	8'h00

4.75 Reg84 Address: 0x84 Default: 0x00

Bit	Name	Type	Description	Default
7:0	m3_addr_val[23:16]	R/W	数据包模式 3 时, 地址值	8'h00

4.76 Reg85 Address: 0x85 Default: 0x03

Bit	Name	Type	Description	Default
7:0	m3_addr_val[31:24]	R/W	数据包模式 3 时, 地址值	8'h03

4.77 Reg86 Address: 0x86 Default: 0x33

Bit	Name	Type	Description	Default
7:0	m3_addr_bit_mask[7:0]	R/W	数据包模式 3 时, 地址过滤, mask 为‘1’时对应地址位不作比较	8'h33

4.78 Reg87 Address: 0x87 Default: 0x33

Bit	Name	Type	Description	Default
7:0	m3_addr_bit_mask[15:8]	R/W	数据包模式 3 时, 地址过滤, mask 为‘1’时对应地址位不作比较	8'h33

4.79 Reg88 Address: 0x88 Default: 0x10

Bit	Name	Type	Description	Default
7:0	m3_addr_bit_mask[23:16]	R/W	数据包模式 3 时, 地址过滤, mask 为‘1’时对应地址位不作比较	8'h10

4.80 Reg89 Address: 0x89 Default: 0x00

Bit	Name	Type	Description	Default
7:0	m3_addr_bit_mask[31:24]	R/W	数据包模式 3 时, 地址过滤, mask 为‘1’时对应地址位不作比较	8'h00

4.81 Reg8A Address: 0x8A Default: 0x00

Bit	Name	Type	Description	Default
7:6	reserved	R/W	-	2'b00
5	m3_seqnum_auto_inc	R/W	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	1'b0
4	m3_seqnum_en	R/W	数据包模式 3 时, seqnum 数据使能: 0: Disable 1: Enable	1'b0
3	m3_seqnum_bit_size	R/W	数据包模式 3 时, seqnum 数据长度: 0: 8bits 1: 16bits	1'b0
2:0	reserved	R/W	-	3'b000

4.82 Reg8B Address: 0x8B Default: 0x00

Bit	Name	Type	Description	Default
7:0	seqnum_Reg[7:0]	R/W	数据包模式 3 时, seqnum 值	8'h00

4.83 Reg8C Address: 0x8C Default: 0x50

Bit	Name	Type	Description	Default
7:0	seqnum_Reg[15:8]	R/W	数据包模式 3 时, seqnum 值	8'h50

4.84 Reg8D Address: 0x8D Default: 0x30

Bit	Name	Type	Description	Default
7:1	reserved	R/W	-	7'h30
0	m3_fcs2_en	R/W	数据包模式 3 时, FCS2 数据域使能: 0: Disable 1: Enable	1'b0

4.85 Reg8E Address: 0x8E Default: 0x32

Bit	Name	Type	Description	Default
7	m3_fcs2_val	R/W	数据包模式 3 时, ACK request	1'b0
6:0	reserved	R/W	-	7'h32

4.86 Reg94 Address: 0x94 Default: 0x08

Bit	Name	Type	Description	Default
7:5	reserved	R/W	-	3'b000
4	crc_bit_order	R/W	CRC bit 顺序: 0: LSB 1: MSB	1'b0
3	crc_inv	R/W	CRC bit 取反: 0: Disable 1: Enable	1'b1

Bit	Name	Type	Description	Default
2	crc_man_en	R/W	CRC 曼彻斯特编码使能: 0: Disable 1: Enable	1'b0
1:0	crc_len	R/W	CRC 长度: 00: 8 位 01: 16 位 10: 24 位 11: 32 位	2'b00

4.87 Reg95 Address: 0x95 Default: 0x20

Bit	Name	Type	Description	Default
7:0	crc_poly[7:0]	R/W	CRC 多项式配置	8'h20

4.88 Reg96 Address: 0x96 Default: 0x26

Bit	Name	Type	Description	Default
7:0	crc_poly[15:8]	R/W	CRC 多项式配置	8'h26

4.89 Reg97 Address: 0x97 Default: 0x1F

Bit	Name	Type	Description	Default
7:0	crc_poly[23:16]	R/W	CRC 多项式配置	8'h1F

4.90 Reg98 Address: 0x98 Default: 0x68

Bit	Name	Type	Description	Default
7:0	crc_poly[31:24]	R/W	CRC 多项式配置	8'h68

4.91 Reg99 Address: 0x99 Default: 0x04

Bit	Name	Type	Description	Default
7:0	crc_init_val[7:0]	R/W	CRC 移位寄存器初始化值	8'h04

4.92 Reg9A Address: 0x9A Default: 0xAE

Bit	Name	Type	Description	Default
7:0	crc_init_val[15:8]	R/W	CRC 移位寄存器初始化值	8'hAE

4.93 Reg9B Address: 0x9B Default: 0x89

Bit	Name	Type	Description	Default
7:0	crc_init_val[23:16]	R/W	CRC 移位寄存器初始化值	8'h89

4.94 Reg9C Address: 0x9C Default: 0x84

Bit	Name	Type	Description	Default
7:0	crc_init_val[31:24]	R/W	CRC 移位寄存器初始化值	8'h84

4.95 Reg9D Address: 0x9D Default: 0x03

Bit	Name	Type	Description	Default
7	tx_data_inv	R/W	发射 bit 数据取反	1'b0
6	rx_data_inv	R/W	解调器输出的 bit 数据取反	1'b0
5	auto_det_tx_chl	R/W	发射前检测信道 RSSI: 0: Disable 1: Enable	1'b0
4	auto_det_tx_mode	R/W	发射检测信道模式: 0: 检测次数满且信道仍然忙, 退出	1'b0

Bit	Name	Type	Description	Default
			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	发射前最大检测信道次数	4'h3

4.96 Reg9E Address: 0x9E Default: 0x80

Bit	Name	Type	Description	Default
7:0	chl_busy_thr	R/W	发射前检测信道门限值	8'h80

4.97 Reg9F Address: 0x9F Default: 0x26

Bit	Name	Type	Description	Default
7:6	reserved	R/W	-	2'b00
5:0	fifo_empty_thr	R/W	TX FIFO 空门限, FIFO 数据低于门限值时 fifo_flag 置‘1’	6'h26

4.98 RegA0 Address: 0xA0 Default: 0x50

Bit	Name	Type	Description	Default
7:6	reserved	R/W	-	2'b00
5:0	fifo_full_thr	R/W	RX FIFO 满门限, RX FIFO 数据高于门限值时 fifo_flag 置‘1’	6'h50

4.99 RegA3 Address: 0xA3 Default: 0x42

Bit	Name	Type	Description	Default
7	srst_cmd_enable	R/W	Reset Command 使能: 0: Disable 1: Enable	1'b0

Bit	Name	Type	Description	Default
6	reserved	R/W	-	1'b1
5:0	auto_ack_wait_time	R/W	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	6'h02

4.100 RegE0 Address: 0xE0 Default: 0x80

Bit	Name	Type	Description	Default
7:0	rx_seqnum_val[7:0]	R	数据包模式 3 时, 接收 seqnum 数据	8'h80

4.101 RegE1 Address: 0xE1 Default: 0x00

Bit	Name	Type	Description	Default
7:0	rx_seqnum_val[15:8]	R	数据包模式 3 时, 接收 seqnum 数据	8'h00

4.102 RegE2 Address: 0xE2 Default: 0x01

Bit	Name	Type	Description	Default
7:0	seqnum_cnt[7:0]	R	seqnum_cnt	8'h01

4.103 RegE3 Address: 0xE3 Default: 0x40

Bit	Name	Type	Description	Default
7:0	seqnum_cnt[15:8]	R	seqnum_cnt	8'h40

5 功能描述

UM2010 是一款高集成度的 sub-1GHz 无线收发机。支持 OOK、2-(G)FSK 调制解调方式，支持 Direct 和 Packet 数据处理模式，支持标准四线 SPI 和自定义三线 SPI 接口。

5.1 接收机

芯片内部集成了低功耗、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 管脚直接输出，也可以在包模式下通过 SPI 从内部 FIFO 中读取。

5.2 发射机

芯片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-20dBm 至+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。

5.3 系统复位

芯片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 拉低就可以完成对芯片的复位。此外，还可以通过 SPI 写命令的方式对芯片进行复位操作。芯片一旦复位，内部所有配置都恢复到上电默认状态，MCU 需要对芯片重新进入初始化操作。

5.3.1 POR 复位

芯片上电或者退出全局关断模式 (SDN 拉低) 后，芯片会触发上电复位 (POR)，复位之后芯片默认处于 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。

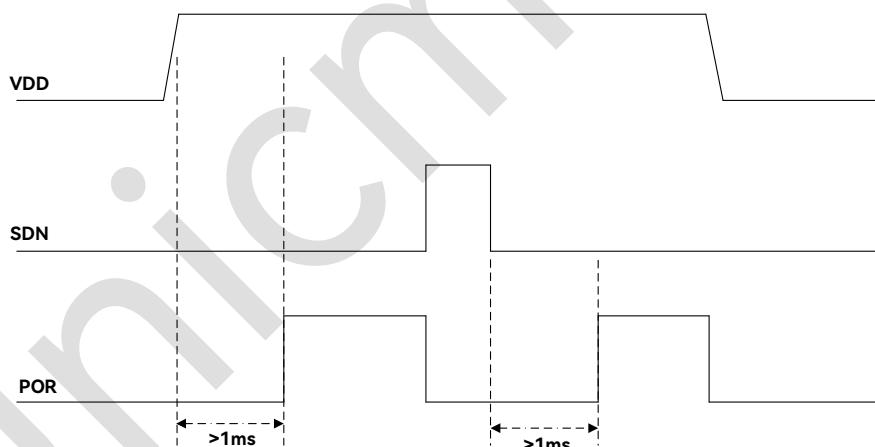


图 5-1: POR 复位时序图

5.3.2 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 引脚电平使芯片处于复位状态，拉高 nRST 引脚电平使芯片处于工作状态。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芯片复位。芯片外部复位后，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: Enable 1: Disable

5.3.3 软件复位

软件复位是通过 SPI 写命令的方式对芯片进行软复位操作，芯片除了 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。软复位是复位整个数字系统，包括 SPI 接口、寄存器、FIFO、数字内核。芯片复位后回到 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: Disable 1: Enable

5.4 频率综合器

芯片内部集成了高精度的 Sigma-Delta 小数分频频率综合器，在 200MHz~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：

$$F_c = F_0 + N \cdot F_{step}$$

其中, F_0 、N 和 F_{step} 都可以通过寄存器进行设置。

- F_0 由{Reg00[6:0],Reg01,Reg02,Reg03}共 31bits 位设置, 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 20 位为小数, 单位为 MHz。

如设置 $F_c=433.92\text{MHz}$, 可以设置 $F_0 = 433.92$, $N = 0$, $F_{step} = 0$ 。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 $f_0[30:24]$ 。 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 $f_0[23:16]$
0x02	7:0	R/W	rf_f0[15:8]	信道频率 $f_0[15:8]$
0x03	7:0	R/W	rf_f0[7:0]	信道频率 $f_0[7:0]$
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

寄存器地址	位数	R/W	比特名	功能说明
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

5.5 晶体振荡器

晶体振荡器用于为锁相环提供基准时钟, 为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 C_L , 以使晶体准确振荡在参考值范围内。

$$C_L = \frac{1}{\frac{1}{C1} + \frac{1}{C2}} + C_{PCB} + C_{IC}$$

$C1$ 和 $C2$ 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 管脚引入的等效电容 (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。也可以用一个外部信号源连接至 XI 管脚来取代传统的晶体。外部时钟信号的峰峰值在 300mV 到 700mV 之间, 并通过隔直电容耦合到 XI 管脚。晶体的精度越高, 接收性能越好。UM2010 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 $C1$ 和 $C2$ 的容值约 20pF。

芯片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。
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

5.6 AGC

芯片的接收通路集成了 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，芯片接收机的灵敏度、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: Disable 1: Enable

5.7 RSSI

芯片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。

UM2010 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	rssi_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	rssi_Reg_clr	写‘1’清除被锁定的 RSSI
	6	R/W	rssi_sel	收到同步字后锁定 RSSI 值： 0: Disable 1: Enable
	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：

寄存器地址	位数	R/W	比特名	功能说明
				0: Disable 1: Enable
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

5.7.1 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:

$$\text{RSSI} = -(\text{rssi_value}/2)$$

其寄存器优化值如下:

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

UM2010 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。

UM2010 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 $F_{RF}=433.92\text{MHz}$, $DR=10\text{kbps}$, $F_{DEV}=22\text{K}$ 条件下实测的 RSSI 值对应的输入功率。从图上看信号强度在 $[-120\text{dBm}, -35\text{dBm}]$ 区间, RSSI 与信号强度存在线性关系。

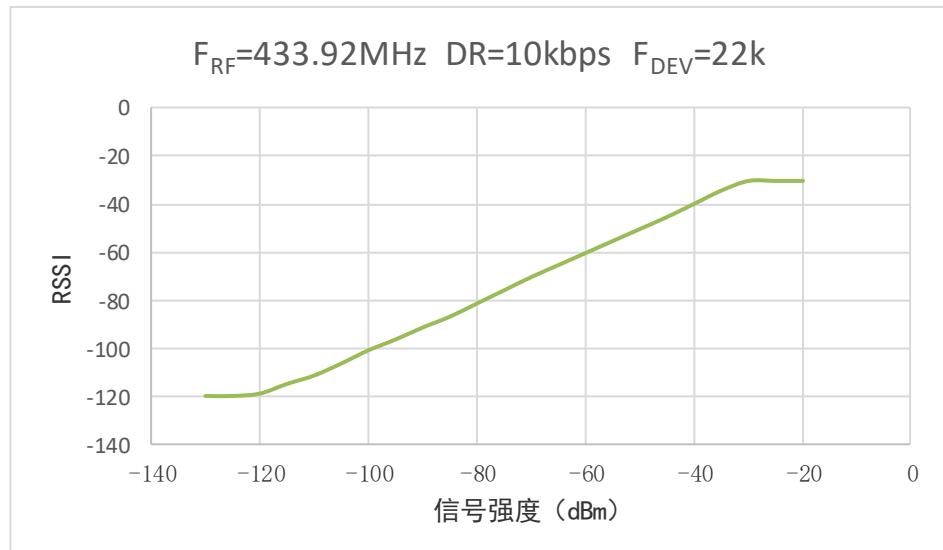


图 5-2: RSSI 线性图

5.7.2 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 对应增益表如下：

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

5.7.3 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

5.8 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: Disable 1: Enable

6 芯片运行

6.1 SPI 接口

芯片内置 SPI 从机模块，通过标准 4 线 SPI 接口与主机 MCU 进行通讯。SPI_CS、SPI_CLK、SPI_MOSI 和 SPI_MISO。SPI 接口可在最大 16MHz 下工作运行。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。芯片内部访问都是以 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，SPI 接口不能写数据，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。

SPI 默认是 4 线的，在上电后可配置成 3 线。在 SPI 3 线模式下，SPI_MOSI 同时用于数据输入和输出，在读寄存器数据时，接口会在地址和数据之间对 SPI_MOSI 的方向进行切换。

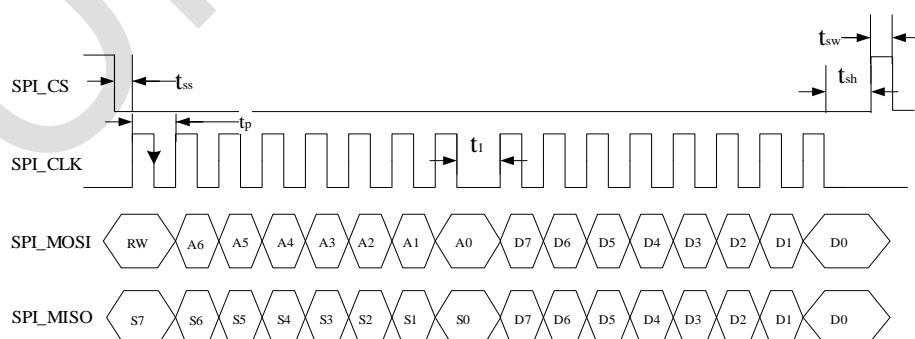


图 6-1: SPI 读写寄存器时序图

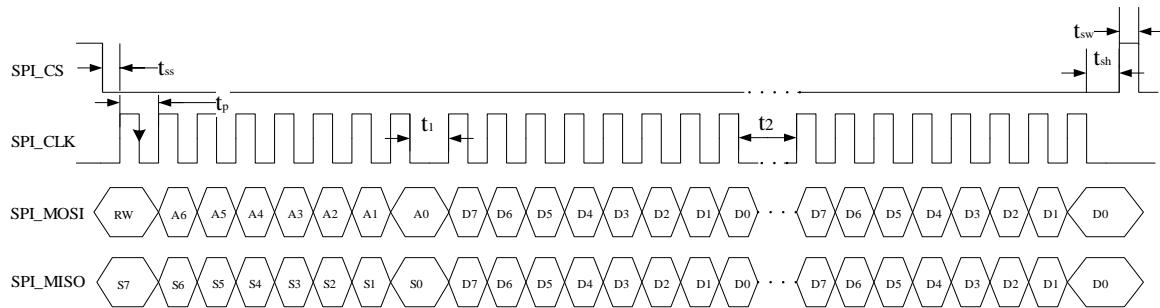


图 6-2: SPI 读写 FIFO 时序图

表 6-1: 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

表 6-2: SPI 相关寄存器

寄存器地址	位数	R/W	比特名	功能说明
0x57	2	spi_3w_sel	R/W	三线 SPI 选择: 0: 4 线标准 SPI 1: 3 线 SPI, 数据线只使用 SPI_MOSI

6.2 FIFO

芯片内置 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 6-3: 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’

6.2.1 FIFO 的工作模式

芯片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 就可合成一个 256-byte 的 FIFO, 在 TX 和 RX 下共同使用。

6.2.2 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。

6.2.3 FIFO 的应用场景

1. 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:

- 初始化寄存器优化值、数据率以及数据包的配置。
- 清 TX FIFO 写指针: Reg53[7:0] = 0x80
- 把数据写入 TX FIFO: Reg55 = data
- 启动发射: Reg60 = 0x40
- 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。
- 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。

2. 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:

- 初始化寄存器优化值、数据率以及数据包的配置。
- 写 Reg60 = 0x20 进入接收, 等 MCU 收到 nIRQ 中断。
- 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。
- 清 RX FIFO 读指针, Reg51 = 0x80。
- 读 RX FIFO 数据 Read Reg(0x52)。
- 读 RSSI (Reg0x64): 根据需要可读出该包的 RSSI 值。
- 写接收命令重新进入接收数据状态。

3. 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:

- 初始化寄存器优化值、数据率以及数据包的配置。
- 清 TX FIFO 写指针: Reg53[7:0] = 0x80
- 把数据写入 TX FIFO: Reg55 = data
- 启动发射: Reg60 = 0x40

- 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。
- 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。
- 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。

4. 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:

- 初始化寄存器优化值、数据率以及数据包的配置。
- 使能 FIFO_flag 中断, 并清 RX FIFO 读指针, Reg51 = 0x80。
- 写 Reg60 = 0x20 进入接收, 需要监听 NIRQ 和 FIFO_flag 中断或者查询 Reg61。
- 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。
- 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。
- 需要读取剩余 FIFO 的数据, 读 RX FIFO 数据 Read Reg(0x52)。
- 读 RSSI (Reg0x64): 根据需要可读出该包的 RSSI 值。
- 写接收命令重新进入接收数据状态。

6.3 GPIO 和中断

UM2010 有 4 个 GPIO, 分别是 nIRQ 和 GPIO[2:0], 每个 GPIO 都可以配置成不同的输入或者输出。芯片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 6-4: 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 IO 口方向: 0: Output 1: Input
	6	R/W	gpio_dir[0]	gpio[0] 方向: 0: Output 1: Input
	5	R/W	gpio_dir[1]	gpio[1] 方向: 0: Output 1: Input
	4	R/W	gpio_dir[2]	gpio[2] 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, 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

寄存器地址	位数	R/W	比特名	功能说明
				Others: 低电平
0x4E	7:4	R/W	gpio_0_sel	gpio[0]输出功能选择, 参考 Reg4D[3:0]描述
	3:0	R/W	gpio_1_sel	gpio[1]输出功能选择, 参考 Reg4D[3:0]描述
0x4F	7:4	R/W	gpio_2_sel	gpio[2]输出功能选择, 参考 Reg4D[3:0]描述

表 6-5: 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
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

6.4 启动时序和状态机

芯片上电稳定或退出关断模式 (SDN 拉低) 后, 至少需要 0.5ms 的时间, POR 才会释放。POR 释放后, 芯片默认进入 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。芯片一共有 6 种工作状态: Shutdown、Standby、IDLE、FSon、TX、RX。

6.4.1 状态机控制图

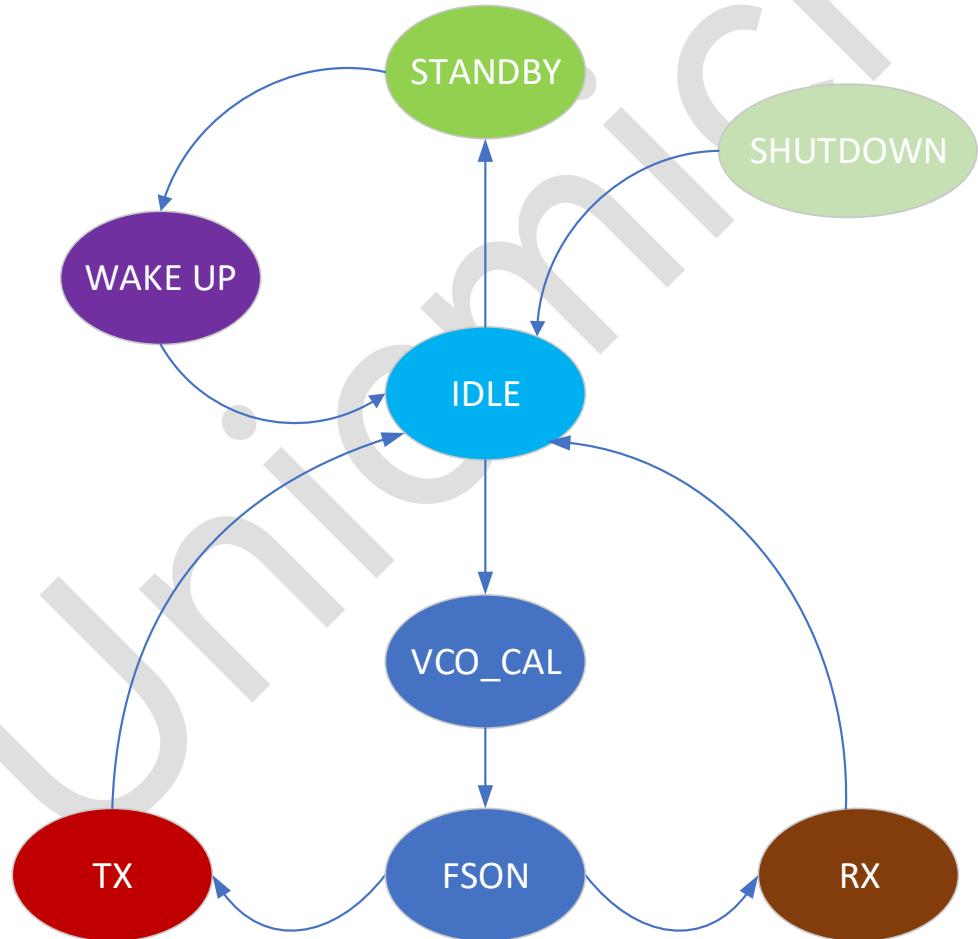


图 6-3: 状态机控制图

6.4.2 工作模式

表 6-6: 工作模式表

状态/模式	描述	命令
SHUTDOWN	芯片处于关断状态	SDN 拉高
IDLE	空闲状态, 复位后进入此状态	上电复位或发送 IDLE 命令
STANDBY	数字模块工作, 其他模块关闭, 寄存器保持	SPI 写 STANDBY 命令
FSON	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

6.4.3 状态机说明

1. 关断状态 (SHUTDOWN)

当 SDN 管脚拉高之后, 芯片进入关闭状态, 芯片内部所有的电路模块都关闭, 消耗电流接近 10nA。

2. IDLE 状态

上电复位或发送 IDLE 命令后, 芯片进入 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。

3. STANDBY 状态

SPI 写命令进入 STANDBY 状态后, 芯片的数字供电开启, 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 SPI_CS 拉低 (2ms) 后, 芯片从 STANDBY 状态退出, 进入 IDLE 状态。

4. 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。

5. 发送状态 (TX)

收到发射数据包命令后, 芯片先打开 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。

6. 接收状态 (RX)

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。当芯片收到数据包后，会给出 nIRQ 中断指示信号然后退回到 IDLE 状态或者 STANDBY 状态 (定时唤醒模式下)。当 AUTO_ACK 功能开启时，芯片收到数据包后会自动发射 ACK 信号然后再退回到 IDLE 状态。

7 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。

7.1 直通模式 (Direct)

数据直通 (direct) 指的外部 MCU 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，让芯片工作在循环状态，此时 TX/RX 的启动或停止完全由 MCU 的命令控制。

表 7-1: 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: Disable 1: Enable
	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 MCU 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 1: Input
	6	R/W	gpio_dir[0]	gpio[0] 方向： 0: Output 1: Input
	5	R/W	gpio_dir[1]	gpio[1] 方向： 0: Output 1: Input

寄存器地址	位数	R/W	比特名	功能说明
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] 1x: gpio[1]

7.1.1 直通模式操作流程

1. 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 GPIO0/1/2。典型的直通模式的 RX 工作顺序如下:

- 1) 寄存器配置 GPIO0/1/2 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。

2. 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。
- 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。

7.2 数据包 (Packet)

数据包 (Packet) 中的 Payload 数据都是从 FIFO 读写, 芯片内部实现了两个 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 7-2: 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	<p>数据包控制模式：</p> <p>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</p> <p>01: 模式 1, 由寄存器控制数据包长度</p> <p>10: 模式 2, TX FIFO 中的第 1 个或 2 个字节作为包长度</p> <p>11: 模式 3, 含有 length、address、seqnum 等多种数据域的模式</p>

7.2.1 数据包模式

数据包支持 4 种数据包模式：Mode0~Mode3

7.2.1.1 MODE 0

MODE 0 为 FIFO 循环模式。在此模式下启动 TX/RX，如果没有收到 MCU 的停止命令，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，并且 RX 端会检测 Preamble 和 Syncword。对于 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。



图 7-1: Mode 0 帧格式

7.2.1.2 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。

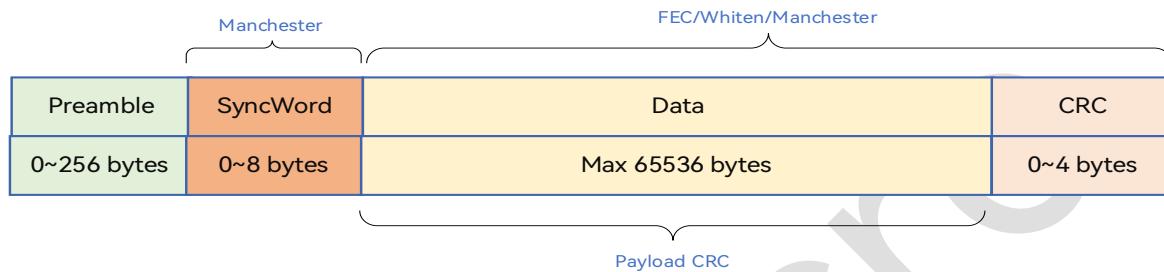


图 7-2: Mode 1 帧格式

7.2.1.3 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。

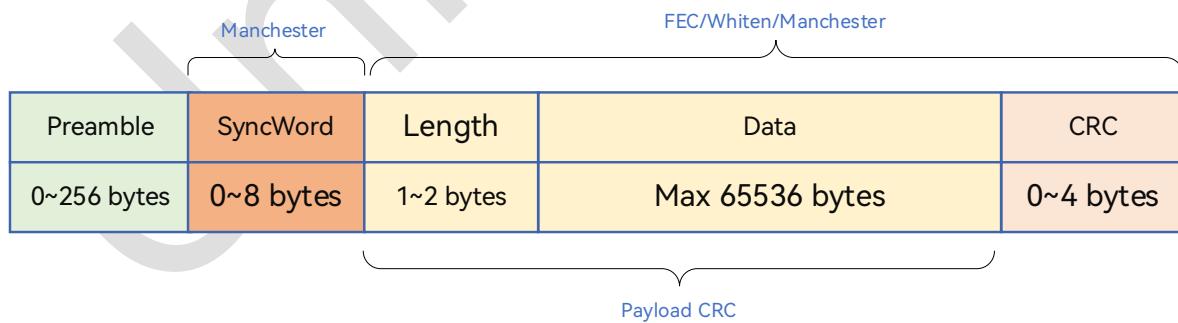


图 7-3: Mode 2 帧格式

7.2.1.4 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。

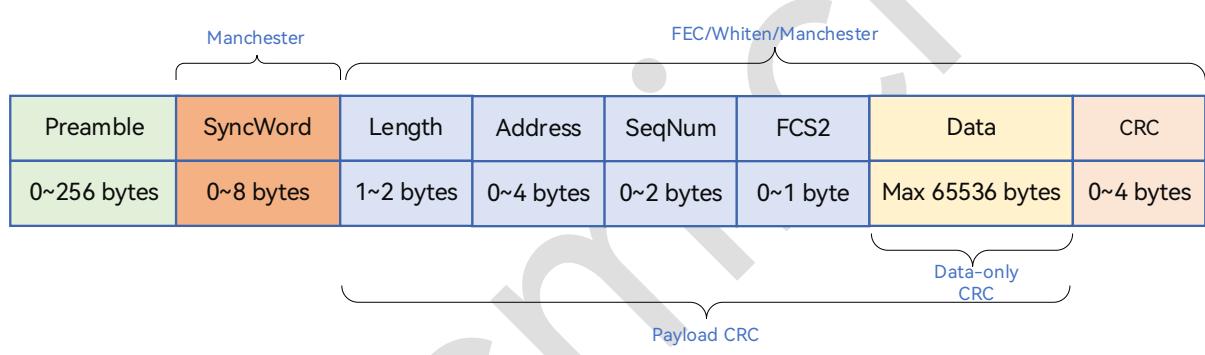


图 7-4: Mode 3 帧格式

7.2.2 数据包操作流程

1. 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：

- 初始化寄存器。
- 配置对应的接收模式。
- 发送RX命令进入接收状态。
- 根据相关的中断状态读取RX FIFO。

- 自动进入IDLE状态以节省功耗。
- 清除接收中断状态。

2. 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：

- 初始化寄存器。
- 发送IDLE状态命令，并将数据填入TX FIFO。
- 寄存器设置发送中断。
- 配置TX命令进入发送状态。
- 在相应的中断状态中将数据写入TX FIFO。
- 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。

7.2.3 数据包字段

7.2.3.1 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 7-3: 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: Disable 1: Enable
0x10	2	R/W	preamble_int_en	接收 Preamble(前导码)匹配中断使能： 0: Disable 1: Enable

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
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

7.2.3.2 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, UM2010 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。UM2010 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 7-4: 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: Disable 1: Enable
0x10	1	R/W	syncword_int_en	接收 Syncword (同步字) 匹配中断使能: 0: Disable 1: Enable
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: Disable 1: Enable
	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},

寄存器地址	位数	R/W	比特名	功能说明
				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, 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

7.2.3.3 Payload

Payload 是数据包通信中真正承载用户信息的核心部分, UM2010 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 7-5: 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: Disable 1: Enable
	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

寄存器地址	位数	R/W	比特名	功能说明
	4	R/W	scramble_type	1: MSB 伪随机数产生器类型: 0: 抽出模式 1: 插入模式
				反馈抽头选择
	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

1. 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。

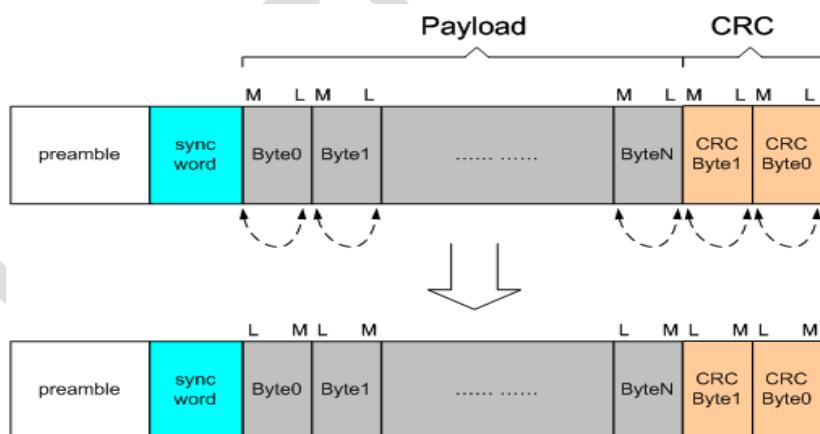


图 7-5: PAYLOAD_BIT_ORDER 操作

2. 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

如下图, 移位顺序不变, 从高位移位到低位:

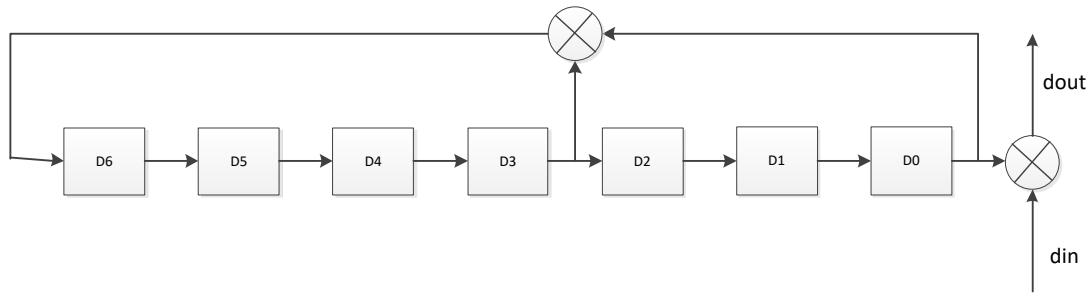


图 7-6: PN7 抽出模式移位顺序

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

如下图, 移位顺序不变, 从高位移位到低位:

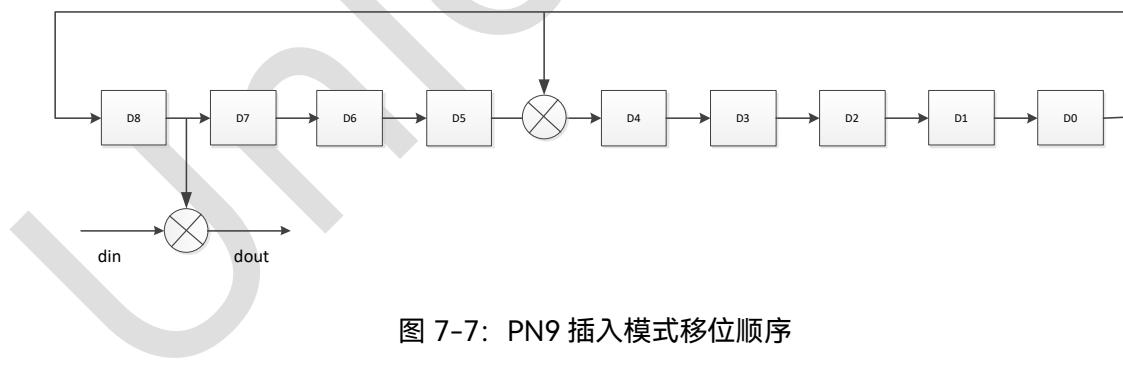


图 7-7: PN9 插入模式移位顺序

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。

7.2.3.4 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 7-6: 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: Disable 1: Enable
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。

1. 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 7-8: 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。

2. 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。

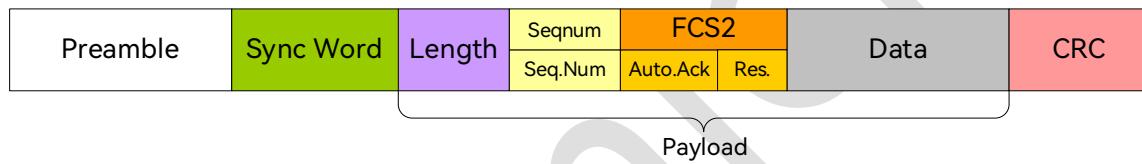


图 7-9: 可变长度包格式, Address 不存在

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。

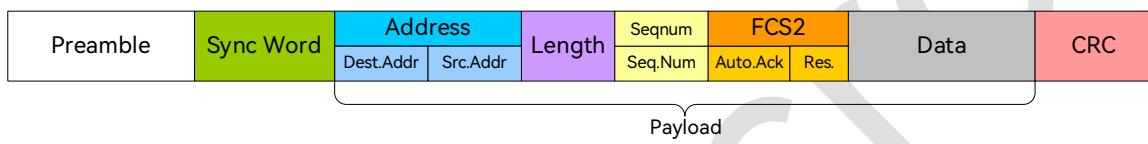


图 7-10: Address 存在, Address 在 Length Byte 之前

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 $2(\text{Dest.Addr}) + 1 = 3$, 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 之后, 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。

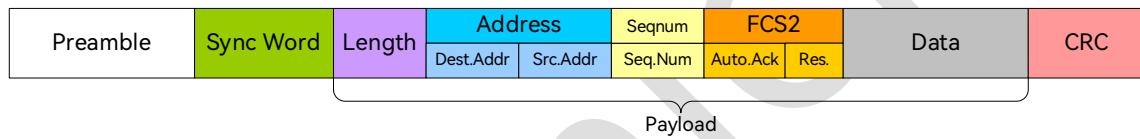


图 7-11: Address 存在, Address 在 Length Byte 之后

当 Address 存在, 且 Addr_pos_sel = 0 (Address 在 Length Byte 之后) 时, 对于 TX 来说, Payload 长度: $1 + \text{payload_length}_{<15:0>}$, 其中 $\text{payload_length}_{<15:0>}$ 的内容就等于 Length Byte 的内容, 指的是 Payload 的长度。如果 length_size = 0, 即 $\text{payload_length}_{<7:0>}$ 有效, 最大值为 255, Length Byte 本身是 1 个 byte 的长度。如果 length_size = 1, 即 $\text{payload_length}_{<15:0>}$ 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 Payload 的长度。示例如下:

- $\text{Payload_length}_{<15:0>} = 17$
- $\text{Addr_en} = 1$
- $\text{Addr_size}_{<1:0>} = 2$
- $\text{Length_size} = 0$
- $\text{Seq_en} = 1$
- $\text{Seqnum_size} = 1$
- $\text{FCS2_EN} = 1$

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 $2(\text{Addr}) + 1 = 3$ ， 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。

7.2.3.5 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 7-7: 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： 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: Disable 1: Enable
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

寄存器地址	位数	R/W	比特名	功能说明
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

7.2.3.6 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 7-8: 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: Disable 1: Enable
	3	R/W	m3_seqnum_bit_size	数据包模式 3 时, seqnum 数据长度: 0: 8bits 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

7.2.3.7 FCS2

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

表 7-9: 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: Disable 1: Enable
	6	R/W	ack_payload_en	发射 ACK 时带 payload 数据返回: 0: Disable 1: Enable
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: Disable 1: Enable
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

1. 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:

Preamble	Sync Word	Address		Seqnum
		Rx.Addr [15:0]	Rx.Addr [31:16]	Rx_Seq.Num

图 7-12: 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。

7.2.3.8 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, UM2010 的 CRC 域支持独立使能, 灵活配置, 支持常用的标准多项式。

表 7-10: 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: Disable 1: Enable
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: 0: LSB 1: MSB
	3	R/W	crc_inv	CRC bit 取反: 0: Disable

寄存器地址	位数	R/W	比特名	功能说明
2	2	R/W	crc_man_en	1: Enable CRC 曼彻斯特编码使能: 0: Disable 1: Enable
				CRC 长度: 00: 8 位 01: 16 位 10: 24 位 11: 32 位
	1:0	R/W	crc_len	
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。

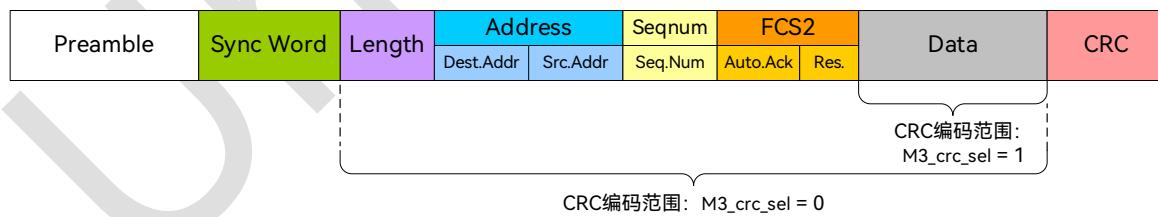


图 7-13: CRC 编码范围

- **crc_inv (Reg94[3])**

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

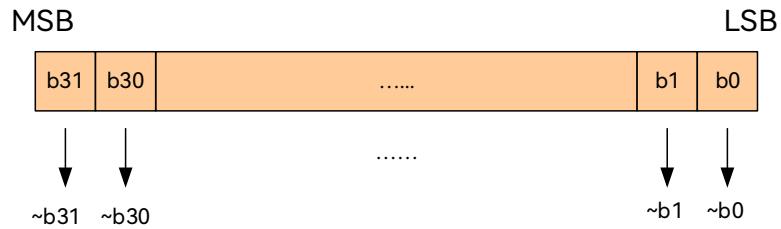


图 7-14: CRC_INV

- **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 7-15: CRC_BIT_ORDER

2. 常用 CRC 标准和多项式

表 7-11: 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	$x^{32}+x^{26}+x^{23}+\dots+x^2+x+1$	0x04C11DB7	ZIP, RAR, IEEE 802 LAN /FDDI, IEEE1394, PPP-FCS
CRC-32c	$x^{32}+x^{28}+x^{27}+\dots+x^8+x^6+1$	0x1EDC6F41	SCTP

3. 配置 CRC8 和 CRC16 示例

- CRC8 的 X8+X2+X1+1 配置示例（简记式为 0x07）

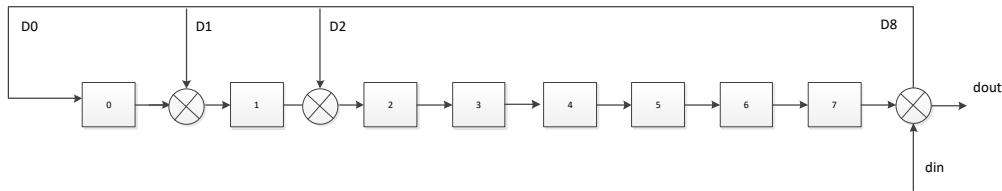


图 7-16: CRC8 的配置示例

如上图所示，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 X16+X12+X5+1 的配置示例（简记式为 0x1021）

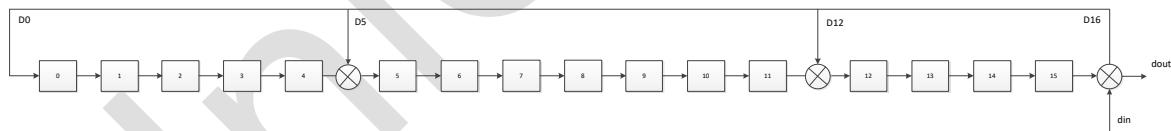


图 7-17: CRC16 的配置示例

如上图所示，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

8 典型应用

8.1 参考原理图

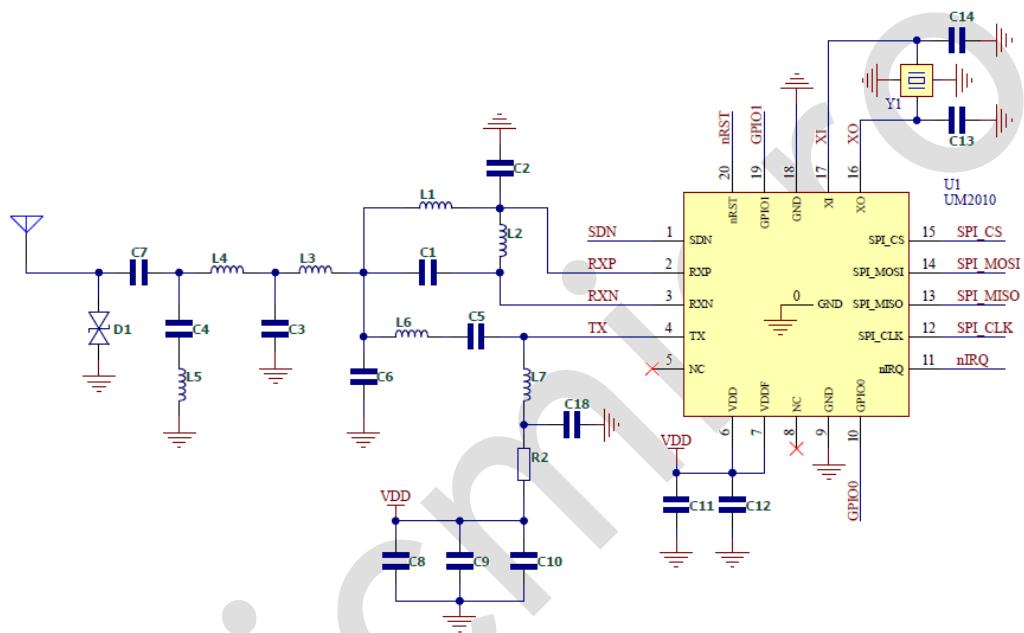


图 8-1: 应用参考原理图

应用说明:

1. SDN 为芯片关断引脚, 高电平为关断。如果该引脚受 MCU 控制, 必须给固定电平。如果应用中不需要关断功能, 可以将 SDN 引脚接地。
2. nRST 为芯片的 GPIO2 引脚, 默认配置为复位功能, 使能上拉功能, 高电平时芯片正常工作。如果应用不需要复位功能, 可悬空。
3. 芯片关断时, SDN、GPIO0、GPIO1、nRST(GPIO2)、SPI_CS、SPI_CLK、SPI_MISO、SPI_MOSI 处于高阻状态。
4. 芯片睡眠时, SDN、SPI_CS、SPI_CLK、SPI_MOSI 为输入。SPI_MISO 为输出, 输出低电平。GPIO0、GPIO1、GPIO2 方向保持, 如果为输出, 则输出低电平。
5. D1 为 ESD 保护管, 可根据应用要求来定是否需要。

8.2 +13dBm 匹配参数

表 8-1: +13dBm 匹配参考参数表

位号	描述	元件值				单位
		315	433	868	915	
C1	±5%, 0402, NPO, 50V, YAGEO	5.6	3	2	2	pF
C2	±5%, 0402, NPO, 50V, YAGEO	4.7	8.2	2	2	pF
C3	±10%, 0402, NPO, 50V, YAGEO	8.2	8.2	5	5	pF
C4	±5%, 0402, NPO, 50V, YAGEO	5	2.7	2.7	2.7	pF
C5	±10%, 0402, NPO, 50V, YAGEO	8.2	5.6	4.7	4.7	pF
C6	±5%, 0402, NPO, 50V, YAGEO	3.3	3	3	3	pF
C7	±5%, 0402, NPO, 50V, YAGEO	0R	20	5.6nH	5.6nH	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, X5R, 10V, muRata	2.2	2.2	2.2	2.2	μ F
C11	±10%, 0402, X7R, 50V, YAGEO	100	100	100	100	nF
C12	±20%, 0603, X5R, 10V, SUMSUNG	10	10	10	10	μ F
C13	±1%, 0402, NPO, 50 V, YAGEO	27	20	27	27	pF
C14	±1%, 0402, NPO, 50 V, YAGEO	27	20	27	27	pF
C18	±10%, 0402, X7R, 50V, YAGEO	100	100	100	100	nF
L1	±5%, 0402, 贴片高频电感, Sunlord	68	33	12	12	nH
L2	±5%, 0402, 贴片高频电感, Sunlord	82	33	12	12	nH
L3	±5%, 0603, 贴片高频电感, Sunlord	68	39	8.2	8.2	nH
L4	±5%, 0603, 贴片高频电感, Sunlord	47	33	10	10	nH
L5	±5%, 0402, 贴片高频电感, Sunlord	12	10	2.7	2.7	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
Y1	3225 无源晶振, ±10ppm, 20pF 负载	26	26	26	26	MHz

8.3 +18dBm 匹配参数

表 8-2: 天线匹配参考参数表

位号	描述	元件值				单位
		315	433	868	915	
C1	±5%, 0402, NPO, 50V, YAGEO	5.6	3	2	2	pF
C2	±5%, 0402, NPO, 50V, YAGEO	4.7	8.2	2	2	pF
C3	±10%, 0402, NPO, 50V, YAGEO	8.2	8.2	5	5	pF
C4	±5%, 0402, NPO, 50V, YAGEO	5	2.7	2.7	2.7	pF
C5	±10%, 0402, NPO, 50V, YAGEO	8.2	5.6	3.9	3.9	pF
C6	±5%, 0402, NPO, 50V, YAGEO	3.3	3	5	5	pF
C7	±5%, 0402, NPO, 50V, YAGEO	27nH	20	5.6nH	5.6nH	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, X5R, 10V, muRata	2.2	2.2	2.2	2.2	μ F
C11	±10%, 0402, X7R, 50V, YAGEO	100	100	100	100	nF
C12	±20%, 0603, X5R, 10V, SUMSUNG	10	10	10	10	μ F
C13	±1%, 0402, NPO, 50 V, YAGEO	27	20	27	27	pF
C14	±1%, 0402, NPO, 50 V, AGEO	27	20	27	27	pF
C18	±10%, 0402, X7R, 50V, YAGEO	100	100	100	100	nF
L1	±5%, 0402, 贴片高频电感, Sunlord	68	33	12	12	nH
L2	±5%, 0402, 贴片高频电感, Sunlord	82	33	12	12	nH
L3	±5%, 0603, 贴片高频电感, Sunlord	22	22	8.2	8.2	nH
L4	±5%, 0603, 贴片高频电感, Sunlord	47	33	10	10	nH
L5	±5%, 0402, 贴片高频电感, Sunlord	12	10	2.7	2.7	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
Y1	3225 无源晶振, ±10ppm, 20pF 负载	26	30	26	26	MHz

9 封装尺寸

9.1 QFN20 (4*4mm)

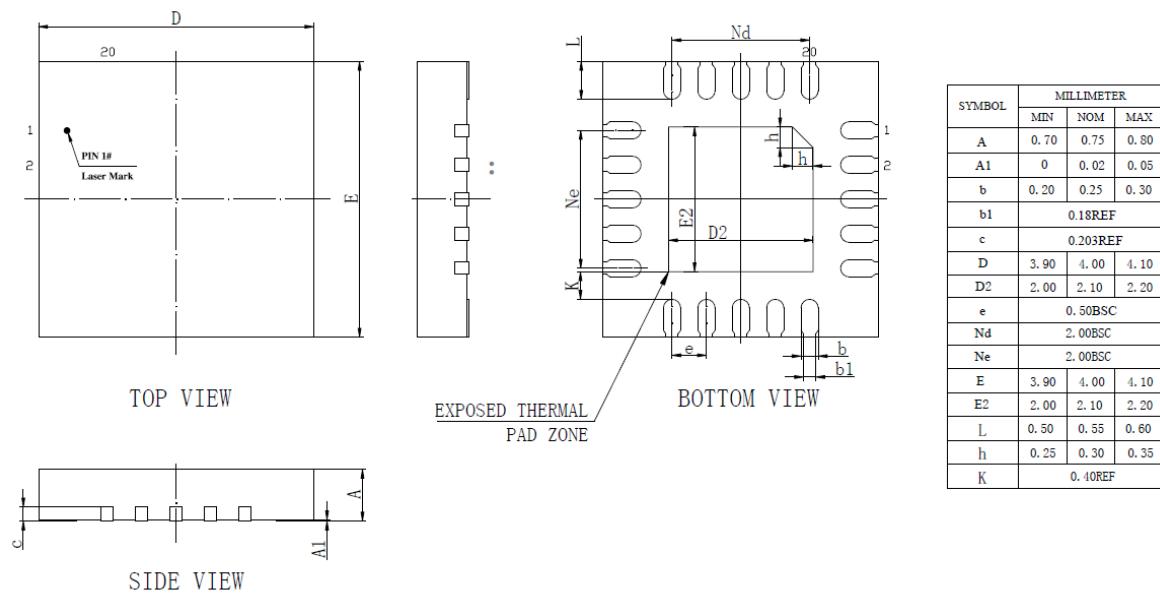


图 9-1: QFN20 封装图

10 版本维护

版本	日期	描述
V1.0	2021.12.15	初始版
V1.1	2022.03.18	新增并更新寄存器列表。
V1.2	2022.04.22	<ol style="list-style-type: none">1. 更新寄存器定义章节。2. 更新功能框图。3. 新增 RSSI 线性图。4. 更新电气参数。5. 更新 QFN20 封装尺寸图。6. 新增 RSSI、BUCK DCDC、GPIO、中断及 FIFO 等相关描述的章节。
V1.3	2023.06.20	<ol style="list-style-type: none">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. 调整功能描述、芯片运行、数据处理机制内容及排版。
V1.4	2025.12.21	<ol style="list-style-type: none">1. 删除 BUCK DCDC 模式。2. 优化功能描述、芯片运行描述和匹配参数。3. 调整数据包及包处理机制的描述及格式。