编 号	HR_SJ_QRSW2
密 级	内部公开
阶 段	发布
页 数	22

代 号

名 称 HR-RTLS1 嵌入式软件开发手册

基于 DW1000/DW3000 的高精度 UWB 定位开发套件

会 签

大连浩如科技有限公司

文档控制

变更记录

版本号	日期	增加/修改/删除	简单描述	嵌入式版本号
V1.0	20220130	创建	根据模板创建，描述嵌入式软件方案和流程。	V63
V1.1	20231010	修改	修改基站和标签间通信协议，新增字段	V66
V1.2	20240327	修改	修改POLL帧结构	V70

目 录

1 系统概况

本嵌入式开发手册适用于 HR-RTLS1 系列 UWB 定位模块，基于 DecaWave 公司 DW1000 与 DW3000 系列（以下统称为 DWIC）的 UWB 定位系统；HR-RTLS1 相关模块型号如下表，模块整体软件架构相同，差异化代码通过宏定义预编译适配，模块间可兼容搭配使用，代码维护简单并方便移植。

表 3-1 HR-RTLS1 模块型号

序号	型号	主要特点
1	ULM1	官方 DWM1000 模组，显示器，50 米
2	LD150	外置全向天线，外壳，内置电池，150 米
3	LD150-I	LD150+IMU
4	LD600	外置全向天线，外壳，内置电池，内置 PA，600 米
5	LD600-I	LD600+IMU
6	ULM1-SH	手环外壳，内置电池，运动检测，内置 PA，400 米
7	ULM1-GP	工牌外壳，内置电池，运动检测，内置 PA，400 米
8	ULM3	官方 DWM3000 模组，显示器，40 米
9	ULM3-SH	手环外壳，内置电池，运动检测，40 米
10	ULM3-PDOA	PDOA 基站，测角度，单基站定位，跟随小车，40 米

本嵌入式软件基于 DecaWave 官方 API 开发高效率的多基站多标签 TWR 测距应用，并通过串口输出测距或定位结果；官方 API 在技术资料包内获取。

ULM1-SH、ULM1-GP、ULM3-SH 模块采用 STM32L151CUB6 低功耗单片机；其他模块均采用 STM32F103CBT6 单片机或其完全兼容替代的国产 HK 或 GD 型号，使用 CUBEmx 初始化配置，HAL 库函数开发，可迅速方便移植到其他型号单片机。

阅读本手册前，建议先观看开箱测试视频和阅读用户手册，先大概了解系统功能和使用方法；本手册可结合《嵌入式开发与代码讲解》视频教程同步学习。

2 嵌入式软件整体架构

嵌入式软件整体架构如图 2-1 所示：主要分为驱动层、DW_API 层、应用层。

驱动层主要实现 STM32 与 DWIC 的 SPI 通信，一般使用 CUBEmx 进行初始化配置，使用 HAL 库进行开发，初始化完成后，自动生成初始化代码，使用 SPI 数据收发函数，对接到 DW_API 层，完成驱动层搭建。另外还有中断配置、IIC 配置、串口配置、看门狗配置等都在 CUBEmx 进行初始化配置，驱动层代码用户开发工作量较小，主要熟练掌握 CUBEmx 配置即可。

DW_API 层使用 DecaWave 官方 API 进行移植搭建，API 将常用功能进行函数封装，DWIC 的主要功能实现基本通过读写相应的寄存器实现，通过官方 API 内的简单 example 结合《DW1000_Software_API_Guide.pdf》可大概了解常用 API 功能。DW_API 层代码用户开发工作量较小，开发相应应用层功能时会调用即可。主要程序目录为 Src\decadriver，Src\platform。

应用层是实现 TWR 测距主要功能的实现代码，完成系统的状态读取，参数配置，TWR 数据收发，TOF 计算，卡尔曼滤波，串口数据发送，OLED 显示等功能。应用层全部由浩如科技开发，也是实现整体功能的核心代码，需重点学习和熟练掌握。二次开发基本也是基于应用层来开发。主要程序目录为 Src\application，Src\kalmanfilter，Src\OLED。

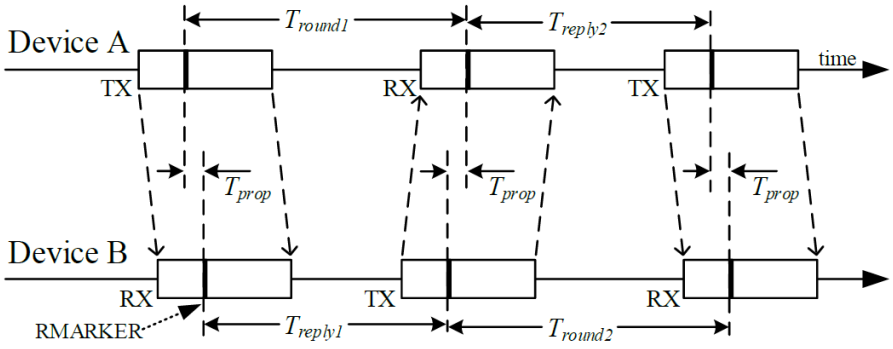
应用层	主要文件 dw_main.c instance_anchor.c instance_tag.c	主要功能 实现多基站多标签TWR通信全过程及TOF计算、OLED显示、卡尔曼滤波、串口数据输出、三边定位算法等
DW_API层	主要文件 deca_device.c port.c deca_spi.c	主要功能 使用官方API库，通过SPI控制寄存器读写完成参数配置、数据收发等功能，功能封装成相应函数，供用户调用使用
驱动层	主要文件 STM32F1xx_HAL_Driver spi.c i2c.c	主要功能 完成SPI、IIC、GPIO、看门狗、时钟、外部中断等相应配置，使用CUBEmx生成初始化代码

图 2-1 嵌入式软件整体架构

3 TWR 方法与算法

3.1 TWR 方法

如图 3-1 所示为 2 设备、3 消息 DS-TWR 过程和飞行时间 Tprop 推导公式。



假设 3 条消息分别为 M1、M2、M3，Device A 测得 3 个时间戳：M1_tx、M2_rx、M3_tx，Device B 测得 3 个时间戳：M1_rx、M2_tx、M3_rx，Device A 测得的 3 个时间戳随 M3 消息发给 Device B，Device B 最终获得 6 个时间戳，通过差值计算 4 个时间长度，最终计算 TOF。

由图示可得：

$$T_{prop} = \frac{1}{2}(T_{round1} - T_{reply1}) \qquad T_{prop} = \frac{1}{2}(T_{round2} - T_{reply2})$$

所以：

$$\begin{aligned} T_{round1} \times T_{round2} &= (2T_{prop} + T_{reply1})(2T_{prop} + T_{reply2}) \\ &= 4T_{prop}^2 + 2T_{prop}(T_{reply1} + T_{reply2}) + T_{reply1}T_{reply2} \end{aligned}$$

即：

$$\begin{aligned} T_{round1} \times T_{round2} - T_{reply1}T_{reply2} &= 4T_{prop}^2 + 2T_{prop}(T_{reply1} + T_{reply2}) \\ &= T_{prop}(4T_{prop} + 2T_{reply1} + 2T_{reply2}) \\ &= T_{prop}(T_{round1} + T_{round2} + T_{reply1} + T_{reply2}) \end{aligned}$$

于是得到如下 Tprop 计算公式：

$$\hat{T}_{prop} = \frac{(T_{round1} \times T_{round2} - T_{reply1} \times T_{reply2})}{(T_{round1} + T_{round2} + T_{reply1} + T_{reply2})}$$

计算得出飞行时间 Tprop（TOF）后，乘以光速常量，得到距离值结果，完成整个 TWR 测距过程。

3.2 多基站高效型 TWR 方法

如 3.1 所述，1 标签对 1 基站需 3 条消息（标签发送 2 条，基站发送 1 条）

完成 DS-TWR 过程，因定位时系统内一般存在 4 基站同时测距，则按此方法需要 12 条消息（标签发送 8 条，基站发送 4 条）。

下述方法提出多基站高效型 TWR 方法，1 标签 4 基站时，仅需 6 条消息（标签发送 2 条，基站发送 4 条）完成，不但节约了时间，也节省了标签功耗。

为了更好的区分消息，将标签发送的第一条消息定位为 poll 消息，4 个基站回复的消息定义为 resp 消息，标签最后发送的消息定义为 final 消息。

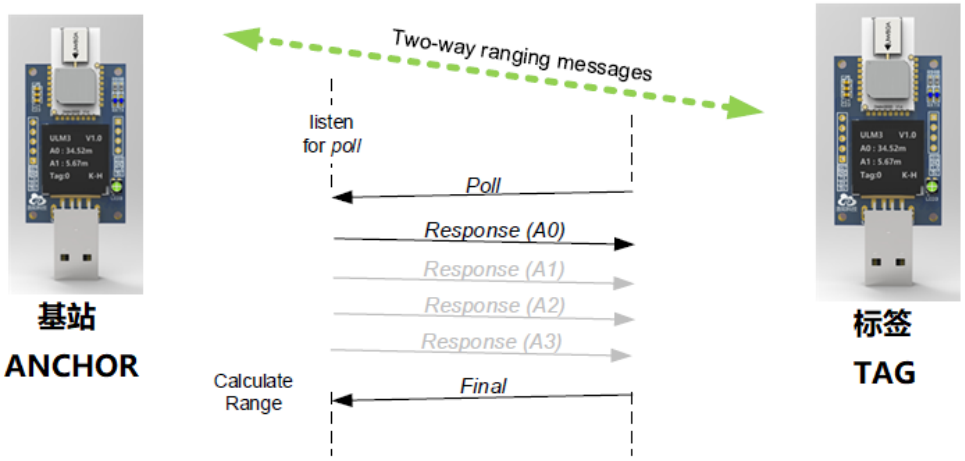
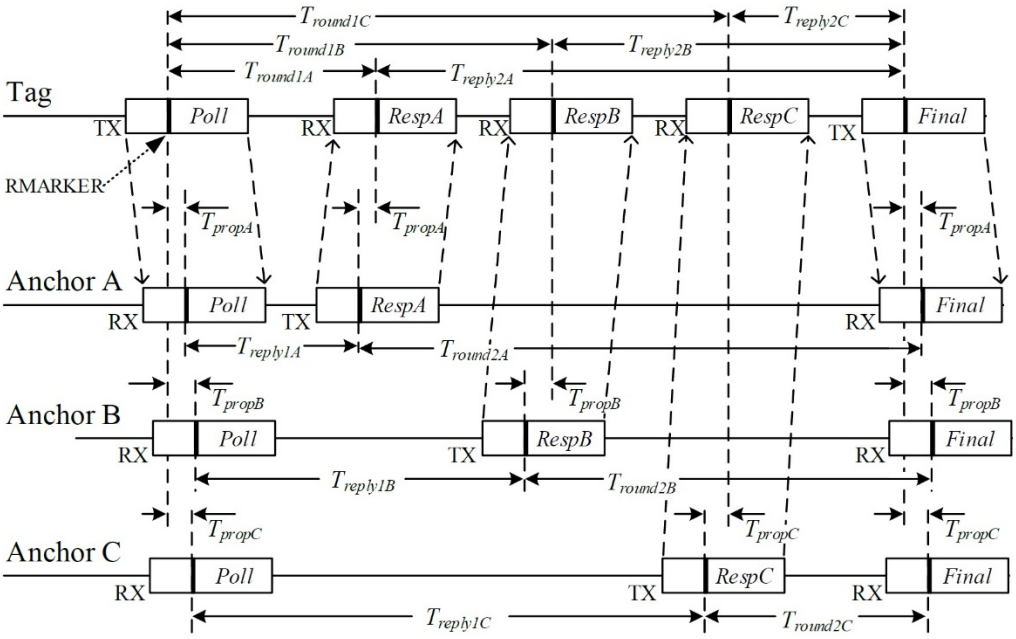


图 3-1 多基站高效型 TWR 方法示意

3.3 多基站高效型 TWR 算法

如图 3-3 所示使用标签发送一条 poll 消息，4 个基站依次回复 resp 消息，标签发送 final 消息为一个测距周期的方法进行测距，该算法相比较传统的一次 poll、resp、final 进行一次测距的传统 TWR 算法的优势是可以减少标签发送消息的数量，降低标签功耗，但对基站和标签的工作时序要求比较严格，是开发难点。



$$T_{propA} = \frac{T_{round1A} \times T_{round2A} - T_{reply1A} \times T_{reply2A}}{T_{round1A} + T_{round2A} + T_{reply1A} + T_{reply2A}}$$

图 3-2 测距算法过程图

4 TWR 通信协议

4.1 通信数据帧基本结构

通信数据遵循 IEEE 802.15.4 协议 MAC 层帧格式，如表 4-1 所示，数据帧包含帧头 MAC Header (MHR)、负载 MAC Payload 和帧尾 MAC Footer (MFR)三部分。帧头部分由帧控制字节、帧序列号和地址等信息构成；负载部分长度可变，可用户自定义；帧尾部分是帧头和负载数据的 16 位 CRC（FCS）校验序列，一般由 DWIC 自动生成。

表 4-1 通信数据帧格式

2 字节 0-1	1 字节 2	2 字节 3-4	2 字节 5-6	2 字节 7-8	可变字节 9+	2 字节 +
Frame Control (FC)	Sequence Number	PAN ID	Destination Address	Source Address	Ranging Message	FCS
MHR 帧头					MAC 负载	MFR 帧尾

4.1.1 Frame Control

表 4-2 Frame Control 格式

Frame Control (FC)															
Bit0	Bit1	Bit2	Bit3	Bit4	Bit5	Bit6	Bit7	Bit8	Bi9	Bit10	Bit11	Bit12	Bit13	Bit14	Bit15
1	0	0	0	0	0	1	0	0	0	0	1	0	0	0	1
Frame Type			SEC	PEND	ACK	PIC	Reserved			DestAddrMode		Frame Version		SrcAddrMode	

Frame Control 是 16bit 寄存器，其中 Frame Type 设置为 001，表示 Data 数据，PIC 设置为 1 表示 PAN ID 开启，DestAddrMode 设置为 10，表示源地址（本机地址）采用 16bit 设备短地址，SrcAddrMode 表示目标地址（接收方地址）采用 16bit 设备短地址。TWR 通信中，Frame Control 设置为 0x8841。

表 4-3 Frame Type 含义

Frame Type Field (FC bits 2 to 0)	Frame
0, 0, 0	Beacon
0, 0, 1	Data
0, 1, 0	Acknowledgement
0, 1, 1	MAC command
1, 0, 0	Reserved
1, 0, 1	Reserved
1, 1, 0	Reserved
1, 1, 1	Reserved

表 4-4 DestAddrMode 含义

Destination addressing mode (FC bits 11 & 10)	Meaning
0, 0	No destination address or destination PAN ID is present in the frame
0, 1	Reserved
1, 0	The destination address field is a short (16-bit) address.
1, 1	The destination address field is an extended (64-bit) address.

表 4-5 SrcAddrMode 含义

Source addressing mode (FC bits 15 & 14)	Meaning
0, 0	No source address or source PAN ID is present in the frame
0, 1	Reserved
1, 0	The source address field is a short (16-bit) address.
1, 1	The source address field is an extended (64-bit) address.

4.1.2 Sequence Number

帧序列号，每帧数据自增 1。

4.1.3 PAN ID

网络 ID，收发数据的设备需设置同一个 PAN ID 下才能正常收发，否则将产生帧拒绝中断。固定为 0xDECA。

4.1.4 Destination Address

目标设备地址，在帧过滤模式下，接收方设备地址和发送方目标地址相同时，产生接收正确中断，否则产生帧拒绝中断。

4.1.5 Source Address

本机设备地址。

4.1.6 FCS

Frame Check Sequence，简称（FCS），数据校验，由 DWIC 自动计算完成。

4.1.7 Ranging Message

4.1.7.1 Poll Message

1字节 9	1字节 10	1字节 11	1字节 12	1字节 13	12字节 14-25
Function Code	Range Number	SOS 标签报警上传	BATTERY 电池电量	ALARM 执行下发报警状态	USER 自定义数据
0x21	0-255	1/0	0-100%	1/0	-

4.1.7.2 Response Message

1字节 9	1字节 10	2字节 11-12	4字节 13-16	1字节 17	1字节 18	1字节 19
Function Code	Range Number	Sleep Correction	Distance(n-1) 单位mm	ALARM 下发报警	GROUP_ID 基站组编号	SLOT 系统SLOT容量
0x10	0-255	-	-	1/0	0-128	0-255

4.1.7.3 Final Message

1字节 9	1字节 10	1字节 11	4字节 12-15	4字节 16-19	1字节 20	4字节 21-24	1字节 25	4字节 26-29
Function Code	Range Number	Valid Resp	Poll TX time	Final TX time	A0 Group ID	A0 Resp RX time	A1 Group ID	A1 Resp RX time
0x23	0-255	-	-	-	0-128	-	0-128	-

1字节 30	4字节 31-34	1字节 35	4字节 36-39	1字节 40	4字节 41-44	1字节 45	4字节 46-49
A2 Group ID	A2 Resp RX time	A3 Group ID	A3 Resp RX time	A4 Group ID	A4 Resp RX time	A5 Group ID	A5 Resp RX time
0-128	-	0-128	-	0-128	-	0-128	-

1字节 50	4字节 51-54	1字节 55	4字节 56-59
A6 Group ID	A6 Resp RX time	A7 Group ID	A7 Resp RX time
0-128	-	0-128	-

5 单周期 TWR 时序

单周期 TWR 时序指的是 1 个标签与多个基站（本手册均以 4 个为例）完成多基站高效 TWR 测距方法的过程，主要包括标签发送 poll 消息，4 个基站依次发送 resp 消息，标签发送 final 消息的全过程，并描述在各个过程中设备应处于

的状态。

TAG	SEND POLL 广播	RECV RESP1	RECV RESP2	RECV RESP3	RECV RESP4	SEND FINAL 广播
ANCHOR(0)	RECV POLL	SEND RESP1	RECV RESP2	RECV RESP3	RECV RESP4	RECV FINAL
ANCHOR(1)	RECV POLL	RECV RESP1	SEND RESP2	RECV RESP3	RECV RESP4	RECV FINAL
ANCHOR(2)	RECV POLL	RECV RESP1	RECV RESP2	SEND RESP3	RECV RESP4	RECV FINAL
ANCHOR(3)	RECV POLL	RECV RESP1	RECV RESP2	RECV RESP3	SEND RESP4	RECV FINAL

图 5-1 单周期 TWR 时序图

如图 5-1 所示，标签首先广播发起 poll 消息，基站 0-3 一起接收 poll 消息后，按自身 ID 号进行顺序回复 resp，其中基站 0 第一个回复，基站 1 第二个回复，依次类推；在某基站回复 resp 时，其他基站关闭帧过滤模式，也监听该基站的 resp 消息，从而得到 resp 消息内的 distance(n-1)数据；标签接收完成 resp 后，广播 final 消息给所有基站，进行 TOF 计算。

因 TOF 在基站内计算，且每个基站计算自己到标签的 TOF，常规的方案是把 4 个基站通过以太网连接到上位机进行 4 个测距值的接收并进行坐标解算。HR-RTLS1 嵌入式软件由于各个基站的 resp 消息内包含了 distance(n-1)数据，并被其他基站监听到，单周期结束后，所有基站和标签均获得了 4 个基站的 distance(n-1)数据。此时，只需通过串口将任意一个基站连接上位机或者标签连接上位机即可发送 4 个基站的测距值，从而进行定位解算，大大减小了系统部署的复杂度。

6 标签时序管理

在大多数应用中，系统内可能存在不止 1 个标签，多个标签如不进行时序管理则极易发生冲突，导致某个标签测距一直失败；简单的时序管理是“轮询调度”法，即主基站按时序轮询若干个标签，轮询到哪个标签则该标签工作，以避免冲突，类似 RS485 主从结构；该方法虽简单，但是所有标签需时刻处于接收状态，等待主机站轮询，而标签处于接收态时功耗很大，一般标签是电池供电，对待机时间非常不利，所以使用 UWB 技术业内主流的的标签时序管理一般不采用此方法。

HR-RTLS1 系统内默认由 A0 基站进行标签时序管理，A0 基站按系统内标签最大容量分配若干个 SLOT，SLOT 意为“时间槽”，每个 SLOT 内预留一定的保护时间（guard time），用于接收其他标签的测距请求，A0 基站监控每个标签发起测距请求的时间，并通过 resp 消息内的 Sleep Correction 将标签按 ID 分配到相应 SLOT 内，使标签有序工作，如图 6-1 所示。

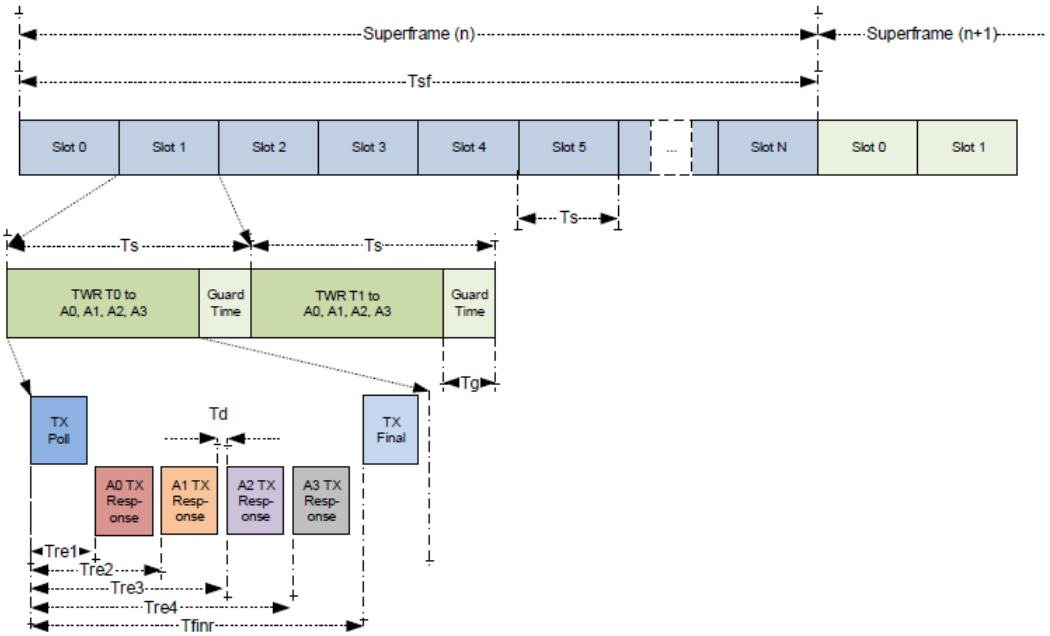


图 6-1 SLOT 分配

7 程序流程

7.1 主程序工作流程

主程序位于 Src/application/dw_main.c，主要完成设备参数初始化和按拨码开关状态进行基站或标签状态机运行，最后进行串口数据打包发送。HR-RTLS1 嵌入式软件将基站和标签维护在同一套工程内，通过上电读取拨码开关来判断执行相应的角色，大大减小了系统维护的复杂度，仅需一套程序可以适配所有的基站和标签模块。主程序工作流程图如图 7-1 所示。

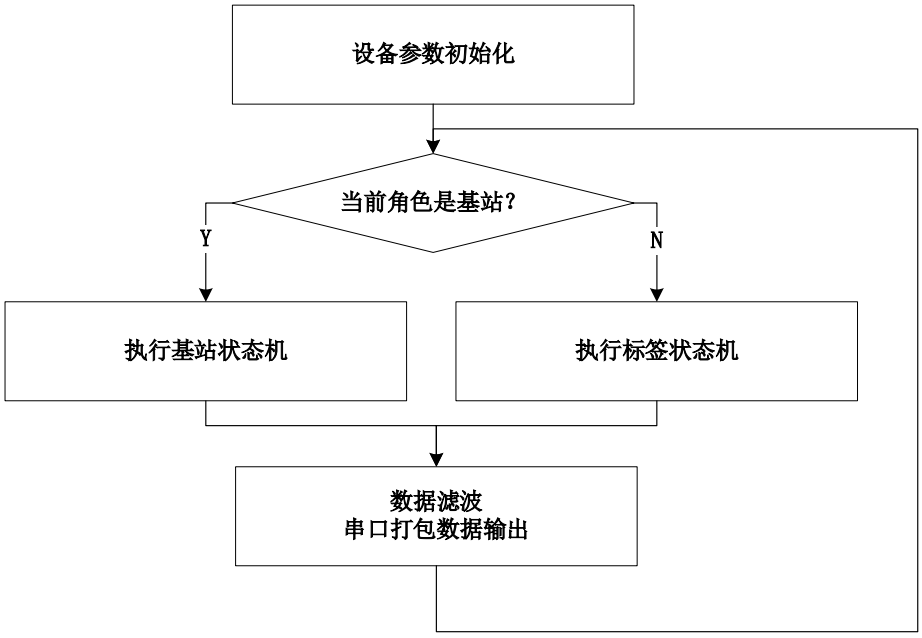


图 7-1 主程序工作流程图

7.2 标签工作流程

标签工作过程由状态机实现，完成单周期 TWR 全过程，主要工作流程如图 7-2 所示，其中蓝色 STA 开头的是程序内状态机标识符，建议结合源码同步学习。

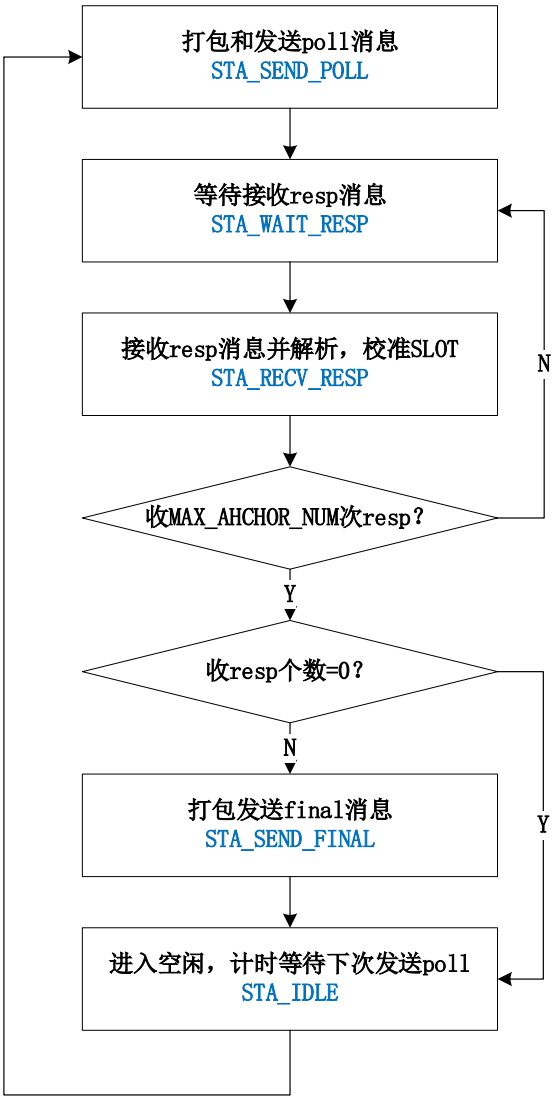


图 7-2 标签工作流程图

7.3 基站工作流程

基站工作过程由状态机实现，完成单周期 TWR 全过程，主要工作流程如图 5-7 所示，建议结合源码同步学习。

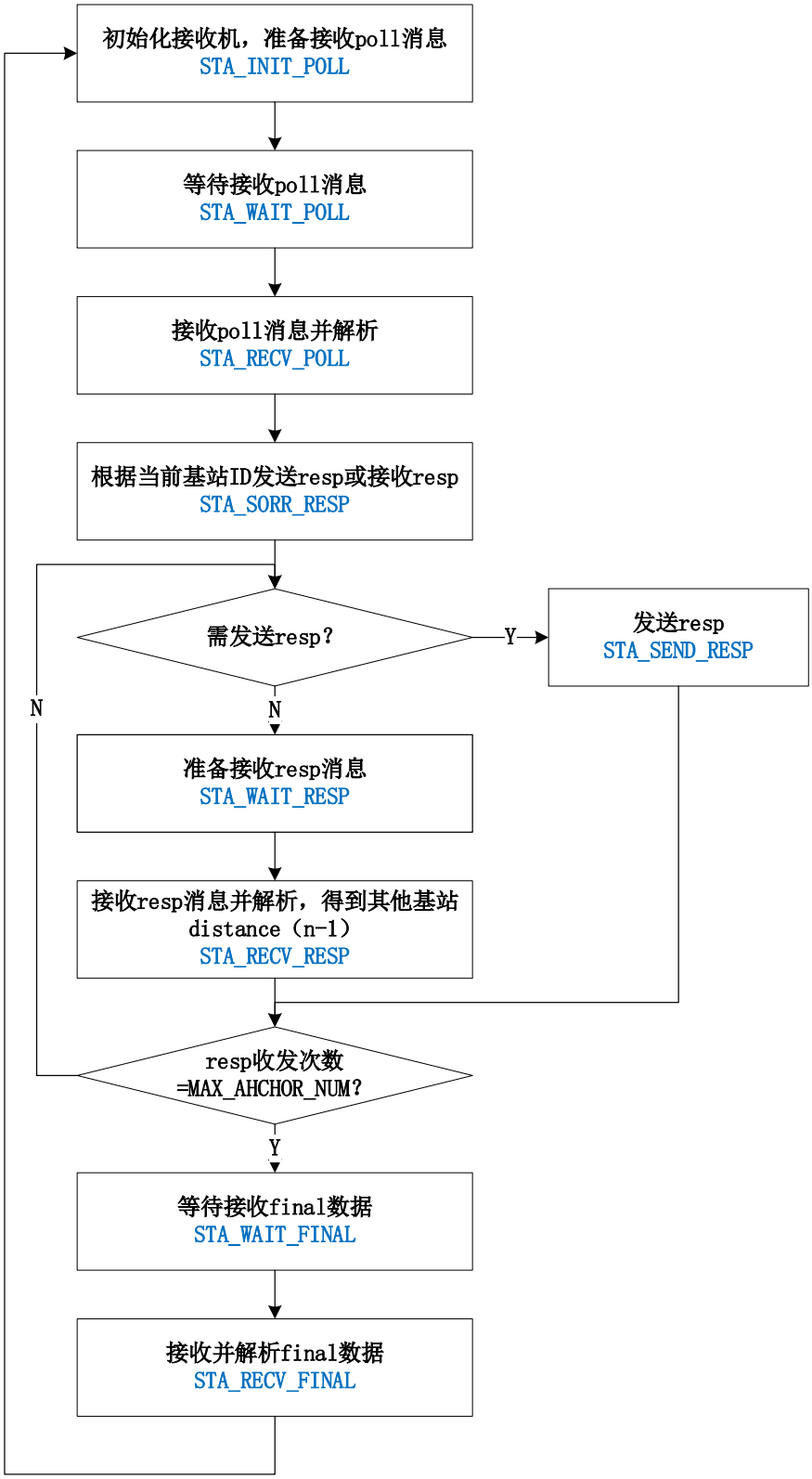


图 7-3 基站工作流程图

8 开发环境说明

使用 CUBEmx 配置系统参数和底层参数，随代码工程提供 CUBE 配置 IOC

文件，如只对 UWB 模块进行应用层二次开发，基本不用修改 CUBEmx 配置，CUBEmx 属可视化操作，较为简单，此处不再赘述。开发 IDE 使用 KEIL-MDK，我公司开发时使用的版本为 V5.25.2，该版本自带 ARM CompilerV6.9，理论上 V5.25.2 版本以上自带 ARM CompilerV6.9 以上版本都可向下兼容，如编译和烧录出厂程序后出错，请降级至 ARM CompilerV6.9 使用。

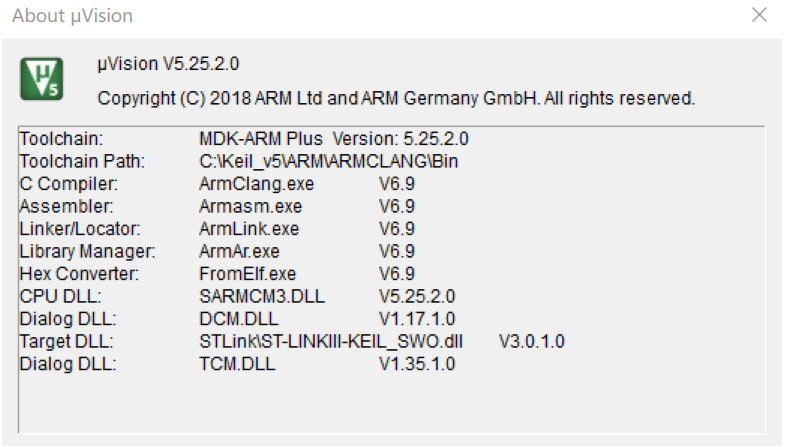


图 8-1 KEIL-MDK 版本

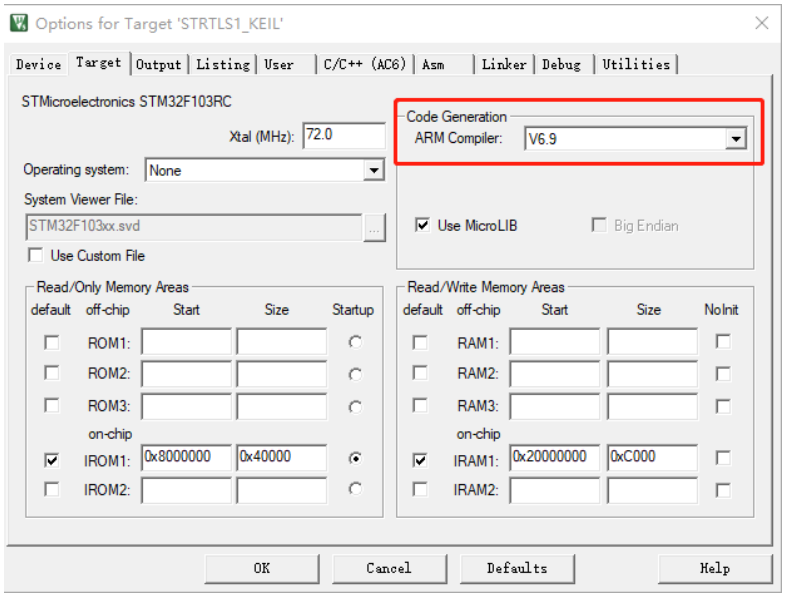


图 8-2 ARM Compiler 版本

另需注意工程配置 Options for Target 内的 c/c++内按如下配置：

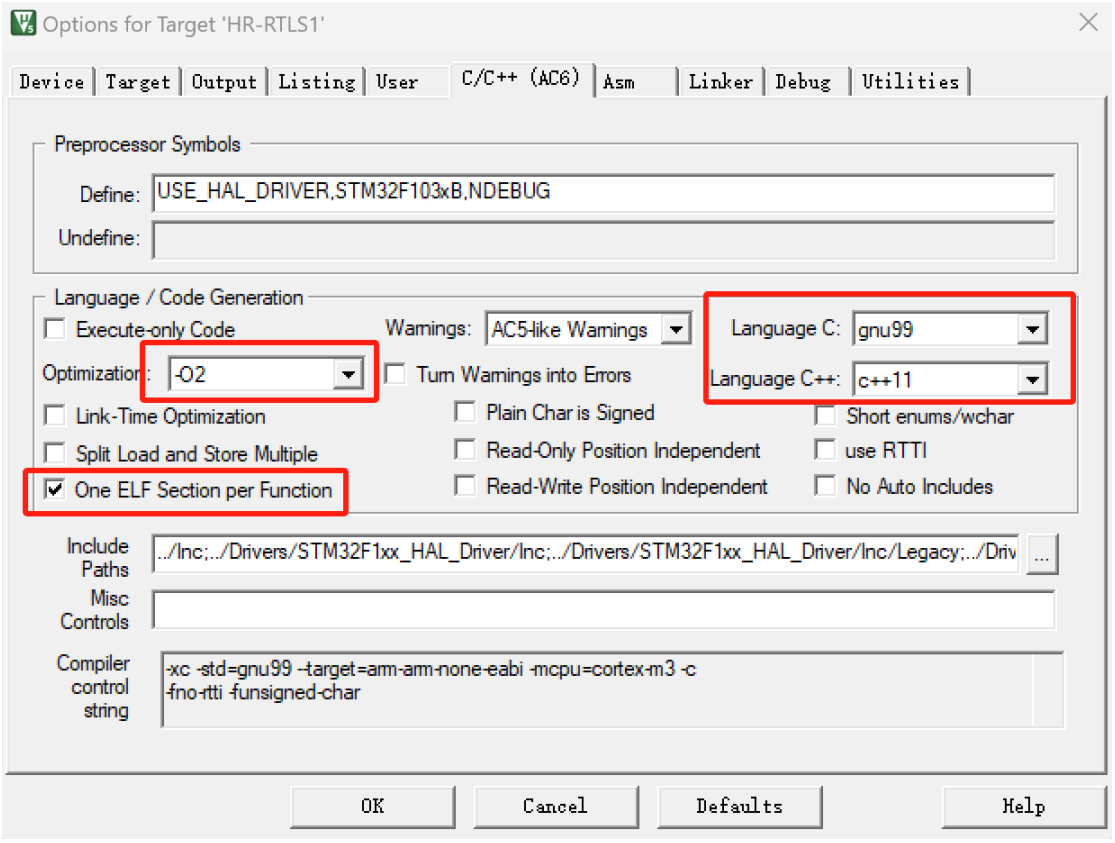


图 8-3 Options for Target 配置

9 串口通信协议

波特率 115200bps，8 位数据，无奇偶校验，1 位停止位，字符串 ASSCII 接收。

例：mc 0f 00000663 000005a3 00000512 000004cb 095f c1 008e8c8a a0:0 185c $KT0,1.69,2.93,4.98,NULL,LO=[-2.45,5.44,1.43]

表 9-1 串口通信协议说明

内容	例子	功能
HEAD	mc	消息头，固定为 mc
MASK	0f	表示 4 个测距值有哪几个是有效的； 例如 mask=0x07(0000 0111)表示 RANGE0,1,2 有效
RANGE0	00000663	标签到基站 A0 的距离 16 进制，单位 mm
RANGE1	000005a3	标签到基站 A1 的距离 16 进制，单位 mm
RANGE2	00000512	标签到基站 A2 的距离 16 进制，单位 mm
RANGE3	000004cb	标签到基站 A3 的距离 16 进制，单位 mm

FN	095f	frame number 串口消息流水号，每帧数据+1
RN	c1	Range number 不断累积增加
RANGETIME	008e8c8a	测距时间（单片机系统时间）16 进制，单位 ms
rIDt:IDa	a0:0	r 为当前角色，a 为基站，t 为标签； IDt 为标签地址，IDa 为基站地址
RX_POWER	185c	转换为 10 进制后除以 100 取负数极为接收功率

最后一个字段 rIDt:IDa 补充说明：

如当前基站连接电脑：r=a 表示当前是基站，IDt 表示标签 ID，即该条数据是基站到哪个标签的距离值，IDa 是基站 ID，代表当前连接电脑的基站 ID。例：

1、基站 A0 接电脑，标签 T0 开机【a0:0】

2、基站 A0 接电脑，标签 T1 开机【a1:0】

2、基站 A1 接电脑，标签 T1 开机【a1:1】

r=t 表示当前接电脑的是标签，IDt 表示标签 ID，冒号:后面的 0 固定不变。例：

1、标签 T0 接电脑， 基站 A0 开机【t0:0】，此时 RANGE0 有数值

如当前设备为标签，则输出 mc 数据后紧跟着会输出测距和定位信息：

例：$KT0,1.69,2.93,4.98,NULL,LO=[-2.45,5.44,1.43]

分别表示当前角色为 T0，K 表示开启卡尔曼滤波，NK 表示不用开启卡尔曼滤波，到 A0 基站的距离值为 1.69m，到 A1 基站的距离为 2.93m，到 A2 基站的距离为 4.98m，到 A3 的距离值未得出或 A3 不存在或未开机。

LO 后面的中括号内为标签的实时定位坐标，该坐标值在标签内部进行解算，需要注意的是要在标签固件代码里提前配置好基站坐标，才能完成解算。

10 常用 API 说明

请结合官方《DW1000/3000_Software_API_Guide.pdf》一起参考学习与使用。

10.1 dwt_writetxdata

int dwt_writetxdata(uint16_t txBufferLength, uint8_t *txBuffer, uint16_t txBufferOffset);

该 API 用于将发送消息数据写入 DWIC 的内部发送缓冲区。

type	name	description

uint16_t	txBufferLength	写入 TX 缓冲区的总长度，它可以包含两字节 CRC 的空间，但不一定要包含。设备将自动在发送的 TX 数据的最后两个字节中放入两字节 CRC。要传输的数据长度由 dwt_writetxfctrl()中的 txFrameLength 参数指定
uint8_t*	txBuffer	发送数据指针
uint16_t	txBufferOffset	开始写入数据的偏移量

10.2 dwt_writetxfctrl

void dwt_writetxfctrl(uint16_t txFrameLength, uint16_t txBufferOffset, uint8_t ranging);

该 API 用于配置 TX 帧控制寄存器。

type	name	description
uint16_t	txFrameLength	帧总长度，需包括 2 个字节的 CRC
uint16_t	txBufferOffset	开始写入数据的偏移量
uint8_t	ranging	如 1 是测距帧，0 是普通数据帧

10.3 dwt_starttx

int dwt_starttx(uint8_t mode);

该 API 用于启动帧数据发送。

type	name	description
uint8_t	mode	DWT_START_TX_IMMEDIATE 立即发送 DWT_START_TX_DELAYED 延时发送，通过 dwt_setdelayedtrxtime()设置发送时间 DWT_RESPONSE_EXPECTED 发送后打开接收机自动启动接收

10.4 dwt_setdelayedtrxtime

void dwt_setdelayedtrxtime (uint32_t starttime);

该 API 设置延迟发送模式中延迟的发送时间或延迟接收模式中接收机开启的时间。在调用 dwt_starttx()或 dwt_rxenable()以延迟为参数时，应调用此函数以

设置所需的发送或开启接收机的时间。

type	name	description
uint32_t	starttime	发送或接收开始的时间，参数为系统时间戳的高 32 位，用于发送消息或开启接收器

10.5 dwt_setrxtimeout

void dwt_setrxtimeout (uint32_t time);

该 API 用于在指定时间内未收到任何帧时，将接收器设置为超时。应在 wt_rxenable()之前调用此函数。

type	name	description
uint32_t	time	超时时间以微秒为单位。如果该值为 0，则将禁用超时。最大值为 0xFFFFF。

10.6 dwt_rxenable

int dwt_rxenable(int mode);

此 API 函数用于打开接收器以等待接收数据帧，在延迟接收模式下，接收器开启直到通过 dwt_setdelayedtrxtime()设置的超时时间后结束。

type	name	description
int	mode	DWT_START_RX_IMMEDIATE 立即开启接收机 DWT_START_RX_DELAYED 延迟开启接收机，开始时间在 dwt_setdelayedtrxtime()中设置