# 底盘代码详解

李心涛 2022/6/28

目录

[底盘代码详解 1](#_Toc107315648)

[1. 整体框架 3](#_Toc107315649)

[1.1 整体框架图 3](#_Toc107315650)

[1.2 框架简单解释 3](#_Toc107315651)

[2. 底层驱动 3](#_Toc107315652)

[2.1 Bsp文件 3](#_Toc107315653)

[BspMotor.c/h 3](#_Toc107315654)

[CanReceiveDecom.c/h 5](#_Toc107315655)

[ChassisSendProtocol.c/h 6](#_Toc107315656)

[bsp\_referee.c/h 6](#_Toc107315657)

[vofa.c/h 6](#_Toc107315658)

[2.2 Algorithm文件 6](#_Toc107315659)

[3. 逻辑层 7](#_Toc107315660)

[Application 7](#_Toc107315661)

[ChassisPowerBehaviour.c/h 7](#_Toc107315662)

[RefereeBehaviour.c/h 7](#_Toc107315663)

[RM\_Cilent\_UI.c/h 7](#_Toc107315664)

[4. 任务层 8](#_Toc107315665)

[Task 8](#_Toc107315666)

[DVTask.c/h 8](#_Toc107315667)

[RefereeTask.c/h 8](#_Toc107315668)

[CanSendTask.c/h 8](#_Toc107315669)

[UI.c 8](#_Toc107315670)

[ChassisTask.c/h 9](#_Toc107315671)

[5. 通信分析 9](#_Toc107315672)

[5.1 裁判系统通信 9](#_Toc107315673)

[5.1.1 流程框图 9](#_Toc107315674)

[5.1.2 步骤分析 9](#_Toc107315675)

[5.2 云台、视觉通信 10](#_Toc107315676)

[5.2.1 流程框图 10](#_Toc107315677)

[5.2.2 步骤分析 10](#_Toc107315678)

[5.2.3 综合分析 10](#_Toc107315679)

[6. 功率闭环以及控制分析 11](#_Toc107315680)

[6.1 框图分析 11](#_Toc107315681)

[6.2 步骤分析 13](#_Toc107315682)

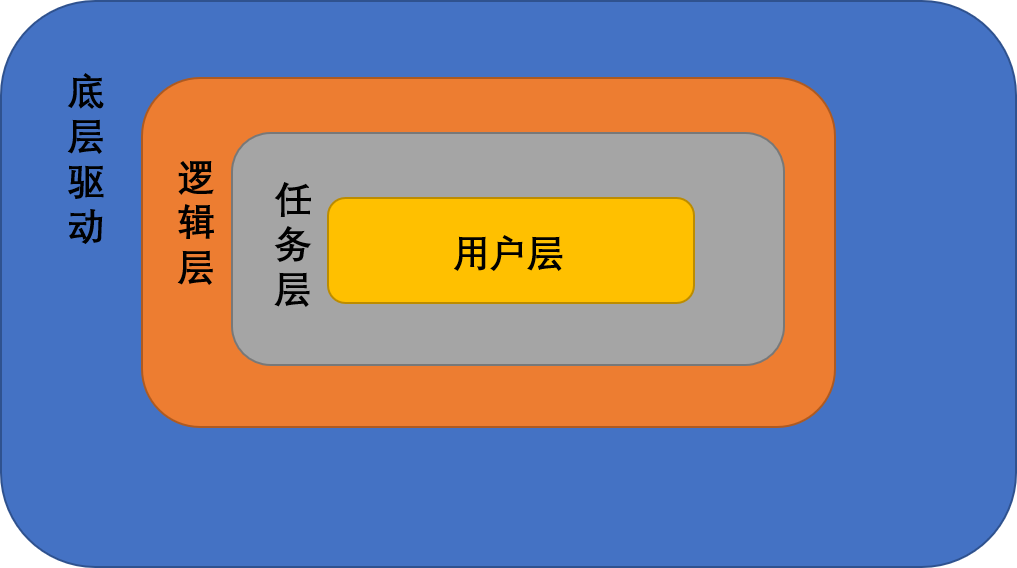
[6.3 理论分析 13](#_Toc107315683)

[7. UI分析 13](#_Toc107315684)

[8. 调试效果 13](#_Toc107315685)

## 1. 整体框架

### 1.1 整体框架图



### 1.2 框架简单解释

1. 底层驱动：包含Algorithm和Bsp两部分
2. 逻辑层：包含Application
3. 任务层：包含Task
4. 用户层：包含UI以及未实现的用户交互部分

## 2. 底层驱动

### 2.1 Bsp文件

#### BspMotor.c/h

功能：实现电机控制指令的发送

函数：



#### CanReceiveDecom.c/h



#### ChassisSendProtocol.c/h



#### bsp\_referee.c/h



#### vofa.c/h



### 2.2 Algorithm文件

这部分属于基础代码，没有多余内容需要强调。

注意：pid.c/h文件多添加了PID\_ANGLE模式

## 3. 逻辑层

### Application

由于电容任在开发中，超级电容部分相关函数被暂时不做解释。

#### ChassisPowerBehaviour.c/h



这一部分包含底盘功率闭环的重要核心，后面会单独详细介绍

#### RefereeBehaviour.c/h



#### RM\_Cilent\_UI.c/h

功能：UI绘制的基本函数，这份BSP库由“山东理工大学 齐奇战队“提供，注释完整，不做说明。

注意：结构体的定义必须按照字节对齐的方式，即使用#pragma pack(1)、#pragma pack()函数进行对结构体定义的包裹。

## 4. 任务层

### Task

#### DVTask.c/h

功能：用于数据可视化(data view),主要是向VOFA+发送数据用于图像绘制。

#### RefereeTask.c/h



#### CanSendTask.c/h

功能：向云台与视觉发送存储在FIFO中的数据包

函数：

1. static void Can\_Send(CAN\_HandleTypeDef \*hcan,uint32\_t id,uint8\_t lenth,uint8\_t \*buffer)

功能：发送指定字长与ID的CAN数据包

#### UI.c

功能：UI绘制

#### ChassisTask.c/h



[缓冲函数](https://github.com/AIRitane/Code_Base/blob/main/README.md)

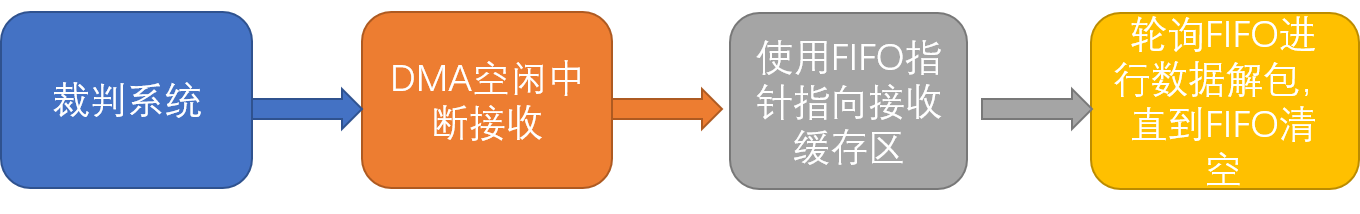
## 通信分析

通信部分主要包含裁判系统通信以及云台、视觉通信两部分。

#### 裁判系统通信

裁判系统通信使用串口6进行。

##### 5.1.1 流程框图



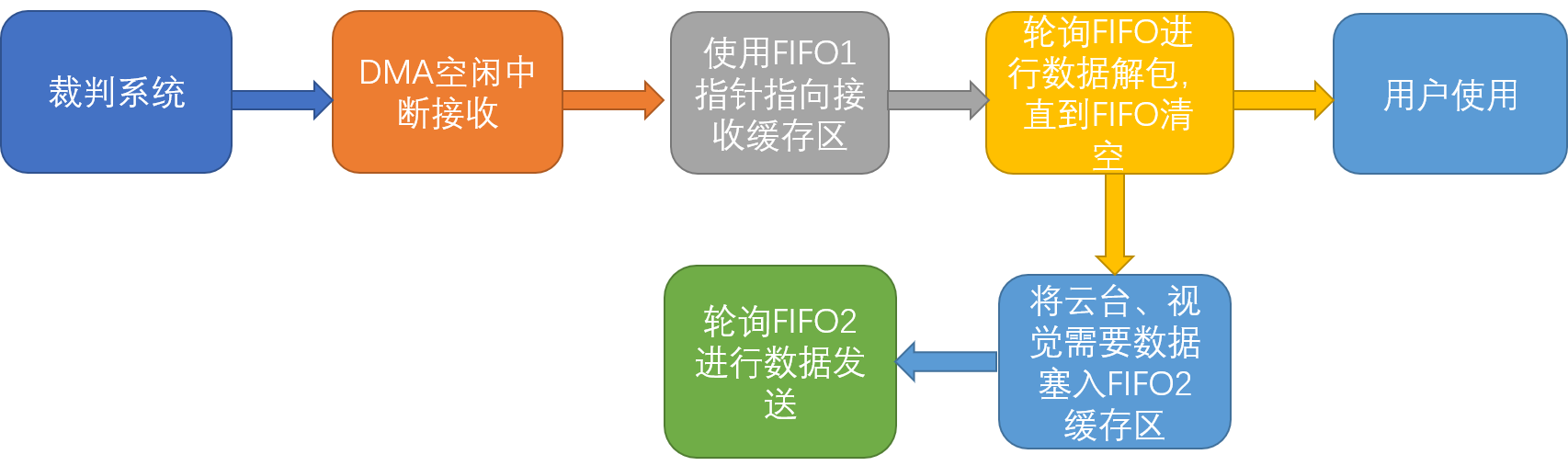
##### 5.1.2 步骤分析

* DMA空闲中断接收：使用双缓存区，详见void usart6\_init(uint8\_t \*rx1\_buf, uint8\_t \*rx2\_buf, uint16\_t dma\_buf\_num)函数。注意这里rx1\_buf和rx2\_buf需要在地址空间中连续，最简单的方法解释二维数组。
* 使用FIFO指针指向接收缓存区：这里直接将缓存区作为FIFO数组缓存，由int fifo\_s\_init(fifo\_s\_t \*p\_fifo, void \*p\_base\_addr, int uint\_cnt)定义，其中uint\_cnt为双缓存区的总长度。
* 轮询FIFO进行数据解包，直到FIFO清空：这部分解包协议参考接收机/遥控器附带的大疆例程，官方并未提供具体的解包协议，只能通过代码反推。

#### 云台、视觉通信

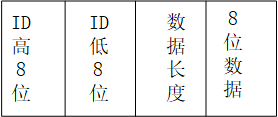
云台、视觉通信使用CAN1总线进行通信。

##### 流程框图



##### 步骤分析

* 将云台、视觉需要数据塞入FIFO2缓存区：这部分代码的实现需要先进行数据包的制作，详见"ChassisSendProtocol.c/h"文件，数据包格式为：



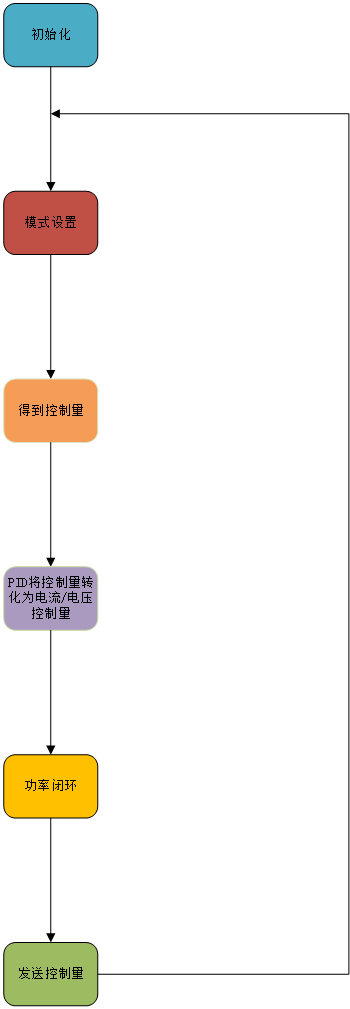
* 将云台、视觉需要数据塞入FIFO2缓存区：详见void EnQueue(Queue\_t \*q, uint8\_t \*val,uint8\_t lenth)函数，注意这里一共使用两个FIFO，分为高低优先级。
* 轮询FIFO2进行数据发送：避免CAN\_FIFO满，最多一次发送两包数据。这里使用FIFO，一次发送一包数据，先轮询高优先级FIFO0，发送完毕后发送低优先级FIFO1。

##### 综合分析

这种通信使用FIFO的方法，错开发送时间，可以维持个较高的数据发送效率，提高数据发送频率。

## 功率闭环以及控制分析

##### 6.1 框图分析



##### 6.2 步骤分析

* 初始化：引入数据指针以及初始化PID结构体
* 模式设置：根据云台发送指令确定模式。
* 设定值得到数据量：根据等级与底盘功率一对一进行对点调试，Erro\_angle用于补偿相位，rote\_powkp控制前后左右速度，roting\_speed控制小陀螺速度，调控底盘主要就是调控这三个参数。
* 根据设定值进行PID
* 功率闭环。

##### 6.3 理论分析

点击见[链接](代码注释附件.pdf)

## UI分析

* 需要先添加UI\_Graph\_ADD执行一次，再执行更改UI\_Graph\_Change
* Imagename为3个字符，每个独立UI前3个字符必须不同，客户端才能区分
* 注意裁判系统对于带宽的需求

## 调试效果

* 随动模式下功率 = 功率上限-5，允许开启或刹车时使用缓存功率，1秒后缓冲功率回复，2秒内不使用缓存功率。该模式下除了启动与刹车外，前后左右走直线，不包含启动后底盘斜着走，但是为直线的情形，必须以云台坐标系为准。
* 小陀螺不动模式下功率 = 功率上限 – 5，允许开启或刹车时使用缓存功率，1秒后不使用缓存功率。该模式下不能产生底盘中心漂移。
* 小陀螺运动状态下功率 = 功率上限，允许使用缓存区，但离开10-15米的时候缓存功率处于20W及其以上，也不建议过多使用缓存功率
* 小陀螺下除了启动与刹车外，前后左右走直线，不包含启动后底盘斜着走，但是为直线的情形，必须以云台坐标系为准。
* 启动与刹车以及运动需要经过操作手确认，是否刹车缓慢、运动是飘的、跟随云台缓慢等等