在Robomaster2020赛季超级电容的开发中，我作为超级电容软件控制部分的负责人，模块的电源切换，功率监控，恒功率控制和通讯由模块上的STM32F103C8

MCU实现。在超级电容的算法设计以及程序设计方面遇到了一些问题并成功解决，以下是我们的超级电容软件部分开发心得，包括遇到的问题以及我们的解决方案。

1.对超级电容以恒功率进行充电控制

恒功率控制需要包含以下三个部分：

一、最大充电电流保护：考虑到超级电容固有特性，在充电初期由于其电压较小，以大功率进行充电时很容易导致电流过大，进而可能被裁判系统判定为超功率的情况。因此我们采用STM32F103C8Tx的MCU进行功率实时监测，并让MCU发送指令控制DAC从而设定充电最大电流与电压的大小，实现开始时的暂时降低充电功率的保护功能。

二、对超级电容进行恒功率充电

初期进行电流电压保护之后，电容控制模块便以恒功率对超级电容进行充电，对电容进行充电的功率要根据底盘实时功率以及功率限制共同决定，因此我们采用PID闭环控制以实现对电容的恒功率充电。

三、超级电容的电压保护：

当充电模块的充电电压达到超级电容的满充电压时，通过MCU控制实现限幅管理，防止电容电压进一步升高。

2.关键设计模块：电容控制模块对超级电容的恒功率充电

考虑到时间以及技术实现复杂度的因素，我们在软件部分采用了PID闭环控制充电功率的算法，即从软件上实现闭环控制，在充电开始时设置限制功率任务，限制功率的程度随着充电时间的增加而减小，在结束限功率任务后，MCU进入恒功率控制充电模式，由于PID闭环控制存在着一定的延时，系统的响应速度有限，因此我们在分给电容的充电功率设置冗余量20W，以避免超功率的情况发生。当给电容充电时：

（1）超级电容在供电时：充电功率 = 功率限制 – 冗余量；

（2）底盘在供电时：充电功率 = 功率限制 – 冗余量 – 底盘功率。

超级电容控制板的MCU连接DAC，DAC的输出端连接压降模块的恒流设置端，从而根据检测到的功率控制电容放电电流的大小。在软件中我们设置的PID参数中电流的最大电流是10A，最大电压是23.5伏特，PID的比例、微分、积分系数分别为1.0、0.07、0.025。超级电容的控制板与主控的通信采用CAN通信，超级电容主控板接收到机器人种类指令进而据此进行不同的功率上限与缓冲热量的限制，并根据主控板发送来的电源切换指令并进行电路供电的切换。通过CAN通信给主控板发送被启动的供电电源的类型、底盘的实时功率、超级电容所剩的电量、充电电压。

3.软件程序设计中遇到的问题及解决方案

在开始的程序恒功率闭环控制中，我们未考虑到初始时电容电压较小从而充电电流与功率较大的问题，只是全过程单一的利用PID进行闭环控制，并没有根据充电的具体情况进行限幅，在硬件组的同学进行仿真指出问题后，我们采用了限幅函数，在开始充电前十秒左右的时间内进行降功率充电，降功率的幅度随着时间的推移而变得越来越弱，以此实现开始时的限功率充电目的。