上周主要是对探讨的问题进行了深入的了解研究、增加了一些功能、调试UI界面。

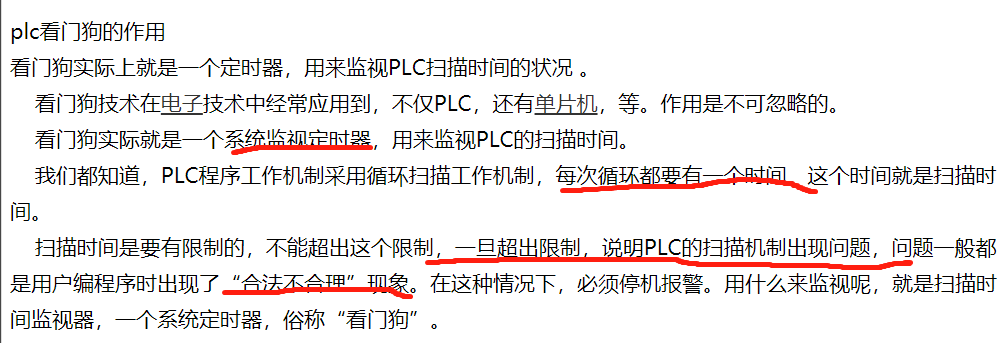
上周总结：

1. 作业比较多，花了挺多时间写作业的（高中同学来北京）
2. 断电重启——看门狗

进行到一半的任务：开机时重启就好（上电自启动）。类比于手机，正在执行的APP在手机断电重启后，也是重新启动的，中间的各个状态记录起来比较麻烦，重启就行。

系统状态会在断电后自动清零，重新来电后需要上位机发送启动信号重新执行任务。相对应的，上位机的交互界面会有个清零（任务标志位）的机制。

看门狗机制：查到的资料，PLC的看门狗是为了防止程序的问题



https://www.dgzj.com/plc/60057.html

主要是为了避免程序合法不合理，而我们能对程序的结果进行监测，如果出现了真正的死循环，可以直接手动停止这一任务。系统自动检测死循环出现防止占用过多资源，这一点还没找到很好的方法，师兄们如果有相关经验欢迎交流。

程序指针。指令寄存器出错。指令指针。硬件强制恢复。周期性的重启。

死机重启：邻居可以检测到某节点失去连接：如果上位机可以连接到，则发送远程重启指令；如果上位机也连接不上，那么可能是断网或者死机，需要工人到现场使其手动重启，然后重新加入网络。

看门狗硬件重启。

Linux reset，已知的起始状态不断重启。

Linux死机原因

周期性重启，不能持续的运行，标志位乱飞。

测试树莓派8小时，连续一周。

1. 上位机数据镜像

做了个数据镜像的例程，可以将各节点的数据库同步到上位机中。基本思路是将本地数据库的数据读取，然后按照基本拓扑的生成树，由根节点向父节点逐步传输数据，最后汇总到根节点，根节点把数据传给上位机，上位机将这些数据恢复为数据库的形式。

是个单独的任务，执行一次进行一次同步，因为数据量可能会很大，所以按需求进行周期执行。在上位机中可以看到各节点的数据库的情况。（此处需要界面、也可以直接在数据库中查看就好）

下面针对函数中的sendUDP函数的探讨，

1. 提示信息：上周的修改意见是发给父节点，而非上位机

了解了广播风暴：广播帧向除源端口之外的所有其他端口转发。如果交换网络中有环路，则这个帧会被无限转发，此时便会形成广播风暴，网络中也会充斥着重复的数据帧。1. 要避免物理上的网络环路。

实际上我们的提示信息函数使用的是UDP单播而非UDP广播，所以本质上不会产生广播风暴。

第二点是考虑到对树莓派的影响，如果是现在这个机制，每个树莓派只需要将信息发到上位机就行（本身上位机就可以给每个树莓派传输文件、发送重启信号），如果是发送到父节点的话，那么根节点就会有大量的信息通信需求，如果节点数很大，每次传输的数据文本较长，就会影响根节点的性能，从而影响整个系统的性能。

对整个网络的查看可以由节点数据镜像获取，所以我这边目前的做法是sendUDP不需要进行父节点的收集。

1. 编号释放、重新分配。现在卸载一个app后，可以给新的任务分配空缺的编号。
2. 界面调试工作

后续功能:

1. 界面的硬件调试机制，开启后记录所有的通信过程，上位机可以通过相应的命令查看。
2. 网络安全防止攻击，加入身份验证机制，在现有通讯基础上加入用户名和密码
3. 将所有的新功能加入界面。（考虑数据库镜像的显示、调试机制的查看等一些功能的显示效果）
4. 接入硬件（时间有限可以调整为数据库的操作，有了界面可以实际化界面），实际演示

山东建筑大学、周五

1. 深度学习人员识别、火灾监测

动态组网

数据末端的节能控制算法

展示的功能：

1. 基础拓扑的变化过程
2. 简单的应用：负载均衡，数人（2个摄像头），大屏展示摄像头的画面。
3. 持续运行