流程：

清华：

1. 根据图纸设计拓扑
2. 调试算法
3. 制作镜像 .img文件

现场（仓库）：

1. 树莓派根据指定的镜像文件装机
2. 第一次启动后将其连入局域网，路由器为其分配IP
3. 为该树莓派分配位置码，并在binding.csv中加入位置码和该IP的绑定关系
4. 在路由器中设置相应的mac地址和IP地址为静态绑定
5. 部署好当前批次后，上位机整理好binding.csv，将改文件通过ip广播到文件中所有能访问到的IP，所以此时需要保持树莓派在开机状态
6. 全部发送成功后，将树莓派安装到实际的房间，并重启，等待上位机指令。

任务运行基本流程：

1. 待树莓派全部部署好，上位机开启监控程序，

python .\script\moniter.py

1. 运行

python . \script\run.py [id]

系统会以位置码为id的树莓派作根节点启动系统并运行系统任务，实施检测拓扑的连接情况

1. 系统任务顺利执行后，运行

python .\ script\starttask.py [id] [t1,t2…tn] [delay1,delay2,delayn]

系统会以位置码为id的树莓派作根节点启动编号为t1到tn的任务，并在延时delayi秒后运行这一任务。

1. 想要在任务运行的时候停止，运行

python .\ script\stoptask.py [id] [ti]

其中id为该任务的发起节点的位置码，ti为任务相应的编号

1. 若某一时刻一个节点故障、与周围邻居失去连接，重启这一节点，待其重新连入网络后，运行

python . \script\restart.py [id]

id 为该故障节点的位置码

1. 如果想更新文件，若要更新binding.csv，在上位机更新好后，运行

python . \script\broadcast\_binding.py

可将本地的binding.csv文件广播到文件中所有的ip所指示的树莓派上。

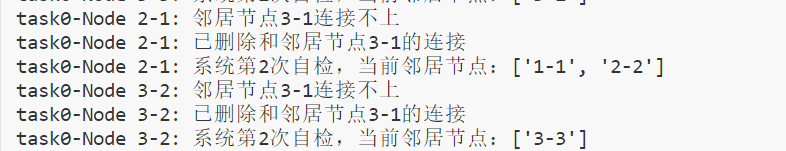
若想增加新的任务，在本地完善好算法文件和拓扑文件后，运行

python . \script\broadcast\_task.py [ti]

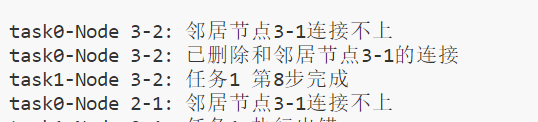
其中ti为相应的任务编号

相比较与部署在树莓派的POS1.0，当前的POS2.0的区别

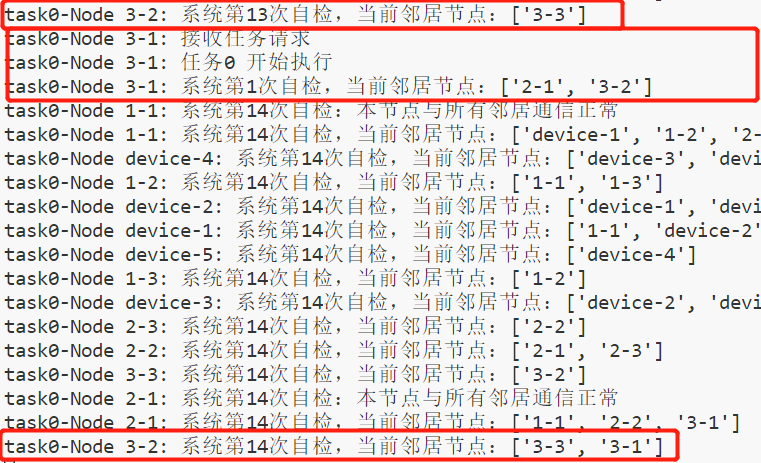
1. 加入了并行机制，运行任务时多个任务可以同时运行，且可以延时启动任务，并可打断运行中的任务。
2. 加入了即插即用机制：
3. 当任务未运行时节点故障，系统任务可以重新组织剩余节点，删去与故障节点的连接



1. 任务执行过程中节点故障，会提示用户节点连接不上，并继续以剩余的节点运行任务，任务可能因此会报错，但系统没事



1. 系统启动前、任务执行前节点故障，会直接忽略这一节点，以剩余节点启动任务
2. 节点故障后，重启这一节点，可以使其重新加入网络



1. 若有节点的增加或删除，属于更新设备，要修改基础拓扑，需要所有节点全部重启。
2. 保持着任意节点启动任务，分布式运行

在DSP3.0的版本时，根节点始终为发起节点，如果这一节点故障，整个系统都会故障

现在任意任务能以网络中的任意节点启动

1. 树莓派和房间的对应关系由binding.csv给定，因此每个树莓派上的程序是完全相同的，下载好系统查看IP后就不需要特意连接调试了。另外此时电脑端调试的拓扑文件可以直接用于树莓派任务执行。
2. 文件广播程序

可以在上位机直接给所有的树莓派节点发送新的任务

1. IP不用手动指定，防止冲突
2. 通信函数修改，现在同步通信函数可以给邻居发送不同的数据

后续功能:

1. 命令行操作->任务管理界面
2. 任务用编号管理->任务内核用编号，用户交互用文件名管理
3. 手动电源重启->可以远程一键重启
4. 接入硬件，实际演示

alter table controller\_fan\_onoff modify column value int;

**insert into controller\_fan\_onoff (`id`, `name`, `value`) VALUES ("0xxxxxxxxx","fanonoff",60);**