TI AM3352开发板的串口和TCP通讯实现方案

**1、app**

1.1功能简介

-- 管理TCP模块和uart模块

1.2输入

-- 串口模块队列消息

-- tcp模块队列消息

1.3处理

-- 消息队列初始化

-- 串口模块初始化

-- TCP模块初始化

-- 启动串口和TCP模块

-- 循环读取消息队列中的消息（cmd，data)

-- 处理消息中对应的命令：

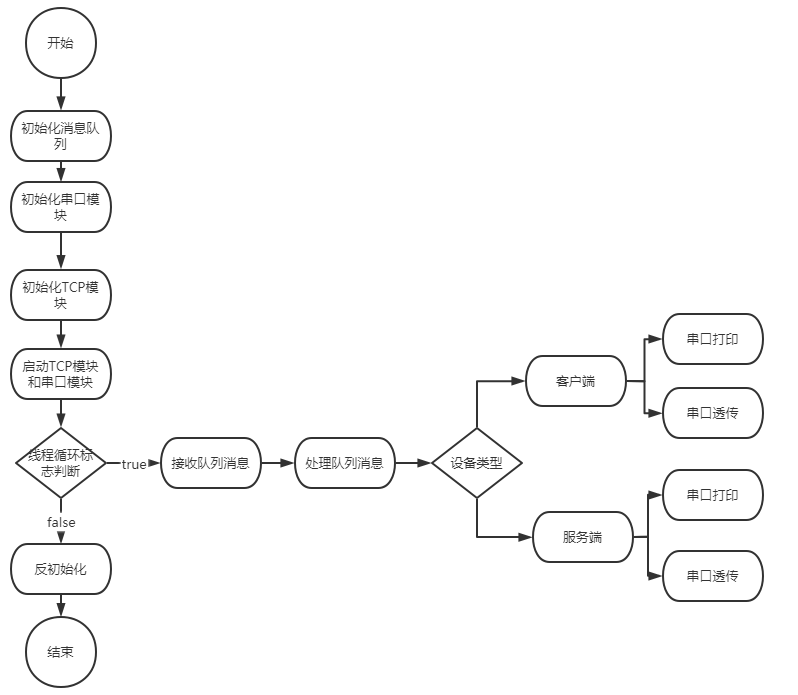
串口命令处理（串口信息打印）

tcp客户端命令处理（串口透传）

tcp服务端命令处理（串口透传）

1.4输出

-- 命令响应



**2、UART模块**

2.1功能简介

-- 负责开发板和上位机的通信

2.2模块输入

-- 串口数据接收

-- app层输入的数据

2.3模块处理

-- 循环监听串口是否可读，将收到数据写入uart\_recv\_fifo

-- 读uart\_recv\_fifo，进行HUP解析，解析完上报处理

-- 将app层数据的写入uart\_send\_fifo中的数据，读出来发给PC机

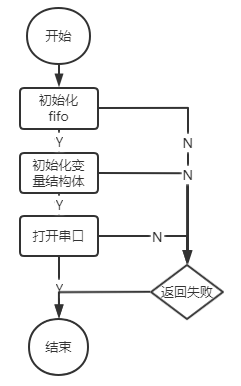
2.4模块输出

-- HUP解析好的数据

-- 发送给PC机的数据

初始化

初始化FIFO，初始化uart\_struct（msg\_fd)，打开串口

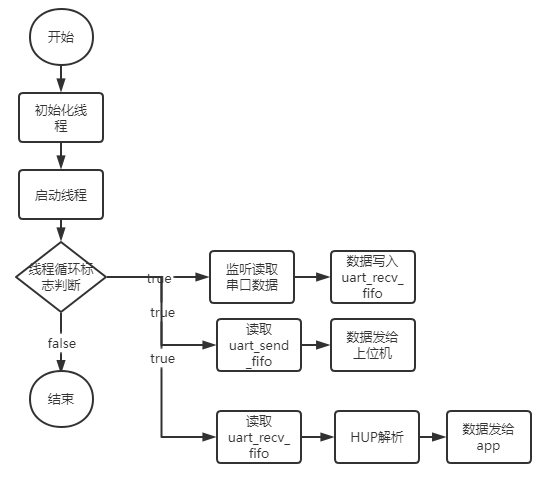


线程处理

线程1：收到数据写入uart\_recv\_fifo

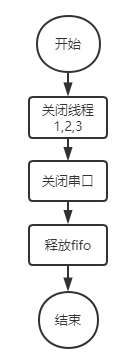
线程2：读uart\_recv\_fifo，进行HUP解析，解析完上报处理

线程3：将app层的写入uart\_send\_fifo中的数据，读出来发给PC机



去初始化

关闭线程1，2，3，关串口fd，释放两个fifo



**模块接口设计**

Init

Int uart\_mod\_init(void \*uart\_ptr, int q\_msg\_fd);

|  |  |
| --- | --- |
| param | uart\_ptr：串口结构体变量地址 |
| param | q\_msg\_fd:消息队列句柄 |
| return | 成功：0  失败：-1 |

* deinit

void uart\_mod\_deinit(void \*uart\_ptr);

|  |  |
| --- | --- |
| param | uart\_ptr：串口结构体变量地址 |
| return | 无 |

int uart\_mod\_start(void \* uart\_ptr);

|  |  |
| --- | --- |
| param | uart\_ptr：串口结构体变量地址 |
| return | 成功：0  失败：-1 |

int uart\_mod\_stop(void \* uart\_ptr);

|  |  |
| --- | --- |
| param | uart\_ptr：串口结构体变量地址 |
| return | 成功：0  失败：-1 |

int uart\_mod\_send( void \*uart\_ptr, char \*data, int data\_size, uint8\_t cmd);

|  |  |
| --- | --- |
| param | uart\_ptr：串口结构体变量地址 |
| param | data：数据 |
| param | data\_size:数据大小 |
| param | cmd:命令 |
| return | 成功：0  失败：-1 |

**3、TCP模块设计**

**3.1、TCP客户端模块**

3.1.1功能简介

-- 负责与TCP服务端的通信

3.1.2模块输入

-- 驱动层数据（TCP服务端发过来的）

-- app层输入的数据

3.1.3模块处理

-- 读取client\_send\_fifo中的数据，发给TCP服务端

-- 阻塞等待接收服务端的数据，并进行HIP粘包解析，本模块可以处理的处理掉，不能处理的就发送消息队列上报

-- 断线重连，发现断线之后（连接flag为为未连接状态）就去连接服务端，并发送登录消息，有回应就置为已连接状态

-- 心跳保持，检测最后一次发送给server消息的时间与当前时间之差是否超过约定发送在线消息时间，超过就发送在线消息，一段时间后，发送多个心跳后，接收服务端数据的时间与当前时间之差超过最后接收时间的约定时间，就关闭套接字，（连接flag状态置为未连接状态）

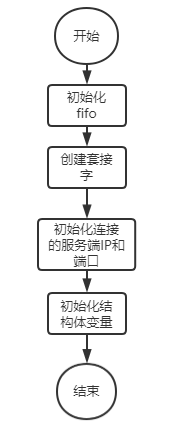
-- 粘包处理，先读recv\_fifo中一个HIP固定长度大小的包头，然后得到包头里面的数据长度信息，最后读取相应大小的数据。

3.1.4模块输出

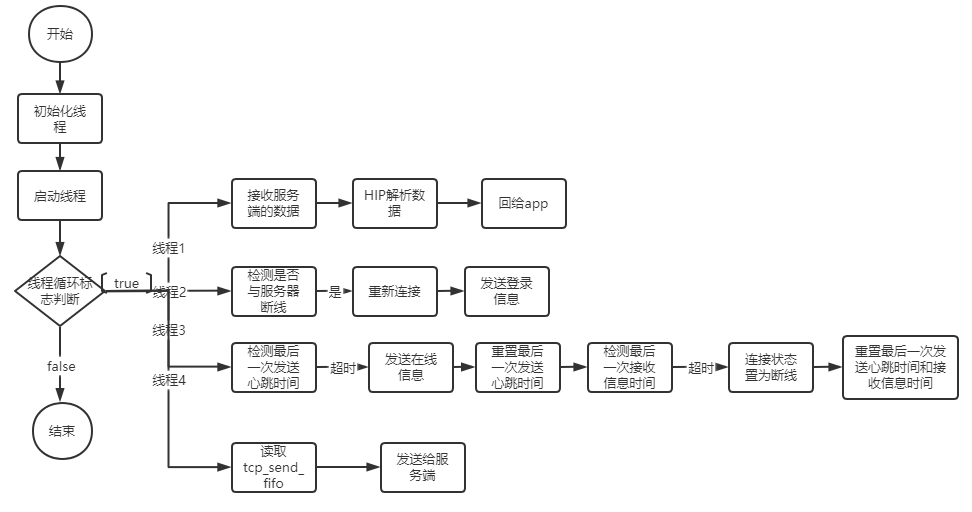
-- HIP打包后发送给服务端的数据

-- HIP解析后回给app的命令和数据

初始化



线程处理



去初始化



**3.2、TCP服务端模块**

3.2.1功能简介

-- 管理TCP的连接，和与多个TCP客户端的通信

3.2.2模块输入

-- 客户端请求连接

-- 驱动层数据（TCP客户端发过来的）

3.2.3模块处理

-- 服务端循环监听请求连接，检测其IP地址，未连接过IP，将对应IP地址，和套接字加入结构体数组中，已连接的关闭收到的套接字。

-- 循环监听所有在线的套接字，读取客户端发过来的数据，将其套接字和数据长度，数据写入对应tcp\_recv\_fifo中

-- 读取tcp\_recv\_fifo的数据，并进行HIP解析，本模块可以处理的先处理，不能处理的就回给app

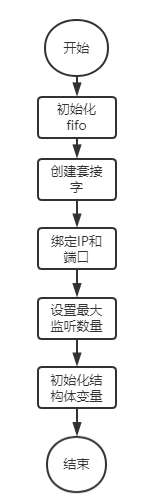
-- 心跳检测，循环检测，对每一个在线的客户端的最后一次接收其消息的时间，进行超时检测，超时就进行清理，已连接客户端数量减少

3.2.4模块输出

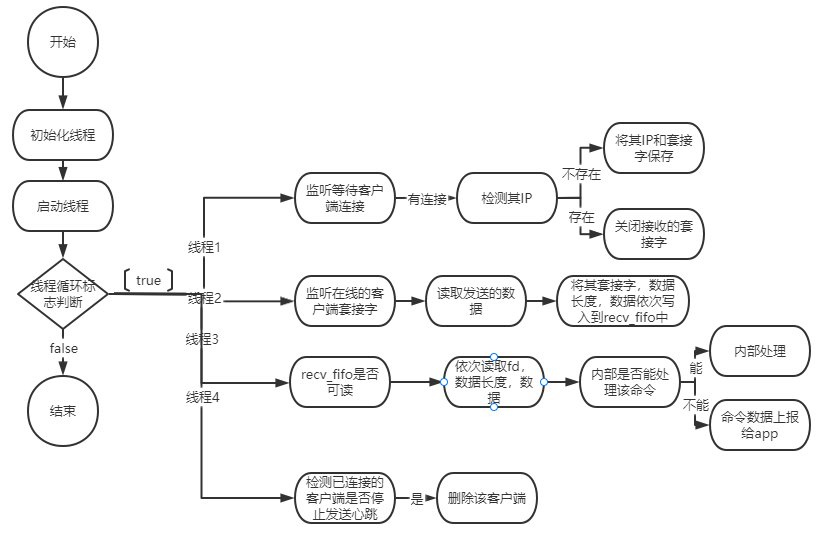
-- HIP协议打包后发送给客户端的数据

-- 解析回给app的命令

初始化



线程处理



去初始化



**模块接口设计**

Init

int tcp\_mod\_init(char \*cs\_type, void \*tcp\_ptr, struct param\_stru param);

|  |  |
| --- | --- |
| param | cs\_type:设备类型 |
| param | tcp\_ptr:变量结构体地址 |
| param | param:初始化参数结构体  ip地址，端口号，设备类型（客户端/服务端）等 |
| return | 成功：0  失败：1 |

* deinit

int tcp\_mod\_deinit(void \*tcp\_ptr);

|  |  |
| --- | --- |
| return | 成功：0  失败：1 |

int tcp\_mod\_stop(void \*tcp\_ptr);

|  |  |
| --- | --- |
| param | tcp\_ptr:变量结构体地址 |
| return | 成功：0  失败：1 |

int tcp\_mod\_start(void \*tcp\_ptr);

|  |  |
| --- | --- |
| param | tcp\_ptr:变量结构体地址 |
| return | 成功：0  失败：1 |

int tcp\_mod\_send(void \*tcp\_ptr, char \*data, int data\_size，uint8\_t cmd);

|  |  |
| --- | --- |
| param | tcp\_ptr:变量结构体地址 |
| param | data：数据 |
| param | data\_size:数据大小 |
| param | cmd:命令类型 |
| return | 成功：0  失败：1 |

**4、协议介绍**

HIP协议

HIP报文结构由固定字段和扩展字段组成，固定字段包含命令类型、报文长度、设备类型、通道号、设备ID 、ntp时间和对应的rtp时间；扩展字段根据commond不同而不同，方便功能扩展，详细说明见下表：

|  |  |  |  |
| --- | --- | --- | --- |
| **元素** | **长度（位）** | **描述** | **说明** |
| Version | 8 | 版本号 | 标识协议版本 |
| commond | 8 | 命令类型 | 详见commond对照表 |
| Length | 16 | 报文长度 | **整个HIP报文的长度,单位是字节** |
| Device type | 8 | 设备型号 | 详见device type对照表 |
| Device id | 64 | 设备id，唯一标识符 | 设备唯一的硬件ID,使用网卡物理地址 |
| Seq | 32 | 命令序列号 | 命令唯一的序列号，用于处理丢包、乱序等场景 |
| reserved | 不定 | 扩展字段 | 用于每个commond差异化的数据传输，可使用length-8得到扩展字段的长度 |