飞利浦太阳能CCU软件设计

**Revisions History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Author** | **Description** |
| 2014/6/15 | Draft | Julian Huang | Draft version |
|  |  |  |  |
|  |  |  |  |

# 1. 定义

在无线太阳能控制系统中，CCU（集中控制器）用于衔接RMS（上位机系统）及LCU（终端控制器），因此具备通过以太网或者GPRS/SMS与RMS连接，接受RMS的对自身或者LCU的控制，并将状态及时反馈给系统的功能。本文即描述CCU软件相关设计。

# 2. 框架

## 2.1 硬件框图

CCU硬件使用的是STM32F2系列MCU，同时外扩了SRAM与NAND falsh，外部接口有100／10M ETH，GPRS，RS－485等。

## 2.2 软件框图

CCU软件框架大致可以按以下层次划分：

Application

Middleware

BSP／Driver

OS

Hardware

## 2.3 框架设计

### 2.3.1 Hardware

该层即底层硬件与平台驱动，例如CCU使用的是STM32F2系列的标准驱动库。

### 2.3.2 BS/Driver/OS

该层提供了对上层统一接口，相当于HAL，不同平台硬件特性在该层实现分离，同时实现了基本的RTOS功能。

### 2.3.3 Middleware

该层及以上应用层与硬件平台无关，分为两部分：其一，通用的TCP/IP协议，文件系统，压缩解压缩等功能；其二，内部抽象的连接，设备，数据操作流，存储等。

### 2.3.4 Application

该层提供了通用业务应用开发的框架，基于此可方便扩展出不同业务场景下的多任务应用程序。

# 3. 概要设计

系统启动后先进行boot引导，如果boot阶段不需要特别操作则可以进一步引导ap程序运行以实现所需要的业务功能。

## 3.1 boot程序

位于 \prj\boot

boot主要用于在需要时对参数进行修改及串口升级等，大多数情况下会直接引导ap程序的运行。

## 3.2 ap程序

ap实现了主要的业务逻辑，基于以下各子模块：

### 3.2.1 Hardware

——STM32F2 MCU平台驱动库位于 \thirdparty\ST\_Libraries

为了实现平台无关，将会经过BSP的封装后才会调用。

### 3.2.2 OS

——RTOS的抽象接口位于 \src\os\kernel

在此定义了系统中所需要一些OS API，这些API具体的由BSP进行实现。如xtos\_thread.c 定义了任务相关API，xtos\_time.c 定义了时间相关API等。

### 3.2.3 BSP/Driver

——平台相关位于 \ bsp\platform\stm32f207

——单板相关位于 \bsp\broad\cc09

——Driver相关抽象接口位于 \src\drv

BSP分为platform与board 两部分，分别是用于多平台扩展及多系列单板扩展。

Xtbsp\_cc09.c 实现了单板BSP的主要硬件驱动功能

bsp\broad\cc09\xtdrv 实现了具体的Driver API

bsp\broad\cc09\xtos 实现了具体的OS API

### 3.2.4 Middleware

分为通用及系统抽象两部分

#### 3.2.4.1 TCP/IP － lwip

TCP／IP 协议棧位于 thirdparty\lwip\_v1.3.2

实现了以太网功能。

#### 3.2.4.2 FS － yaffs2

File system 位于 thirdparty\yaffs2

实现了基于nand flash的文件系统，带有较强的硬件容错能力。

#### 3.2.4.3 Zlib

位于 thirdparty\zlib-1.2.3

实现了数据的压缩及解压缩库的功能，可以减少大数据传输时带来的流量损失。

#### 3.2.4.4 Connection

位于 src\mdl\connection

抽象的连接层，是对具体或抽象的连接通道的抽象对象。

具体连接，如 conn\_tcpc.c 是对tcp 客户端连接的抽象。

抽象连接，如 conn\_msgq.c是对内部消息通信连接的抽象。在太阳能项目中对于CCU（LCU）二合一功能使用了消息连接来统一对LCU的处理。

#### 3.2.4.5 Device

位于 src\mdl\device

抽象的设备层，是对挂载于不同连接的设备的抽象对象。根据不同设备的操作特性，或者通信协议进行了封装。

如 src\mdl\device\dev\_solar 定义了太阳能控制器的一些操作

#### 3.2.4.6 FS

位于 src\mdl\fs

这是对数据存储流的抽象

数据流抽象API位于 lib\x11base

#### 3.2.4.6 Data manager

位于 src\mdl\data\_manager

这是对数据存储的进一步抽象，方便数据访问和修改，分为CCU数据管理，LCU数据管理，CCU（LCU）二合一数据管理。

#### 3.2.4.7 Protocol

位于 src\mdl\protocol

这是对基本通信协议的结构定义与操作等。

### 3.2.5 Application

Ap 应用框架是基于多任务与消息队列来进行通信的，所有的消息都需要经过 ap\_framework的转发处理，基于扩展的考虑，每个ap可以分为两部分：通用部分位于 src\ap ；特定部分位于 prj\blade\ap。

#### 3.2.5.1 ap framework

位于 \src\ap

application 框架， 其中定义了每个ap及其可以处理的消息对象id。

#### 3.2.5.2 ap comm

位于 src\ap\ap\_comm

ap comm 主要负责与RMS建立连接，接收或者回复命令，对通信相关的加解密处理也在这里进行。

#### 3.2.5.3 ap plc

位于 src\ap\ap\_plc

ap plc 负责所有与LCU终端相关的操作，开关灯，查询数据等。

#### 3.2.5.4 ap dev

位于 src\ap\ap\_dev

ap dev主要负责485外部接口的通信处理，CCU（LCU）二合一中对太阳能控制器的具体操作在此进行。

#### 3.2.5.5 ap misc

位于 src\ap\ap\_misc

ap misc主要负责其他系统功能，比如任务，升级等

## 3.3典型数据流

以GPRS下控制太阳能控制器开灯为例

**任务ap\_comm**

任务主循环apCommEventLoop()🡪

读取GPRS模块数据ConnRead()🡪

协议解析ProAesParsePktInBuf()🡪

协议解密IotAesDecryptEcb() 🡪

协议校验UTCheckProData()🡪

协议命令分派处理apCommReqDispathCmd()🡪发送消息

**任务ap\_framework**

任务主循环apFrmEventLoop() 🡪

框架命令处理apFrmOnDispathCmd()🡪

框架具体分派命令apFrmDispathCmd()🡪发送消息

**任务ap\_plc**

任务主循环apPLCEventLoop()🡪

plc 终端节点命令处理apPLCOnNodeCmd()🡪

终端命令apPLCDoSubCmd()🡪

终端命令发收apPLCWriteAndRead()🡪

zigbee通信发收NmWirelessZigbeeWriteAndRead()🡪

通信协议加密IotAesEncryptEcb()🡪

通信协议打包ProCmdToPack()🡪

zigbee通信模块发送ConnWrite()🡪

zigbee通信模块接收ConnRead()🡪

通信协议解析ProParsePktInBuf()🡪

通信协议解密IotAesDecryptEcb()🡪发送消息

**任务ap\_framework**

任务主循环apFrmEventLoop() 🡪

框架命令处理apFrmOnDispathCmd()🡪

框架具体分派命令apFrmDispathCmd()🡪发送消息

**任务ap\_comm**

任务主循环apCommEventLoop()🡪

消息命令处理apCommOnRawSendCmd()🡪

通信协议加密IotAesEncryptEcb()🡪

GPRS通信模块发送ConnWrite()

# 4. PC工具

## 4.1 AES dll

位于 util\iotAESLib

RMS 使用的加解密动态链接库源码

## 4.2 iotReleaseTool

位于 util\iotReleaseTool

版本文件生成工具源码