**嵌入式开放实验报告--简单的HTTP服务器**

**15组**

目录

[1. 系统总体介绍 1](#_Toc454447865)

[2. 信息交互 2](#_Toc454447866)

[3. 程序流程图及函数调用关系 3](#_Toc454447867)

[4. 函数功能表 9](#_Toc454447868)

[5. 重要函数输入输出 10](#_Toc454447869)

[6. 库文件说明 11](#_Toc454447870)

[7. 实现功能 12](#_Toc454447871)

## ****系统总体介绍****

功能描述：本次实验使用了基于Cortex-M4内核的TM4C1294XL实验板,以及CC3100 WiFi扩展子板、SensorHub等外设，实现了HTTP服务器的基本功能，用户可以通过浏览器和M4板进行交互，从而远程控制LED灯的开关、呼吸，并可获取温度湿度等信息。系统的总体框图如图1所示。

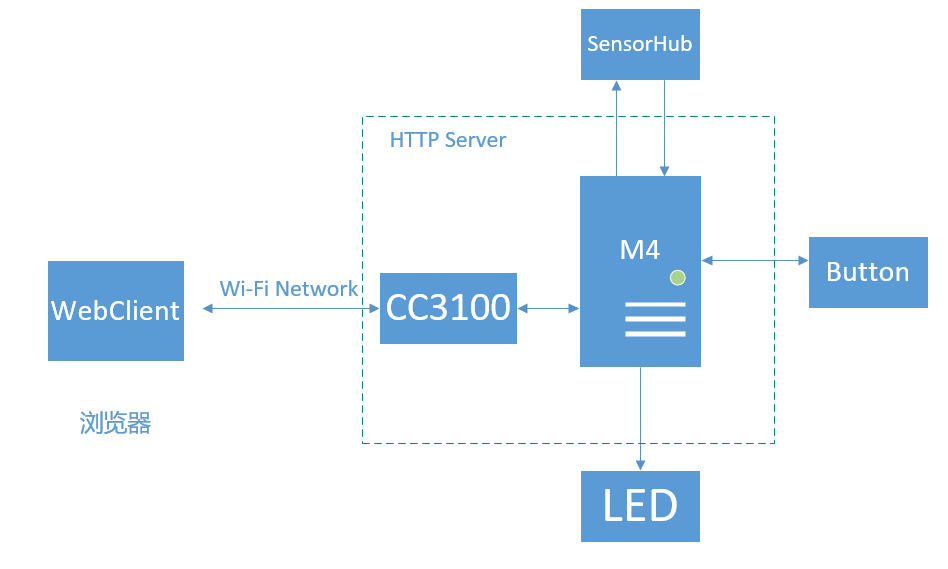


图1 系统总框图

开发环境：本次编程工作基于CCS开发平台，使用了TI提供的底层操作库，TI-RTOS等，主要使用C语言在应用层上进行服务器和硬件控制进行编程，使用javascript等对前端网页进行编写。软件层示意图如图2所示。

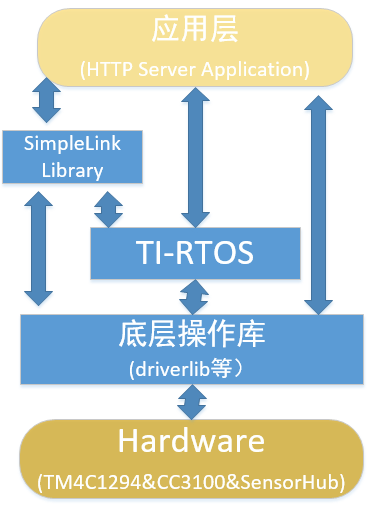


图2 系统软件层

## ****信息交互****

2.1 TM4C1294XL和CC3100之间：主板通过GPIOinWrite连接CC3100的CC3100的nHIB端口对PC6写高低电平的方式使能CC3100。主板GPIOintrupt(PM3)连接CC3100的CC\_IRQ端口，控制打开或关闭CC3100的中断。当CC3100需要服务的时候，通知主板开启中断。主板和CC3100之间的信息交换基于SPI通信协议。SDI(PD1\_SSI2XDAT0)，SDO(数据输-PD0\_SSIXDATA1 )，SCLK (PD3\_SSI2CLK)，CS(片选- GPIO PH2)，主板的SSI2用作SPI模式。

如图3和图4所示。

|  |  |  |
| --- | --- | --- |
| **CC3100** | **TM4C1294** | **功能** |
| nHIB | GPIO PC6 | Disable/enable CC3100 |
| CC\_IRQ | GPIO PM3 | 打开或关闭CC3100中断 |
| SPI | SSI2 | 数据输入、输出；时钟；片选 |

图3 管脚对应关系

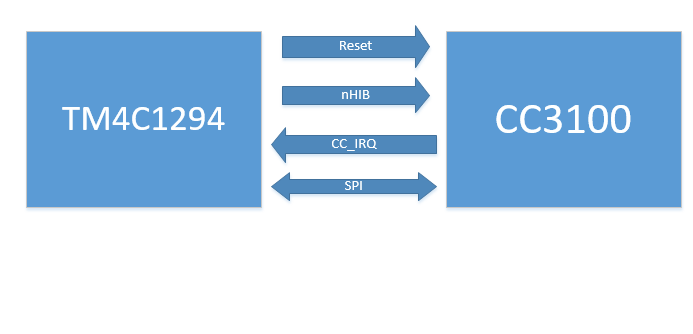


图4 TM4C1294与CC3100的连接

2.2 服务器前后端：建立起网络连接通路后，把网页放到服务器上，即可通过浏览器访问服务器，并进行远程交互。服务器会监听用户请求，收到请求后，由一个函数来处理监听到的事件，执行相应任务，做出响应。图5中以GETtemperature和LEDon两个功能为例展示了信息交互的过程。

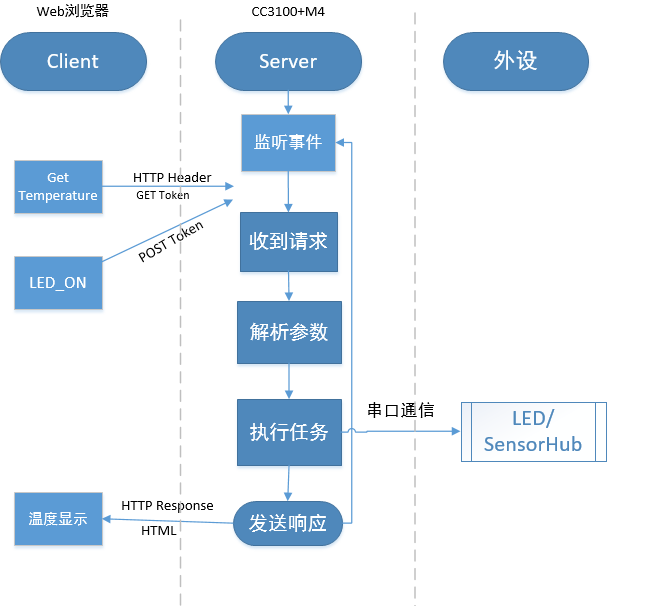
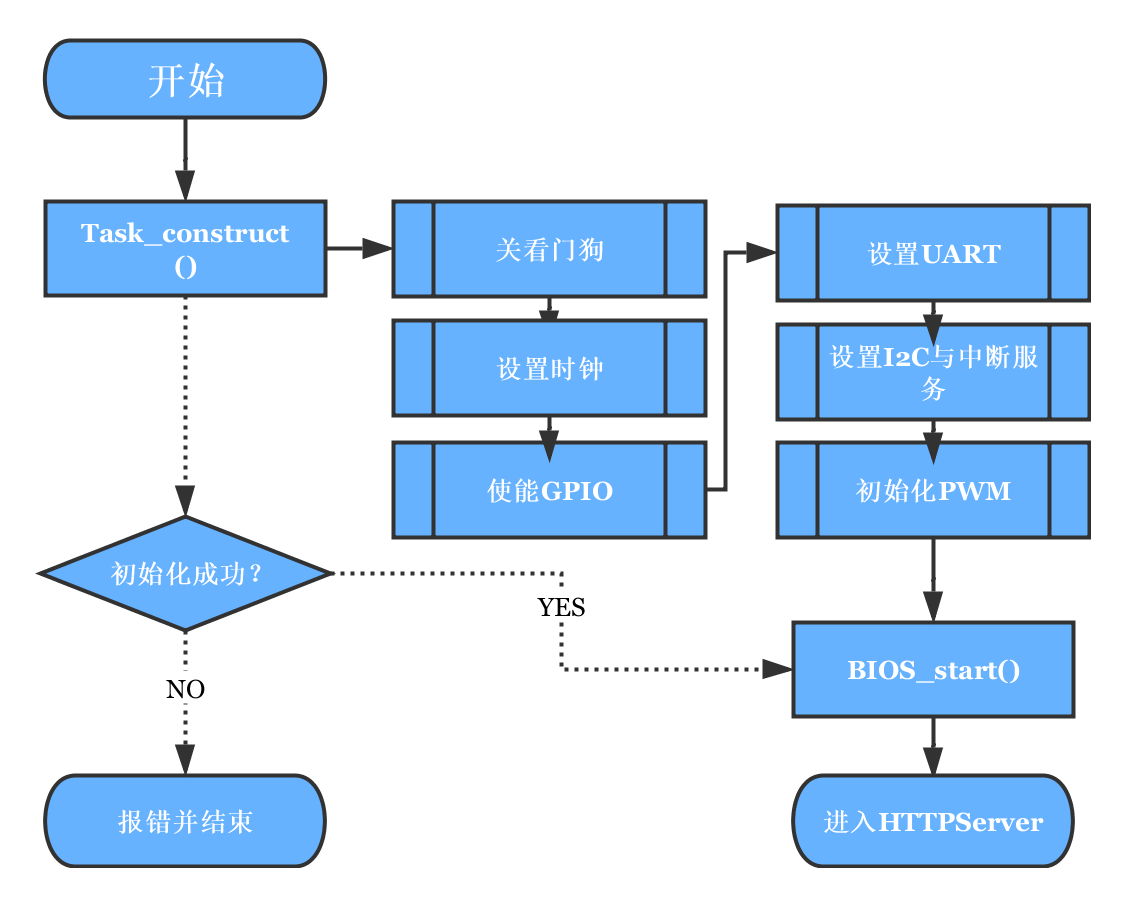


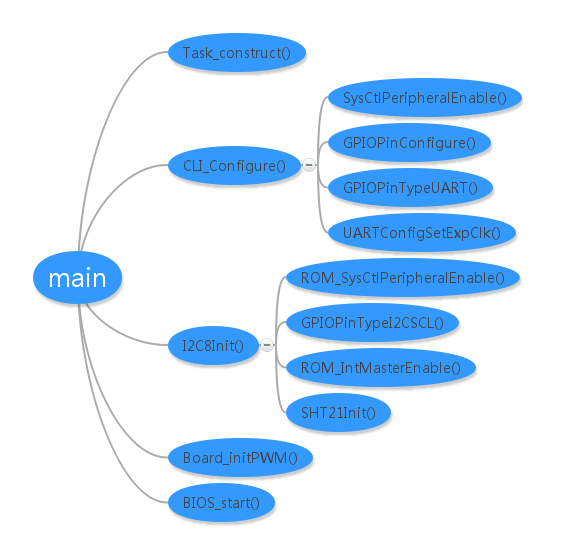
图5 前后端信息交互示意图

## ****程序流程图及函数调用关系****

* 1. main函数



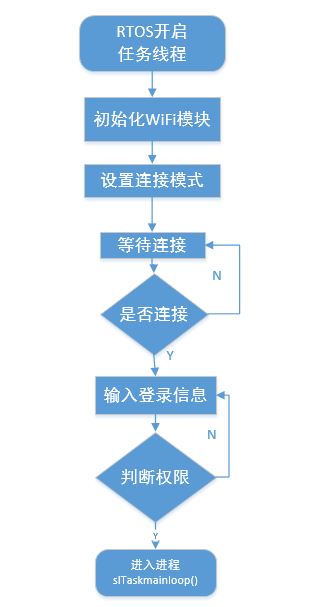
main函数程序



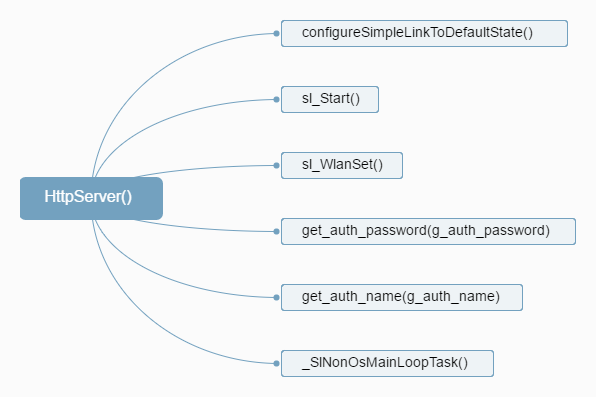
main()中调用的函数关系图

main函数里面主要是进行了初始化操作，对UART（用于串口通信），GPIO（控制LED跑马灯），PWM（控制呼吸灯），I2C（温湿度传感器与M4交互）和中断进行了使能与基础设置。除此之外，因为本次项目使用了rtos，故我们使用了task\_construct()函数来创建服务器初始化进程

* 1. httpserver()



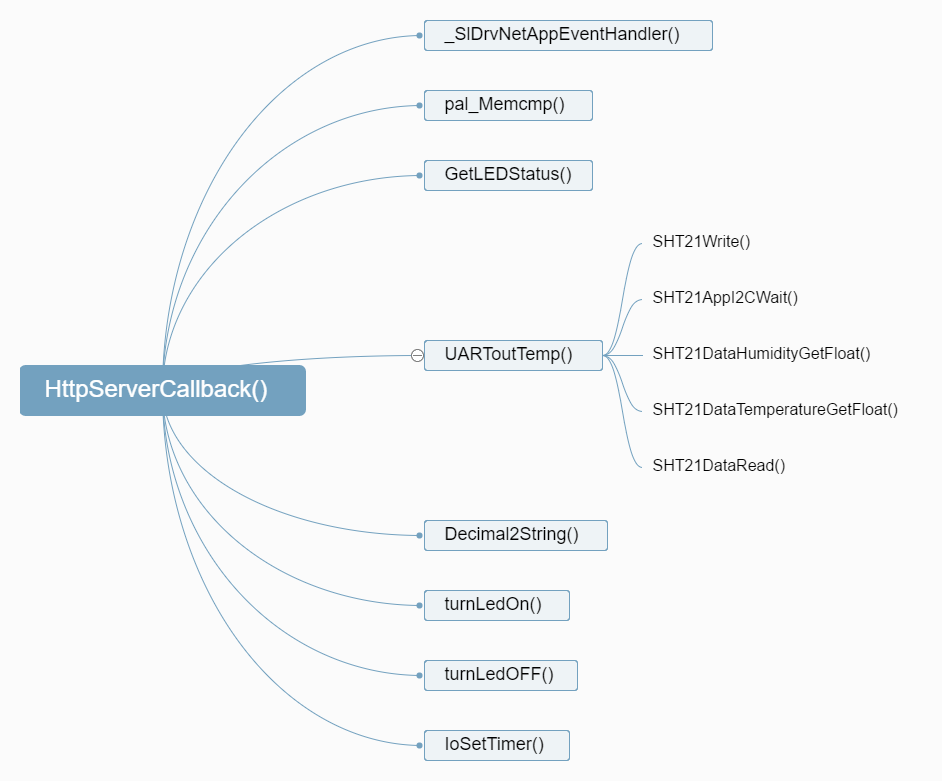
httpserver()流程图



httpserver()调用的函数

httpserver函数主要是对CC3100板进行了初始化和设置，在我们的代码里，将其设置为了AP模式，此外还对是否有设备接入AP进行检测，并将一些调试信息反馈至串口。

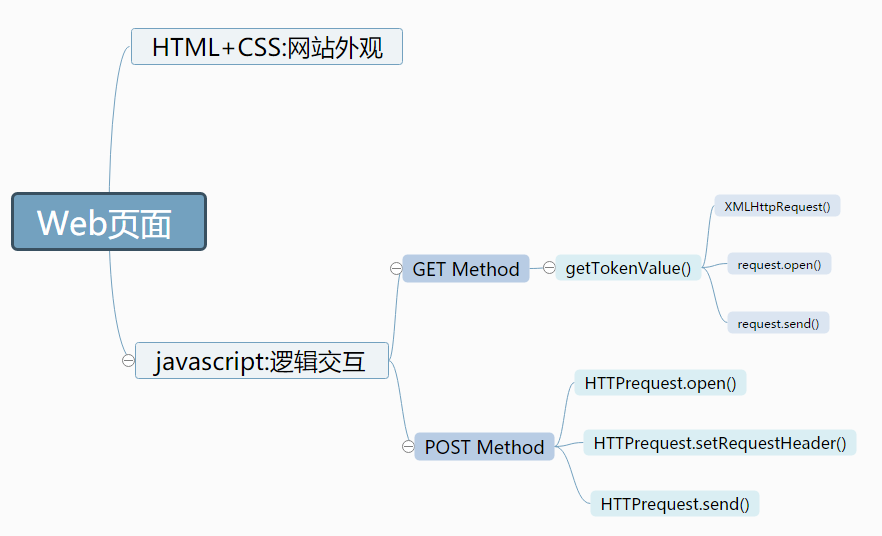
* 1. HttpServerCallback()



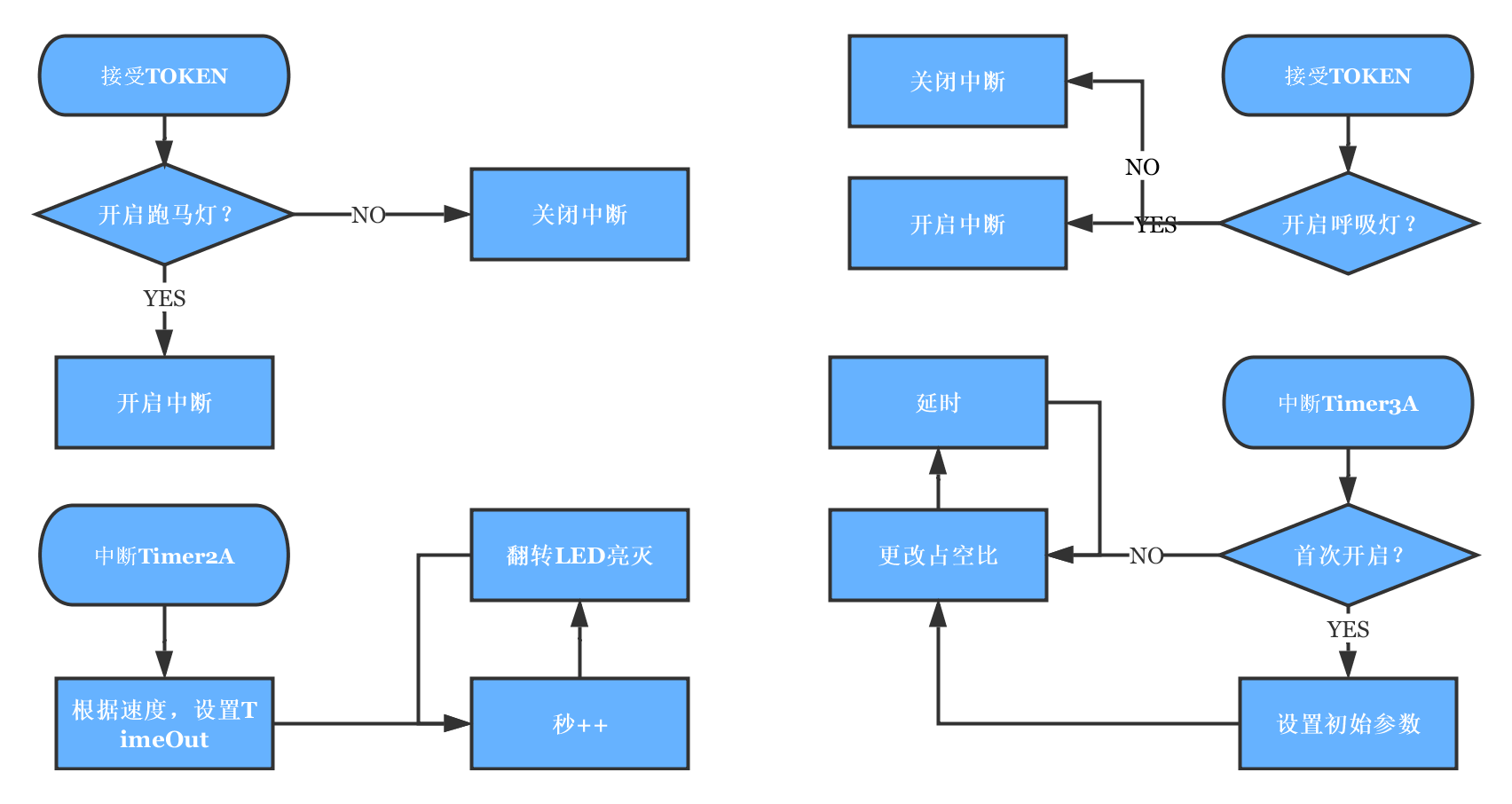
HttpServerCallback()调用的函数

HttpServerCallback是

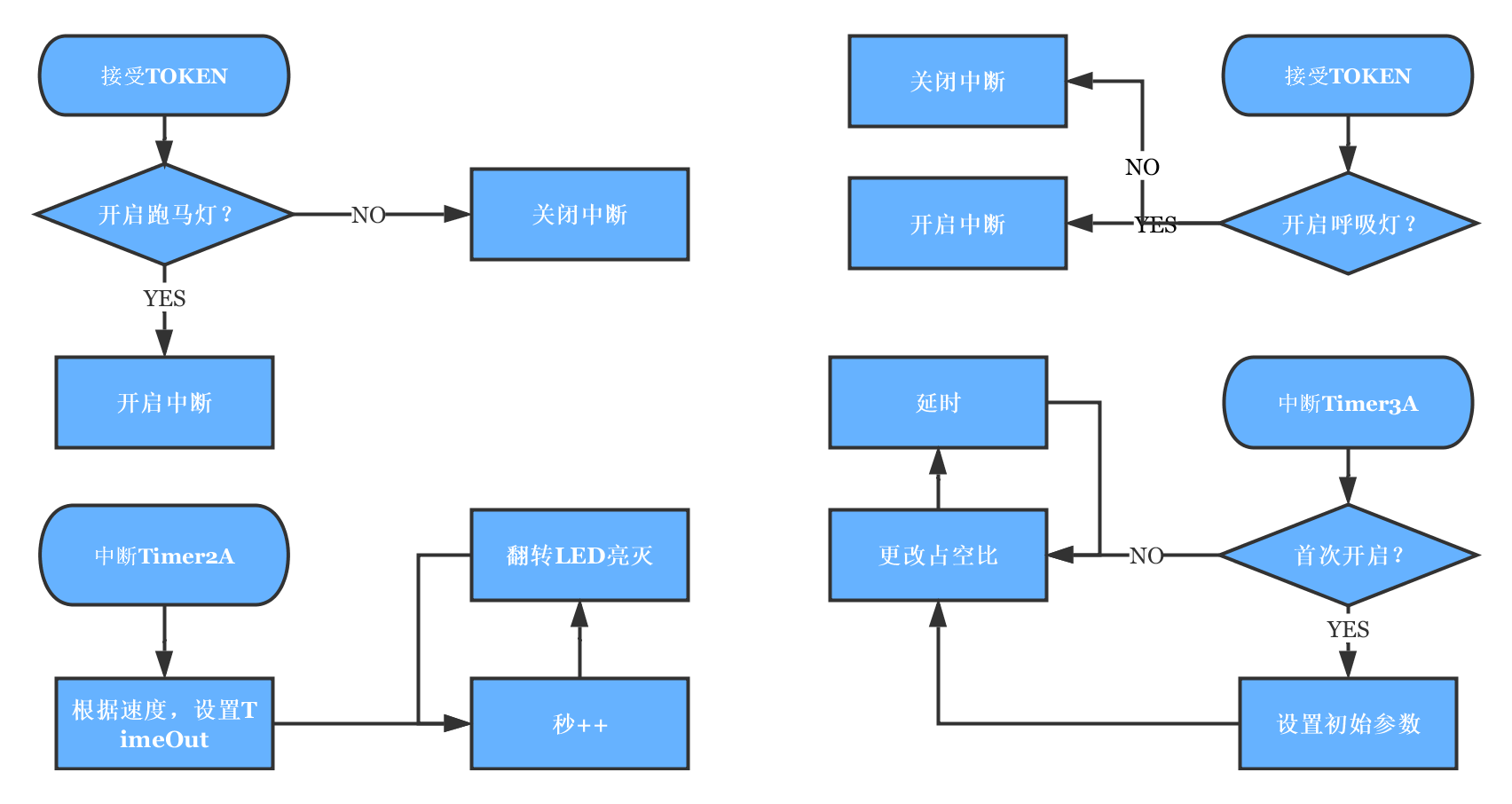
* 1. web前端



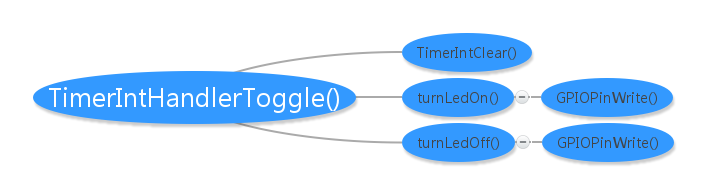
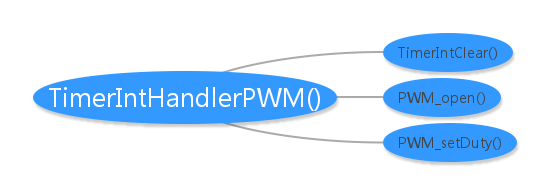
* 1. LED控制



LED开关和闪烁

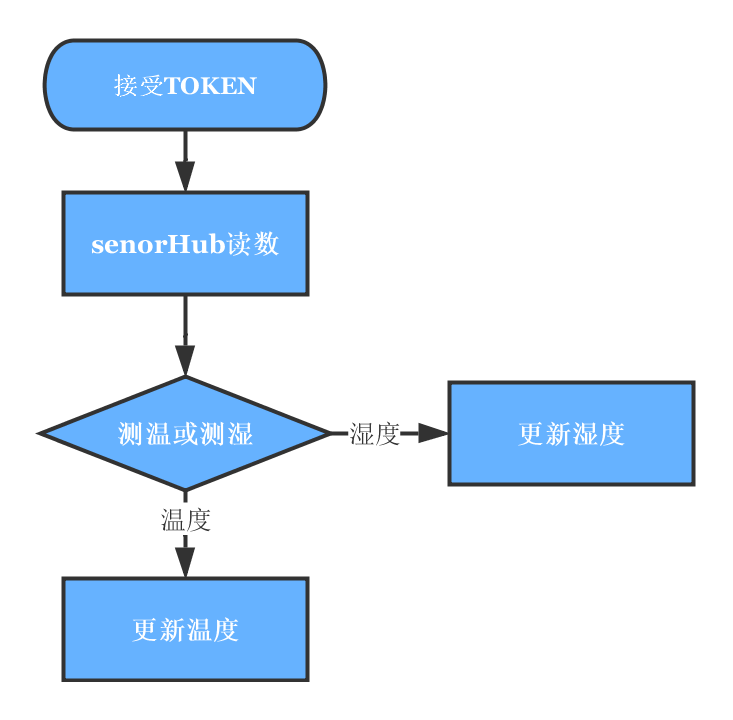


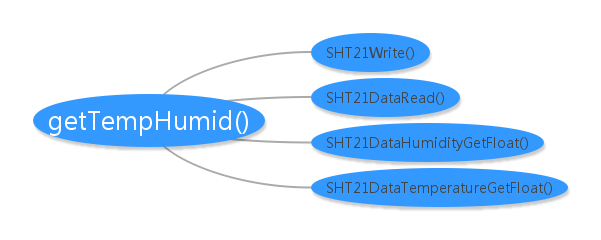
呼吸灯



Web段发出对LED的控制指令后，后端会打开或关闭相应的中断，或者是对TIMER等等timeout值进行设置，以达到控制频率的作用。

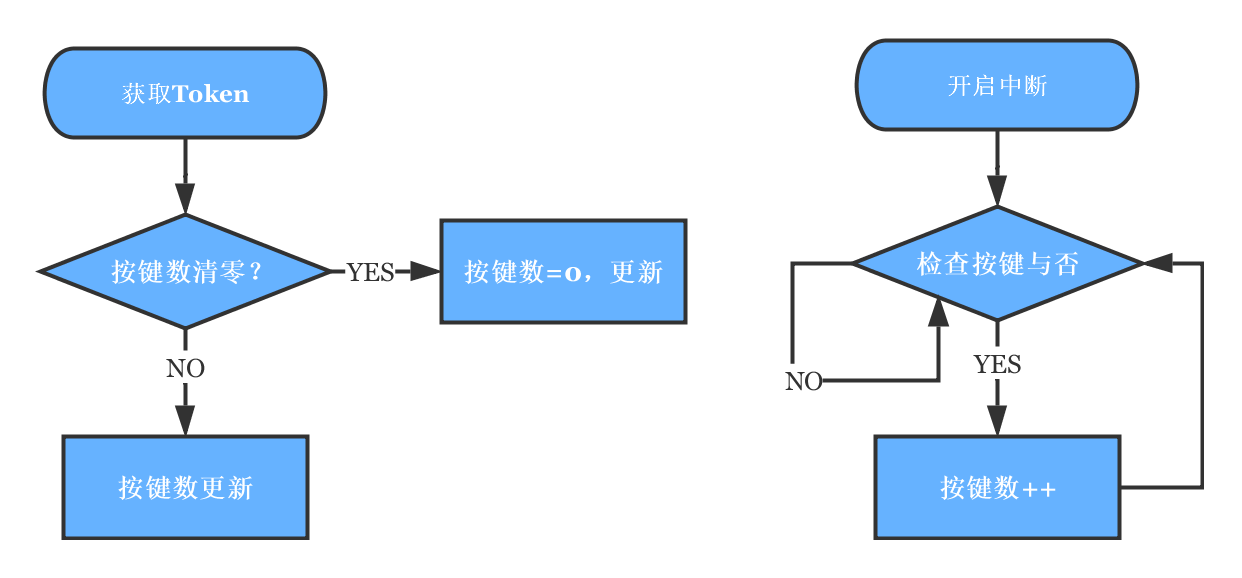
* 1. SensorHub

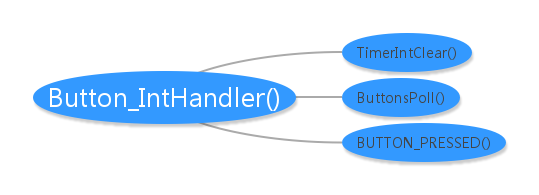




每次web端接受指令后，后端会通过I2C访问传感器读取相应的值并反馈。

3.7 按键信息获取

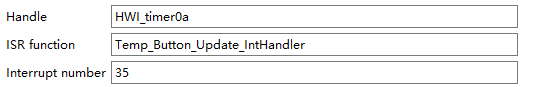


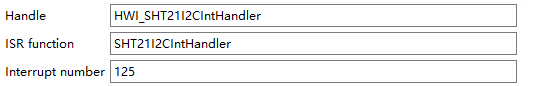


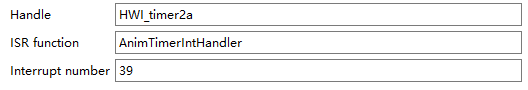
与之前类似，按键值的获取也是由web端触发的。

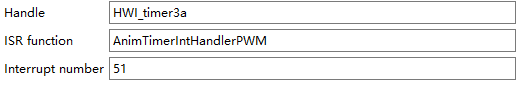
3.8中断设置

本次项目直接使用了rtos的cfg文件提供的gui进行中断向量表的设置。









## ****函数功能表****

后端

|  |  |
| --- | --- |
| 函数名 | 函数功能 |
| configureSimpleLinkToDefaultState | 对CC3100进行配置，使其工作在默认状态。 |
| get\_auth\_name | 获取用户输入的用户名 |
| get\_auth\_password | 获取用户输入的密码 |
| getTempHumid | 获取温度和湿度 |
| SHT21Write | SHT21传感器开启测试 |
| ROM\_SysCtlDelay | 延时 |
| SHT21DataRead | 通过I2C总线从Sensorhub处获取数据 |
| SHT21DataHumidityGetFloat/ SHT21DataTemperatureGetFloat | 获取最新湿度/温度浮点数形式数据 |
| CLI\_Write | 输出信息到串口工具 |
| Task\_construct | 在执行BIOS后，开启一个任务线程 |
| initializeAppVariables | 初始化服务器部分变量 |
| stopWDT() | 关看门狗 |
| initClk()/initLEDs()/I2C8Init()/IoInit()/Board\_initPWM()/CLI\_Configure() | 初始化时钟脉冲/LED灯/I2C/IO设备/PWM/串口输出 |
| BIOS\_start() | 开启RTOS |
| PinoutSetDevice | 配置管脚与外设关联 |
| HttpServer | 开启并配置HTTP服务器 |
| set\_port\_number | 设置HTTP端口号 |
| HttpServerCallback | 处理HTTP监听到的事件 |
| GetLEDStatus | 获取LED灯的状态 |
| pal\_Memcmp | 对比字符串，确认参数 |
| pal\_Memcpy | 拷贝字符串 |
| Decimal2String | 十进制转换为字符串 |
| turnLedOn/ turnLedOff | 打开/关闭LED灯 |
| IoSetTimer | 关闭或使能并设置Timer,控制相应中断 |
| IOEnablePWM | 使能/关闭PWM |
| SimpleLinkNetAppEventHandler | 处理IP地址及动态分配相关的事件 |
| UpdateButtons | 记录按钮被按的次数 |
| AnimTimerIntHandlerPWM | 处理PWM（中断号51，对应timer3A） |
| AnimTimerIntHandler | 处理跑马灯（中断号39，对应timer2A） |
| Timer0IntHandler | 处理Time0中断 |
| Temp\_Button\_Update\_IntHandler | 处理按钮中断（中断号35，对应timer0A） |
| SHT21I2CIntHandler | SHT21传感器中断处理（中断号125，对应I2C8） |
| ConfigureTimer0 | 初始化Timer |
| ConfigureADC0 | 初始化ADC |

前端(javascript)

|  |  |
| --- | --- |
| 函数名 | 功能 |
| loadPage | 异步加载页面 |
| setRequestHeader | 设置请求头文件 |
| SetFormDefaults | 设置默认值 |
| getTokenValue | 异步获取参数值 |
| send | 向服务器发送数据 |
| led\_on/led\_off/led\_breath\_on/ led\_toggle\_on | 发送开灯/关灯/呼吸/闪烁请求 |
| temperatureC/humid | 请求温度/湿度数据 |
| speedSet | 闪烁/呼吸速度设置 |

## ****重要函数输入输出****

HttpServerCallback

|  |  |  |  |
| --- | --- | --- | --- |
| Item | Name | Type | Description |
| Inputs | \*pEvent | SlHttpServerEvent\_t | Http server event object |
| \*pResponse | SlHttpServerResponse\_t | Http server response object |
| Outputs | void | -- | -- |
| Parameters | ulSpeed | int | Get speed value |
| humid | int | Get humidity value |
| temp | int | Get temperature value |
| GET\_token | \_u8 | Pass GET token |
| POST\_token | \_u8 | Pass POST token |

getTempHumid

|  |  |  |  |
| --- | --- | --- | --- |
| Item | Name | Type | Description |
| Inputs | \* humid | int | humidity value |
| \* temp | Int | temperature value |
| Outputs | void | -- | -- |
| Parameters | len | int | Store length of variable |
| ptr | \_u8 | Store a string |
| fHumidity | float | Store humidity |
| fTemperature | float | Store temperature |

HttpServer

|  |  |  |  |
| --- | --- | --- | --- |
| Item | Name | Type | Description |
| Inputs | none | -- | -- |
| Outputs | void | -- | -- |
| Parameter | SecType | \_u8 | AP securety type |
| retVal | \_i32 | to check the state |
| mode | \_i32 | Set the operating mode |
| g\_Status | \_u32 | Check the status of connection |
| g\_auth\_password | \_u8 | Store the authentic password |
| g\_auth\_name | \_u8 | Store the authentic name |

turnLedOn/ turnLedOff

|  |  |  |  |
| --- | --- | --- | --- |
| Item | Name | Type | Description |
| Inputs | LedNUM | char | Name of the LED to open of turn off |
| Outputs | void | -- | -- |

UpdateButtons

|  |  |  |  |
| --- | --- | --- | --- |
| Item | Name | Type | Description |
| Inputs | none | -- | -- |
| Outputs | void | -- | -- |
| Parameter | ui32Timeout | uint8\_t | Check the current debounced state of the buttons |
| ui8ButtonsChanged | uint8\_t | Check if either button has been pressed |
| g\_ui32SW1Presses | Volatil unsigned long | Record the numbers |

IoSetTimer

|  |  |  |  |
| --- | --- | --- | --- |
| Item | Name | Type | Description |
| Inputs | ui32SpeedPercent | unsigned long | Set frequency of PWM and toggle |
| Outputs | void | -- | -- |
| Parameter | ui32Timeout | unsigned long | Time out value of timer2A and timer3A |

IoSetAnimationSpeed

|  |  |  |  |
| --- | --- | --- | --- |
| Item | Name | Type | Description |
| Inputs | ui32Speed | unsigned long | The speed from web |
| Outputs | void | -- | -- |
| Parameter | g\_ui32AnimSpeed | unsigned long | Speed after checking |

AnimTimerIntHandler (timer2A)

|  |  |  |  |
| --- | --- | --- | --- |
| Item | Name | Type | Description |
| Inputs | Void | - | - |
| Outputs | void | -- | -- |
| Parameter | g\_ui32timer2count | unsigned long | Count timer |

AnimTimerIntHandlerPWM (timer3A)

|  |  |  |  |
| --- | --- | --- | --- |
| Item | Name | Type | Description |
| Inputs | Void | - | - |
| Outputs | void | -- | -- |
| Parameter | pwm1 | PWM\_Handle | Set PWM |
| pwmPeriod | uint16\_t | Pwm period |
| duty | uint16\_t | PWM duty |
| dutyInc | uint16\_t | PWM duty increase/decrease amount |

Temp\_Button\_Update\_IntHandler(timer0A)

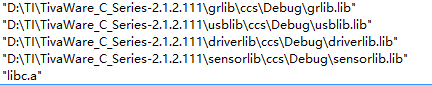
|  |  |  |  |
| --- | --- | --- | --- |
| Item | Name | Type | Description |
| Inputs | Void | - | - |
| Outputs | void | -- | -- |
| Parameter | g\_ui32TimerIntCount | unsigned long | the number of times the handler has been called |

UpdateButtons

|  |  |  |  |
| --- | --- | --- | --- |
| Item | Name | Type | Description |
| Inputs | Void | - | - |
| Outputs | void | -- | -- |
| Parameter | g\_ui32SW1Presses | unsigned long | Times button has been pressed |

## ****引用文件说明****

引用的库：



|  |  |
| --- | --- |
| 引用的文件 | Description |
| gpio.h | Defines and Macros for GPIO API. |
| interrupt.h | Driver for the NVIC Interrupt Controller. |
| pin\_map.h | Mapping of peripherals to pins for all parts. |
| sysctl.h | Prototypes for the system control driver. |
| uart.h | Driver for the UART. |
| ssi.h | Prototypes for the Synchronous Serial Interface Driver. |
| timer.h | Prototypes for the timer module |
| hw\_sht21.h | Prototypes for the SHT21 accelerometer driver. |
| i2cm\_drv.h | Prototypes for the interrupt-driven I2C master driver. |
| cli\_uart.c | TM4C1294 launchpad application uart interface implementation |
| board.c | TM4C1294 launchpad configuration |
| buttons.c | Evaluation board driver for push buttons. |
| io.c | I/O routines |
| EK\_TM4C1294XL.c | Setting up the board specific items for the EK\_TM4C1294XL board. |

## ****实现功能****

最终网页效果如下图所示，点击不同按钮即可触发不同事件。

