TS100开源文件说明

——控制管理

目录

一、系统状态

定义在CTRL.c 的第26行

```
25
26  u8 gCtrl_status = 1;
27  u16 gHt_flag = 0;
28  vs16 gTemp_data = 250;//25℃
```

全局变量gCtrl_status表示状态，其值有8种状态，分别为：IDLE(待机) ,THERMOMETER(温度计), TEMP_CTR(温控),WAIT(休眠),TEMP_SET(设置), CONFIG (配置),MODE_CNG(备用),ALARM(警告);

定义在Ctrl.c 67行

```
66  ************************************************
67  void Set_CtrlStatus(u8 status)
68  {
69      gCtrl_status = status;
70  }
```

各状态之间的切换是通过Set_CtrlStatus(u8 status)函数实现的

定义在Ctrl.c 57行

```
56  ************************************
57  u8 Get_CtrlStatus(void)
58  {
```

读取当前状态是通过Get_CtrlStatus(void)实现

二、TS100状态

通过对状态变量的认识，以及切换读取状态函数，很容易达到自己设置各种状态的来回交替，下面详细分析一下TS100的状态控制。

1. 初始化状态

```
48      Clear_Screen();
49      Init_Gtime();
50      APP_Init();
51
```

调用此函数，Main.c中50行

```
114 ************************************************
115 void APP_Init(void)
116 {
117     int rev;
118
119     HEATING_TIMER = 0;   初始化计时器
120     UI_TIMER = 0;   初始化计时器
121
122     rev = Read_Vb(0); 读取电压值
123     if(rev == 0)   Set_CtrlStatus(ALARM); 电压过高，进入报警状态
124     else if(rev >= 4) {    U盘配置状态
125         Set_LongKeyFlag(1);
126         Set_CtrlStatus(CONFIG);
127     } else {        规定范围内电压，进入正常初始状态
128         Set_CtrlStatus(IDLE);
129         G6_TIMER = device_info.idle_time;
130     }
131
132 }
133
134 /************************************************
```

定义在UI.c 115行

定义在CTRL.c 第203行

```
201 返回参数: NULL
202 ************************************
203 void Status_Tran(void)//状态转换
204 {
```

初始状态确定后，进入状态转此换函数，该函数为一个switch组成，分各种状态，不同的处理变换过程

二、TS100状态

2. IDLE状态

定义在CTRL.c 211行到247行

该状态为待机状态，在开机后显示，待机显示，电压过低会返回到待机界面，休眠一段时间后也会返回到待机界面。

```
211     case IDLE:
212         switch(Get_gKey()) {  获取按键
213         case KEY_V1: 短按按键
214             if(gIs_restartkey != 1) {  是否是软件重启，回到初始状态
215                 if(Read_Vb(1) < 4) {  判断电压是否正常
216                 Set_CtrlStatus(TEMP_CTR); 设置温控状态
217                 init_waitingtime     = 0; 初始化等待计数
218                 TEMPSHOW_TIMER  = 0;初始化计时器
219                 UI_TIMER = 0;
220                 G6_TIMER = 0;
221                 }
222             }
223             break;
224         case KEY_V2:  单按B键
225             if(gIs_restartkey != 1) {  是否是软件重启，回到初始状态
226                 Set_CtrlStatus(THERMOMETER); 进入温度计模式
227                 UI_TIMER = 0;
228                 Set_LongKeyFlag(1); 设置长按标志
229             }
230             break;
231         case KEY_CN|KEY_V3: AB同时按键 无操作
232             break;
233         }
234         if(gIs_restartkey && (KD_TIMER == 0)) { 初始化软件重启按键状态
235             gIs_restartkey = 0;
236             Set_gKey(NO_KEY);
237         }
238         if(Read_Vb(1) == 0){  电压异常
239             if(Get_UpdataFlag() == 1) Set_UpdataFlag(0);
240             Set_CtrlStatus(ALARM);  进入报警状态
241         }
242         if(gPre_status != WAIT && gPre_status != IDLE){  屏幕保护自动黑屏
243             G6_TIMER = device_info.idle_time;
244             Set_gKey(NO_KEY);
245             gPre_status = IDLE;
246         }
247         break;
```

二、TS100状态

3. TEMP_CTR状态

定义在CTRL.c 248行到309行

温控状态，在升温、降温、恒温等过程的状态是烙铁工作的主要状态。

```
248     case TEMP_CTR:
249         switch(Get_gKey()) {
250         case KEY_CN|KEY_V1:
251         case KEY_CN|KEY_V2:  长按任意键
252             Set_HeatingTime(0); 加热时间设置为0,即停止加热
253             Set_CtrlStatus(TEMP_SET); 进入设置模式
254             HEATING_TIMER        = 0;
255             EFFECTIVE_KEY_TIMER = 500; 退出设置模式时间初始化
256             break;
257         case KEY_CN|KEY_V3: AB双键同时按
258             Set_HeatingTime(0); 停止加热
259             Set_LongKeyFlag(0); 设置按键标志，不能长按键
260             Set_CtrlStatus(IDLE); 返回初始状态，即待机界面
261             gPre_status = TEMP_CTR;
262             gIs_restartkey = 1;
263             KD_TIMER = 50; //
264             break;
265         }
266
267         if(Read_Vb(1) >= 4) {读取电压,电压低于设置最低值返回待机界面
268             Set_HeatingTime(0); 停止加热
269             Set_LongKeyFlag(0);
270             Set_CtrlStatus(IDLE); 返回到待机界面
271             gPre_status = TEMP_CTR;
272             gIs_restartkey = 1;
273             KD_TIMER = 50; // 2秒
274         }
275
276         wk_temp = device_info.t_work;
277         if(HEATING_TIMER == 0) {
278             gTemp_data     = Get_Temp(wk_temp);
279             heat_timecnt   = Heating_Time(gTemp_data,wk_temp); 根据PID算法计算加热时间
280             Set_HeatingTime(heat_timecnt); 设置加热时间
281             HEATING_TIMER = HEATINGCYCLE; 赋值加热周期
282         }
283         if(Get_HeatingTime() == 0) { 不加热时，强制检测加热时间
284             HEATING_TIMER = 0;
```

二、TS100状态

3. TEMP_CTR状态

定义在CTRL.c 248行到309行

温控状态，在升温、降温、恒温等过程的状态是烙铁工作的主要状态。

```
285         }
286
287
288         mma_active = Get_MmaShift(); 获取烙铁是否移动标志
289         if(mma_active == 0) {  烙铁静止
290             if(init_waitingtime == 0) { 开始休眠倒计时
291                 init_waitingtime    = 1;
292                 ENTER_WAIT_TIMER = device_info.wait_time;
293             }
294             if((init_waitingtime != 0) && (ENTER_WAIT_TIMER == 0)) { 结束休眠倒计时
295                 gHt_flag       = 0;
296                 UI_TIMER      = 0;
297                 Set_HeatingTime(0);
298                 Set_gKey(0);
299                 G6_TIMER = device_info.idle_time; 以上为初始化相关标志以及时间
300                 Set_CtrlStatus(WAIT);
301             }
302         } else { 如果烙铁移动初始化移动标志
303             init_waitingtime = 0;
304         }
305         if(Get_AlarmType() > NORMAL_TEMP) {   警报类型由警报状态中确定
306             if(Get_UpdataFlag() == 1) Set_UpdataFlag(0);
307             Set_CtrlStatus(ALARM); 进入警报状态
308         }
309         break;
```

二、TS100状态

4. Wait状态

定义在CTRL.c 310行到353行

休眠状态，系统静止放置一段时间后会进入休眠状态，休眠状态的下温度为休眠温度，当休眠温度大于当前温度时，会保留当前温度为休眠温度。

```
310     case WAIT:
311         wk_temp = device_info.t_standby;
312         if(device_info.t_standby > device_info.t_work) { //休眠温度比工作温度高
313             wk_temp = device_info.t_work; 休眠温度比工作温度高,设置休眠温度为工作温度
314         }
315         if(HEATING_TIMER == 0) {
316             gTemp_data     = Get_Temp(wk_temp);获取当前温度
317             heat_timecnt   = Heating_Time(gTemp_data,wk_temp); 根据PID计算加热时间
318             Set_HeatingTime(heat_timecnt); 设置加热时间
319             HEATING_TIMER = 30;
320         }
321
322         if(Read_Vb(1) >= 4) {  电压过低,停止加热,返回待机界面
323             Set_HeatingTime(0);
324             Set_LongKeyFlag(0);
325             Set_CtrlStatus(IDLE);
326             G6_TIMER = device_info.idle_time;
327             gPre_status = WAIT;
328             gIs_restartkey = 1;
329             KD_TIMER = 50; // 2秒
330         }
331
332         if(G6_TIMER == 0) { 当待机时间到了后进入待机
333             Set_HeatingTime(0);
334             Set_LongKeyFlag(0);
335             gIs_restartkey = 1;
336             KD_TIMER = 200; // 2秒
337             gPre_status = WAIT;
338             Set_CtrlStatus(IDLE); 返回待机
339         }
340
```

二、TS100状态

4. Wait状态

定义在CTRL.c 310行到353行

休眠状态，系统静止放置一段时间后会进入休眠状态，休眠状态的下温度为休眠温度，当休眠温度大于当前温度时，会保留当前温度为休眠温度。

```
341         mma_active = Get_MmaShift();读取烙铁移动状态
342         if(mma_active == 1 || Get_gKey() != 0) {  当烙铁移动的时候,返回温控状态
343             UI_TIMER       = 0;
344             G6_TIMER       = 0;
345             init_waitingtime = 0;
346             Set_CtrlStatus(TEMP_CTR); 返回温控状态
347         }
348
349         if(Get_AlarmType() > NORMAL_TEMP) {  警报类型由警报状态中确定
350             if(Get_UpdataFlag() == 1) Set_UpdataFlag(0);
351             Set_CtrlStatus(ALARM);进入警报状态
352         }
353         break;
```

二、TS100状态

5. TEMP_SET设置状态

定义在CTRL.c 355行

温度设置状态，用于设置温度，在温控状态下长按即可进入。

```
354         case TEMP_SET:
355             if(EFFECTIVE_KEY_TIMER == 0) {当进入温度设置的时间耗尽时返回温控模式
356                 Set_CtrlStatus(TEMP_CTR);
357                 TEMPSHOW_TIMER = 0;
358             }
359             break;
```

```
855 void Temp_SetProc(void)    设置过程为此函数，定义在UI.c 855行
856 {
857     u8 theRoll_num = 3;
858     static u16 cont = 0;
859 
860     if(device_info.t_step == 10)  cont = 1; 步进是长按递增型
861 
862     if(EFFECTIVE_KEY_TIMER > 0) {
863         if(gCont != 0) { 设置界面从右往左出来，只出现一次
864             //设置界面从右往左出来
865             gCont = 0;
866             Set_LongKeyFlag(0);
867             Clear_Screen(); 刷屏
868 
869             Show_Set(); 显示温度值
870         }
871         switch(Get_gKey()) {
872         case KEY_V1: 减少,往下翻,当前的显示消失//设置值出现
873             //增加,往下翻,当前的显示消失
874             //设置值出现
875             if(device_info.t_work > gSet_table[1]) { 大于100，可以设置
876                 gTempset_showctrl = 1;
877                 theRoll_num = Roll_Num(device_info.t_step,1); 计算需要滚动的设置值
878                 while(Show_TempReverse(theRoll_num,16,1)); 竖直滚动
879                 device_info.t_work = Calculation_TWork(1);计算减少后的值
880                 if(device_info.t_work < gSet_table[1]) device_info.t_work = gSet_table[1];
                    小于100℃，就等于100℃
```

二、TS100状态

5. TEMP_SET设置状态

```
881                 gTempset_showctrl = 2;
882                 while(Show_TempReverse(theRoll_num,16,1)); 竖直滚动
883                 Show_Triangle(1,0); 显示按键方向
884             }
885             if(device_info.t_work == gSet_table[1])  Show_Triangle(0,1); 显示按键方向
886             break;
887         case KEY_V2: 增加,往上翻,当前的显示消失//设置值出现
888             //减少,往上翻,当前的显示消失
889             //设置值出现
890             if(device_info.t_work < gSet_table[0]) { 实际温度小于400，温度能增加
891                 gTempset_showctrl = 1;
892                 theRoll_num = Roll_Num(device_info.t_step,0);计算滚动字符
893                 while(Show_TempReverse(theRoll_num,16,0)); 竖直滚动
894                 device_info.t_work = Calculation_TWork(0); 计算增加后的值
895                 if(device_info.t_work > gSet_table[0]) device_info.t_work = gSet_table[0];
896                 gTempset_showctrl = 2;                增加后大于400，那么数值就等于400℃
897                 while(Show_TempReverse(theRoll_num,16,0)); 竖直滚动
898                 Show_Triangle(2,0); 显示按键方向
899             }
900
901             if(device_info.t_work == gSet_table[0])  Show_Triangle(0,2); 显示按键方向
902             break;
903         default:
904             break;
905         }
906         if(Get_gKey() > NO_KEY) { 递增型步长的计算
907             if(cont > 0 && EFFECTIVE_KEY_TIMER > 260) {
908                 device_info.t_step = 10 +  cont * 50;
909                 cont++;
910             }
911             EFFECTIVE_KEY_TIMER = 300;
912             Set_gKey(NO_KEY);
913         }
914         if(cont > 0 && EFFECTIVE_KEY_TIMER <= 260) {
915             device_info.t_step = 10;
916             cont = 1;
917         }
```

二、TS100状态

6.THERMOMETER温度计状态

定义在CTRL.c 360行到390行

该温度计为热电偶测温，所得温度只能大致估计，准确度不是太高。

```
360     case THERMOMETER:
361         if(KD_TIMER > 0) {
362             Set_gKey(NO_KEY);
363             break;
364         }
365         switch(Get_gKey()) {                    判断按键
366         case KEY_CN|KEY_V1:
367         case KEY_CN|KEY_V2:                     长按任意键返回
368             back_prestatus = 1;
369             break;
370         case KEY_CN|KEY_V3:                     温度校准
371             Zero_Calibration();                 温度0点校准
372             if(Get_CalFlag() == 1) {            校准成功，保存数据
373                 Disk_BuffInit();
374                 Config_Analysis();              启动虚拟U盘
375             }
376             KD_TIMER = 200; //20150717 修改
377             break;
378         default:
379             break;
380         }
381         if(back_prestatus == 1) {               长按任意键返回待机状态
382             back_prestatus = 0;
383             Set_HeatingTime(0);
384             Set_CtrlStatus(IDLE);
385             gPre_status = THERMOMETER;
386             gIs_restartkey = 1;
387             Set_LongKeyFlag(0);
388             KD_TIMER = 50; //
389         }
390         break;
```

二、TS100状态

7.ALARM警报状态

定义在CTRL.c 392 行到419行

主要功能根据警报类型做出不同的应对措施。

```
392         switch(Get_AlarmType()) {  警报类型
393         case HIGH_TEMP:
394         case SEN_ERR: 不能读取烙铁头数据
395             wk_temp      = device_info.t_work;
396             gTemp_data   = Get_Temp(wk_temp);
397             if(Get_AlarmType() == NORMAL_TEMP) {
398                 Set_CtrlStatus(TEMP_CTR);
399                 Set_UpdataFlag(0);
400             }
401             break;
402         case HIGH_VOLTAGE:
403         case LOW_VOLTAGE: 电压太低
404             if(Read_Vb(1) >= 1 && Read_Vb(1) <= 3) { 电压正常返回待机
405                 Set_HeatingTime(0);
406                 Set_LongKeyFlag(0);
407                 gIs_restartkey = 1;
408                 UI_TIMER = 2; // 2秒
409                 gPre_status = THERMOMETER;
410                 Set_CtrlStatus(IDLE);
411             }
412             break;
413         }
414
415         if(Get_HeatingTime != 0) {
416             Set_HeatingTime(0) ;                    //马上停止加热
417             HEAT_OFF();
418         }
419         break;
```

三、各个状态之间的相互切换

初始化状态

UI.c 123行

```
123     if(rev == 0)   Set_CtrlStatus(ALARM);
```

报警状态(ALARM)(电压过高)

UI.c 126行

```
126         Set_CtrlStatus(CONFIG);
```

U盘配置状态(CONFIG)(电压5V)

UI.c 128行

```
128         Set_CtrlStatus(IDLE);
```

待机状态(IDLE)(电压正常)

以下都是当前状态切换其他状态

待机状态(IDLE)

CTRL.c 226行

```
226                 Set_CtrlStatus(THERMOMETER);
```

温度计状态(THERMOMETER)(按B键)

CTRL.c 216行

```
216                 Set_CtrlStatus(TEMP_CTR);
```

温控状态(TEMP_CTR)(按A键)

CTRL.c 240行

```
240             Set_CtrlStatus(ALARM);
```

报警状态(ALARM)(烙铁头,电压,温度等等有误)

三、各个状态之间的相互切换

温控状态(TEMP_CTR)

CTRL.c 253行

```
253             Set_CtrlStatus(TEMP_SET);
```

设置状态(TEMP_SET)(长按任意键)

CTRL.c 307行

```
307             Set_CtrlStatus(ALARM);
```

报警状态(ALARM)(烙铁头,电压,温度等等有误)

CTRL.c 300行

```
300             Set_CtrlStatus(WAIT);
```

休眠状态(WAIT)(静止一段时间)

CTRL.c 260行270行

```
260             Set_CtrlStatus(IDLE);
261             gPre_status = TEMP_CTR;
262             gIs_restartkey = 1;
263             KD_TIMER = 50; //
264             break;
265         }
266
267         if(Read_Vb(1) >= 4) {
268             Set_HeatingTime(0);
269             Set_LongKeyFlag(0);
270             Set_CtrlStatus(IDLE);
```

待机状态(IDLE)(同时按双键)

三、各个状态之间的相互切换

休眠状态(WAIT):

CTRL.c 346行

```
346             Set_CtrlStatus(TEMP_CTR);
```

温控状态(TEMP_CTR)(移动烙铁)

CTRL.c 251行

```
251         case KEY_CN|KEY_V2:
```

报警状态(ALARM)(烙铁头,电压,温度等等有误)

CTRL.c 325-338行

```
325             Set_CtrlStatus(IDLE);
326             G6_TIMER = device_info.idle_time;
327             gPre_status = WAIT;
328             gIs_restartkey = 1;
329             KD_TIMER = 50; // 2秒
330         }
331
332         if(G6_TIMER == 0) { //进入待机
333             Set_HeatingTime(0);
334             Set_LongKeyFlag(0);
335             gIs_restartkey = 1;
336             KD_TIMER = 200; // 2秒
337             gPre_status = WAIT;
338             Set_CtrlStatus(IDLE);
```

待机状态(IDLE)(静止经过待机时间)

三、各个状态之间的相互切换

设置状态(TEMP_SET)

CTRL.c 356行

```
356                 Set_CtrlStatus(TEMP_CTR);
```

温控状态(TEMP_CTR)(不按键几秒钟)

报警状态(ALARM)

CTRL.c 410行

```
410                 Set_CtrlStatus(IDLE);
```

待机状态(IDLE)(解除报警后)

CTRL.c 398行

```
398                 Set_CtrlStatus(TEMP_CTR);
```

温控状态(TEMP_CTR)(解除报警后)

温度计(THERMOMETER):

CTRL.c 384行

```
384                 Set_CtrlStatus(IDLE);
```

待机状态(IDLE)(长按任意键)

配置文件模式(CONFIG):

进入配置文件模式，不可以转换其他状态