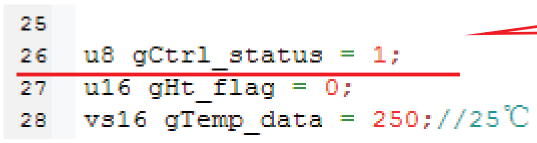
TS100 Source File Description

———Control Managerment

Content

I. System status

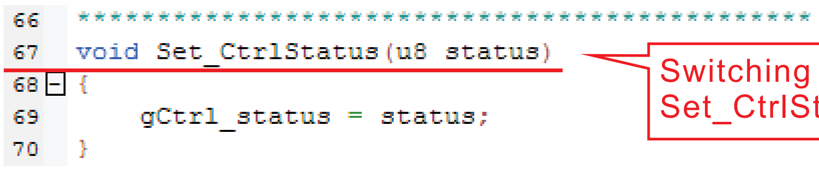
CTRL.c ,Line 26



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

The global variable gCtrl_status represents the status, as defined in Line 26 of CTRL.c; it has 8 statuses, as follows: IDLE, THERMOMETER, TEMP_CTR (Temperature Control), WAIT, TEMP_SET(Temperature Setup), CONFIG(Configuration), MODE_CNG(Backup), and ALARM;

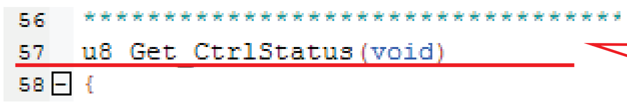
CTRL.c ,Line 67



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

Switching status is achieved through the Set_CtrlStatus (U8 status) function;

CTRL.c,Line 57



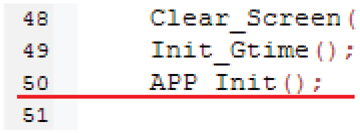
```
56  **********************************
57  u8 Get_CtrlStatus(void)
58  {
```

Reading of the present status is achieved through the Get_CtrlStatus (void) function.

II. TS100 status

By understanding of the status variable, and switching or reading the status functions, it is easy to switch between various statuses. The status control of TS100 is described in details as follows.

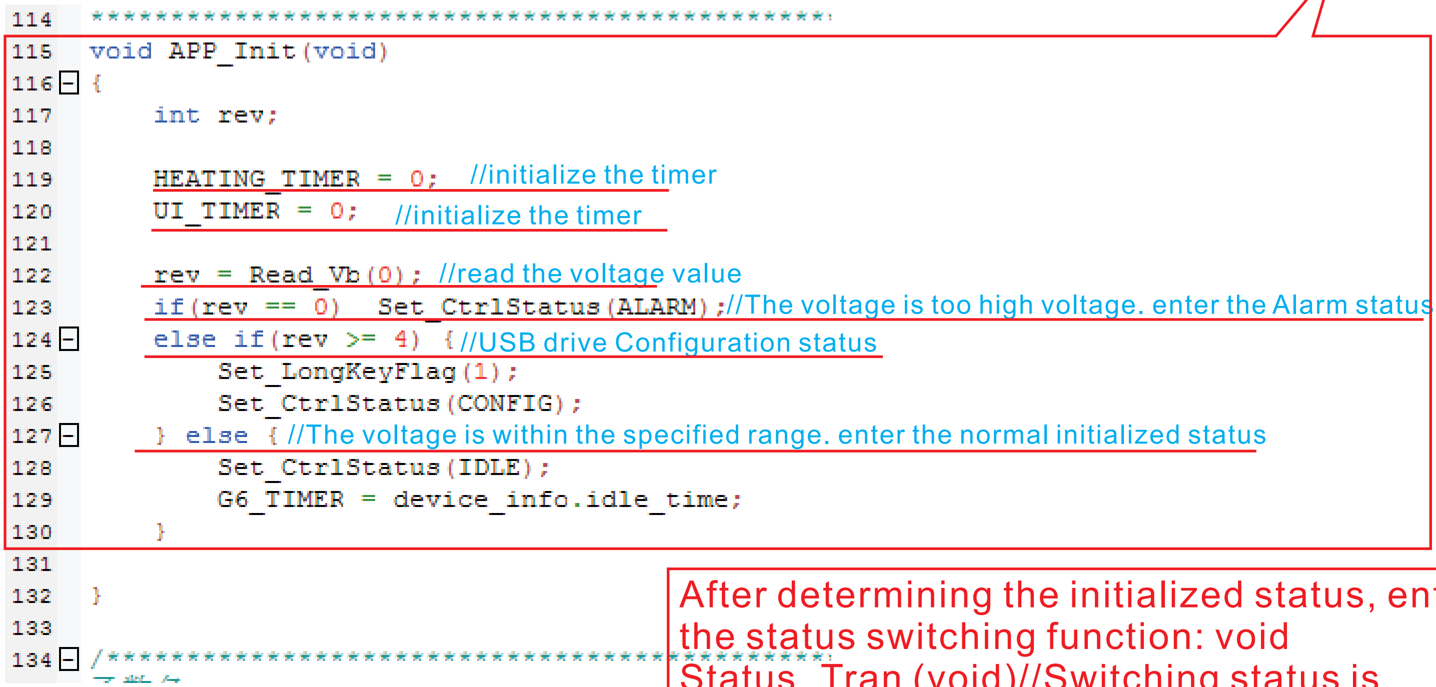
1. Initialized status



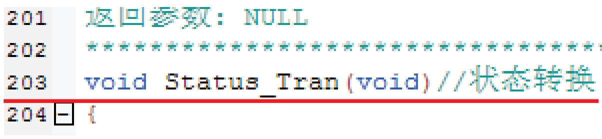
```
48      Clear_Screen();
49      Init_Gtime();
50      APP_Init();
51
```

Call this function at Line 50 of Main.c

UI.c,Line 115



```
114  *******************************************;
115  void APP_Init(void)
116  {
117      int rev;
118
119      HEATING_TIMER = 0;   //initialize the timer
120      UI_TIMER = 0;   //initialize the timer
121
122      rev = Read_Vb(0); //read the voltage value
123      if(rev == 0)   Set_CtrlStatus(ALARM);//The voltage is too high voltage. enter the Alarm status
124      else if(rev >= 4)  {//USB drive Configuration status
125          Set_LongKeyFlag(1);
126          Set_CtrlStatus(CONFIG);
127      } else  {//The voltage is within the specified range. enter the normal initialized status
128          Set_CtrlStatus(IDLE);
129          G6_TIMER = device_info.idle_time;
130      }
131
132  }
133
134  /*******************************************;
```



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

After determining the initialized status, enter the status switching function: void Status_Tran (void)//Switching status is defined in Line 203 of CTRL.c, and consists of a switch, by different statuses, to different handling procedures.

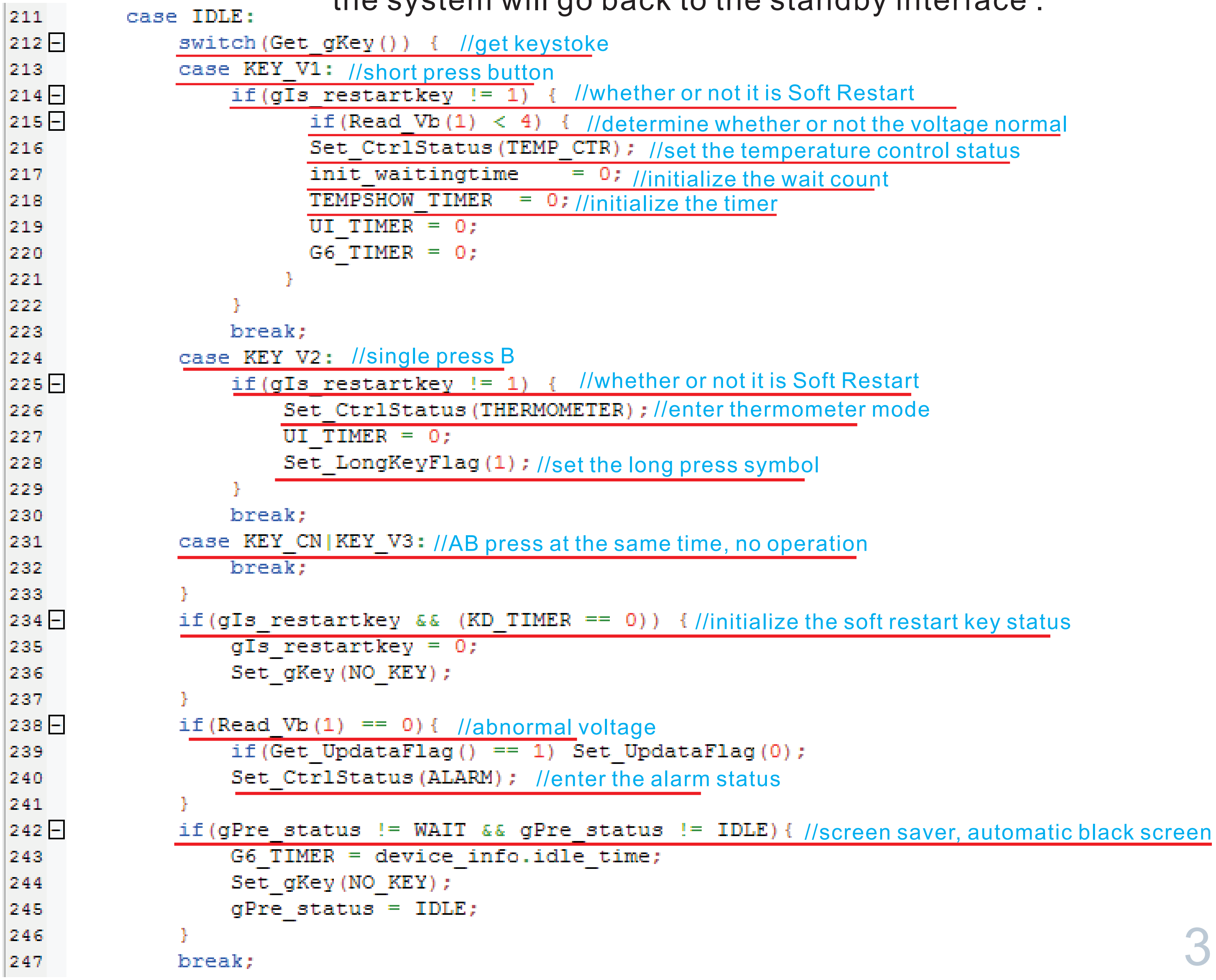
CTRL.c ,Line 203

II. TS100 status

2. IDLE status

CTRL.c,Lines 211 to 247

This status is the standby status and is displayed after powering on; when the voltage is too low, the system will return to the standby interface; after a period of dormancy, the system will go back to the standby interface .



```
211         case IDLE:
212             switch(Get_gKey()) {   //get keystoke
213             case KEY_V1:  //short press button
214                 if(gIs_restartkey != 1) {  //whether or not it is Soft Restart
215                     if(Read_Vb(1) < 4) {  //determine whether or not the voltage normal
216                     Set_CtrlStatus(TEMP_CTR); //set the temperature control status
217                     init_waitingtime     = 0; //initialize the wait count
218                     TEMPSHOW_TIMER   = 0; //initialize the timer
219                     UI_TIMER = 0;
220                     G6_TIMER = 0;
221                     }
222                 }
223                 break;
224             case KEY_V2:  //single press B
225                 if(gIs_restartkey != 1) {  //whether or not it is Soft Restart
226                     Set_CtrlStatus(THERMOMETER); //enter thermometer mode
227                     UI_TIMER = 0;
228                     Set_LongKeyFlag(1); //set the long press symbol
229                 }
230                 break;
231             case KEY_CN|KEY_V3: //AB press at the same time, no operation
232                 break;
233             }
234             if(gIs_restartkey && (KD_TIMER == 0)) { //initialize the soft restart key status
235                 gIs_restartkey = 0;
236                 Set_gKey(NO_KEY);
237             }
238             if(Read_Vb(1) == 0){  //abnormal voltage
239                 if(Get_UpdataFlag() == 1) Set_UpdataFlag(0);
240                 Set_CtrlStatus(ALARM);  //enter the alarm status
241             }
242             if(gPre_status != WAIT && gPre_status != IDLE){ //screen saver, automatic black screen
243                 G6_TIMER = device_info.idle_time;
244                 Set_gKey(NO_KEY);
245                 gPre_status = IDLE;
246             }
247             break;
```

II. TS100 status

3. TEMP_CTR status

CTRL.c,Lines 248 to 309

Temperature Control status, a status in the process such as heating, cooling, and maintaining temperature: main operating status of the soldering iron.

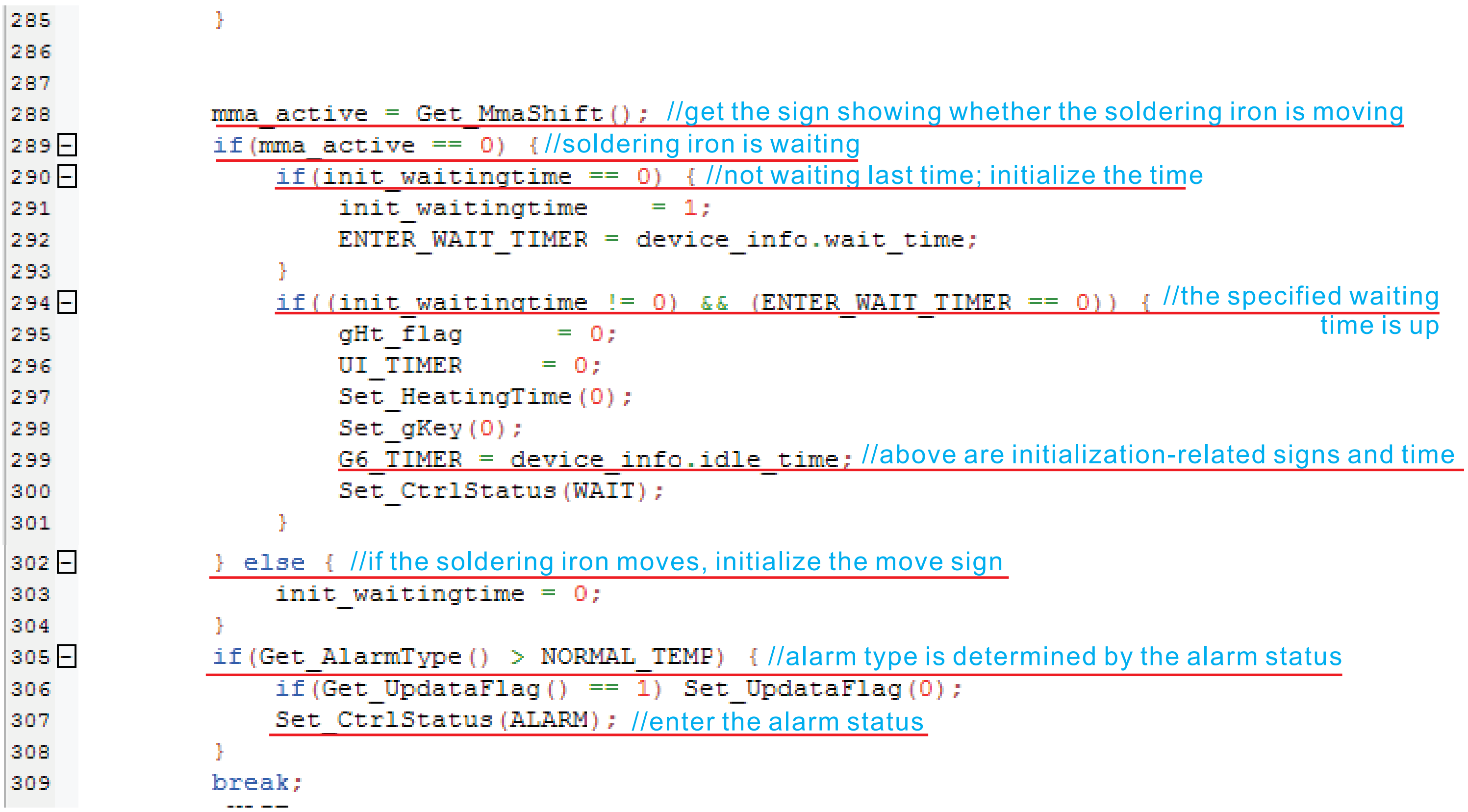
```
248     case TEMP_CTR:
249         switch(Get_gKey()) {
250         case KEY_CN|KEY_V1:
251         case KEY_CN|KEY_V2:  //long press any button
252             Set_HeatingTime(0); //The heating time is set to 0, which is to stop heating.
253             Set_CtrlStatus(TEMP_SET); //enter Setup mode
254             HEATING_TIMER        = 0;
255             EFFECTIVE_KEY_TIMER  = 500; //initialize time for exiting setup mode
256             break;
257         case KEY_CN|KEY_V3: //AB double key press
258             Set_HeatingTime(0); //stop heating
259             Set_LongKeyFlag(0); //set the key sign, not for a long press
260             Set_CtrlStatus(IDLE); //return to the initialized status, that is, the standby interface
261             gPre_status = TEMP_CTR;
262             gIs_restartkey = 1;
263             KD_TIMER = 50; //
264             break;
265         }
266 
267         if(Read_Vb(1) >= 4) { //read the voltage; if the voltage is below the set minimum value,
                                     return to the standby interface
268             Set_HeatingTime(0); //top heating
269             Set_LongKeyFlag(0);
270             Set_CtrlStatus(IDLE); //return to the standby interface
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); //calculate the heating time
                                                                     according to the PID algorithm
280             Set_HeatingTime(heat_timecnt);//set the heating time
281             HEATING_TIMER = HEATINGCYCLE; //assign value to heating cycle
282         }
283         if(Get_HeatingTime() == 0) { //when heating is not going on, conduct mandatory testing
                                           of heating time
284             HEATING_TIMER = 0;
```

II. TS100 status

3. TEMP_CTR status

CTRL.c,Lines 248 to 309

Temperature Control status, a status in the process such as heating, cooling, and maintaining temperature: main operating status of the soldering iron.



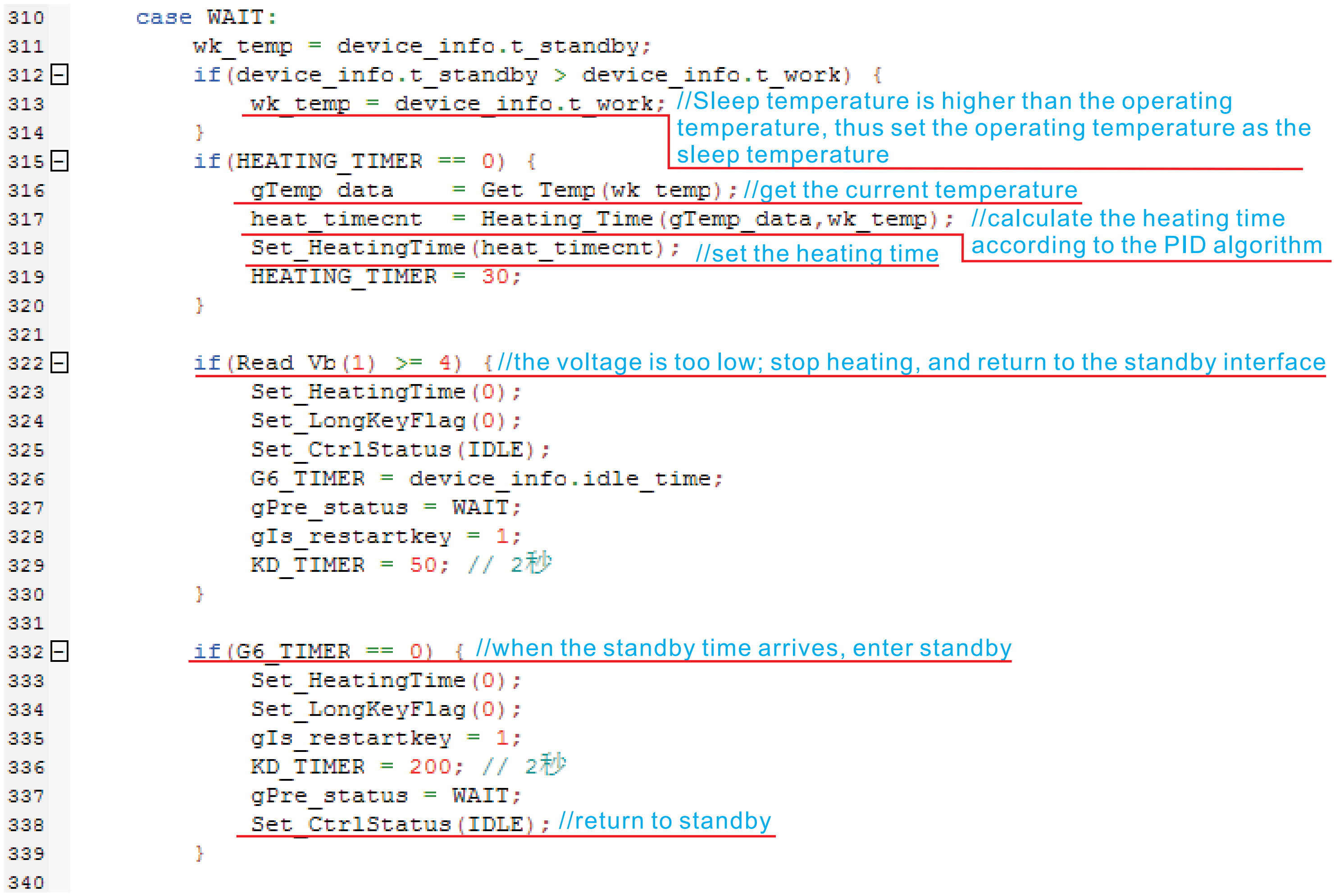
```
285             }
286
287
288         mma_active = Get_MmaShift(); //get the sign showing whether the soldering iron is moving
289         if(mma_active == 0) {//soldering iron is waiting
290             if(init_waitingtime == 0) { //not waiting last time; initialize the time
291                 init_waitingtime    = 1;
292                 ENTER_WAIT_TIMER = device_info.wait_time;
293             }
294             if((init_waitingtime != 0) && (ENTER_WAIT_TIMER == 0)) { //the specified waiting time is up
295                 gHt_flag      = 0;
296                 UI_TIMER      = 0;
297                 Set_HeatingTime(0);
298                 Set_gKey(0);
299                 G6_TIMER = device_info.idle_time; //above are initialization-related signs and time
300                 Set_CtrlStatus(WAIT);
301             }
302         } else { //if the soldering iron moves, initialize the move sign
303             init_waitingtime = 0;
304         }
305         if(Get_AlarmType() > NORMAL_TEMP) { //alarm type is determined by the alarm status
306             if(Get_UpdataFlag() == 1) Set_UpdataFlag(0);
307             Set_CtrlStatus(ALARM); //enter the alarm status
308         }
309         break;
```

II. TS100 status

4. Wait status

CTRL.c ,Lines 310 to 353

After the system has been waiting for a period of time, it will enter sleep status; under sleep status, the temperature is the sleep temperature; if the sleep temperature is greater than the current temperature, the current temperature will be maintained as the sleep temperature.

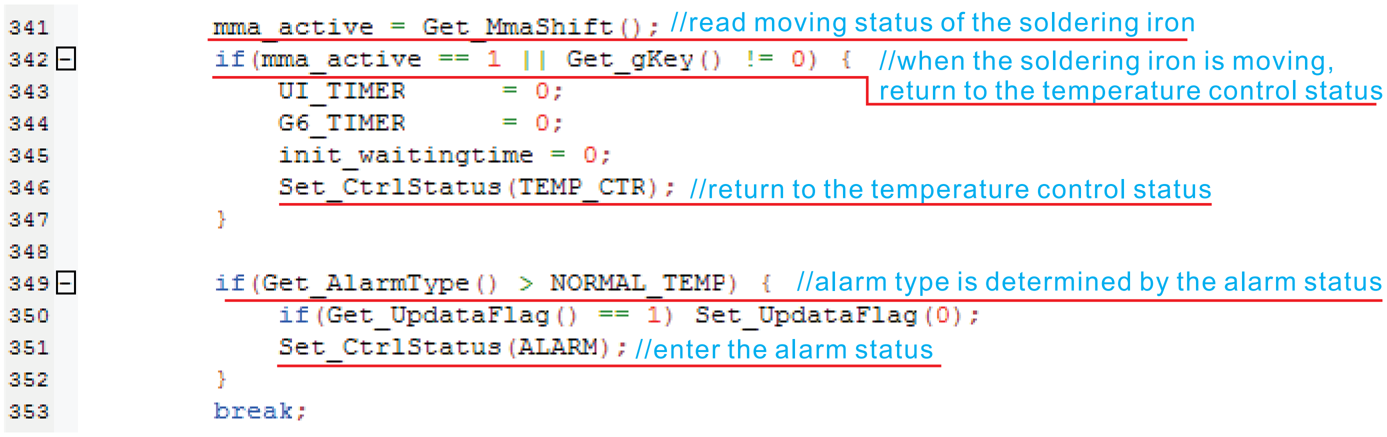


```
310         case WAIT:
311             wk_temp = device_info.t_standby;
312 ⊟           if(device_info.t_standby > device_info.t_work) {
313                 wk_temp = device_info.t_work; //Sleep temperature is higher than the operating
                                                  temperature, thus set the operating temperature as the
314             }                                 sleep temperature
315 ⊟           if(HEATING_TIMER == 0) {
316                 gTemp_data     = Get_Temp(wk_temp); //get the current temperature
317                 heat_timecnt   = Heating_Time(gTemp_data,wk_temp); //calculate the heating time
318                 Set_HeatingTime(heat_timecnt); //set the heating time  according to the PID algorithm
319                 HEATING_TIMER = 30;
320             }
321
322 ⊟           if(Read_Vb(1) >= 4) {//the voltage is too low; stop heating, and return to the standby interface
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) { //when the standby time arrives, enter standby
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); //return to standby
339             }
340
```

4. Wait status

CTRL.c,Lines 310 to 353

After the system has been waiting for a period of time, it will enter sleep status; under sleep status, the temperature is the sleep temperature; if the sleep temperature is greater than the current temperature, the current temperature will be maintained as the sleep temperature.



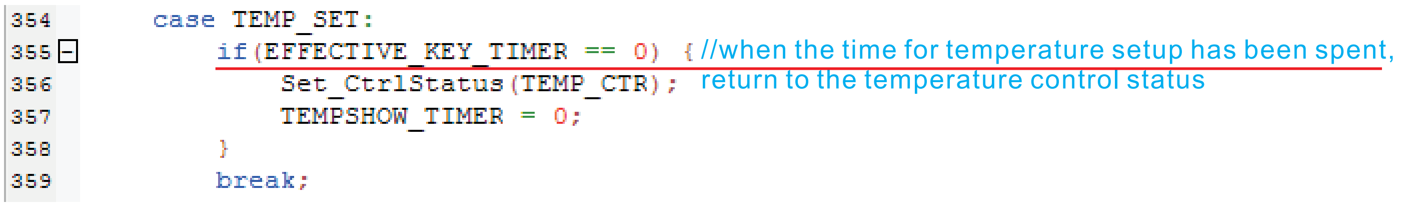
```
341         mma_active = Get_MmaShift(); //read moving status of the soldering iron
342         if(mma_active == 1 || Get_gKey() != 0) {  //when the soldering iron is moving,
                                                       return to the temperature control status
343             UI_TIMER        = 0;
344             G6_TIMER        = 0;
345             init_waitingtime = 0;
346             Set_CtrlStatus(TEMP_CTR); //return to the temperature control status
347         }
348
349         if(Get_AlarmType() > NORMAL_TEMP) {  //alarm type is determined by the alarm status
350             if(Get_UpdataFlag() == 1) Set_UpdataFlag(0);
351             Set_CtrlStatus(ALARM); //enter the alarm status
352         }
353         break;
```

II. TS100 status

5. TEMP_SET status

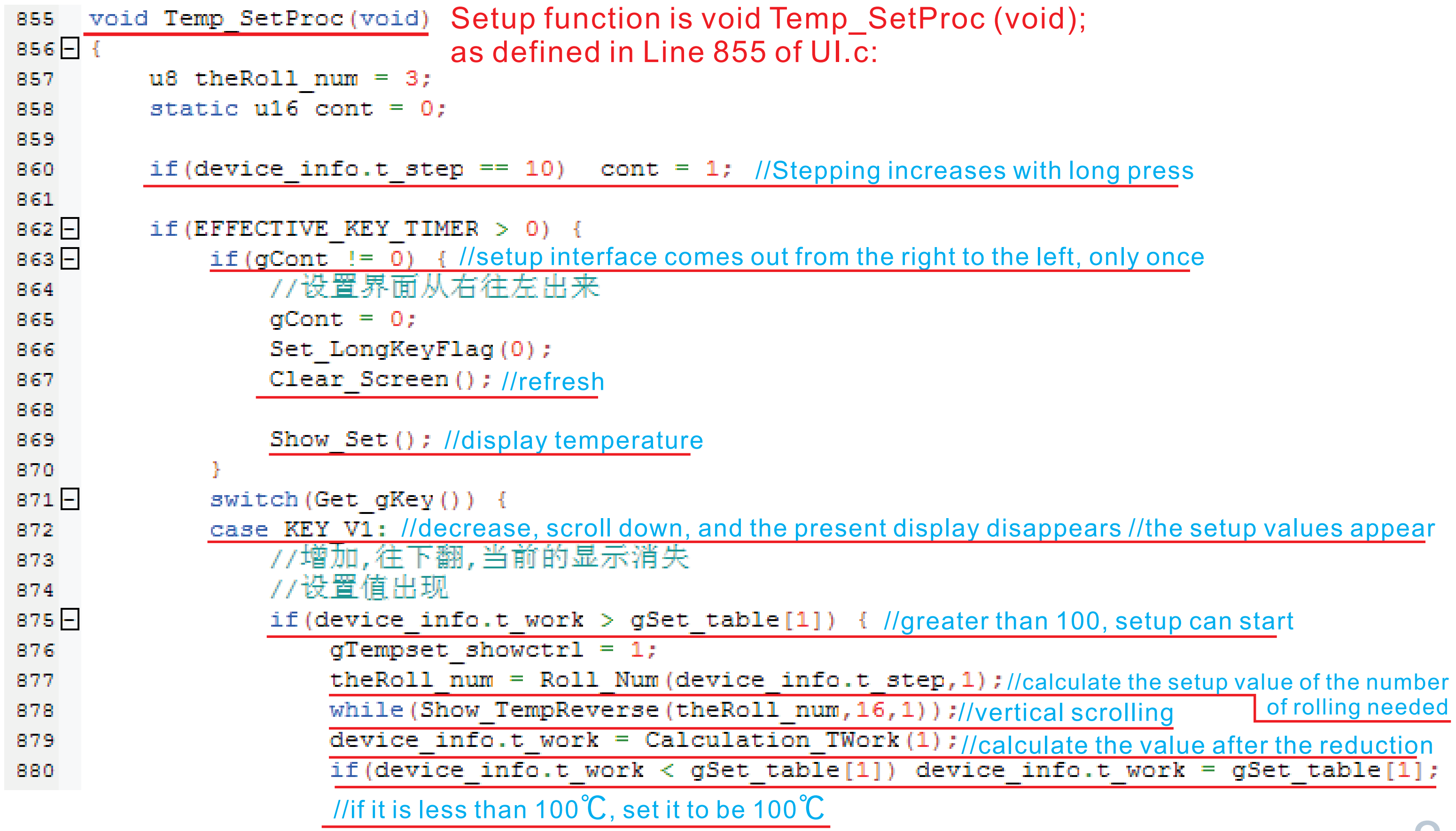
CTRL.c ,Line 355

Temperature setup status is used to set up time;in temperature control status,long press any button to enter the Temperature setup status.



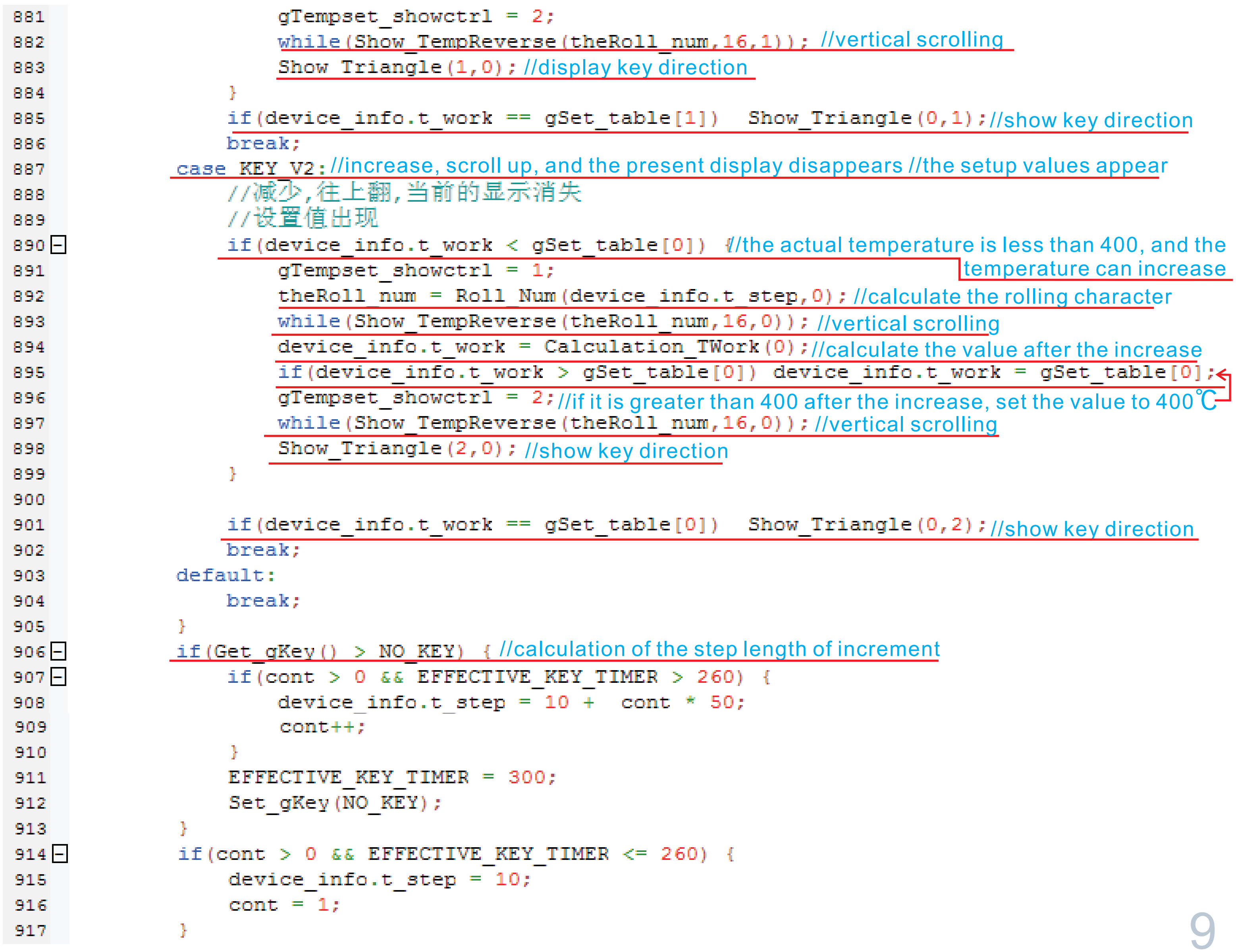
```
354         case TEMP_SET:
355             if(EFFECTIVE_KEY_TIMER == 0) { //when the time for temperature setup has been spent,
356                 Set_CtrlStatus(TEMP_CTR);  return to the temperature control status
357                 TEMPSHOW_TIMER = 0;
358             }
359             break;
```

Setup function is void Temp_SetProc (void); as defined in Line 855 of UI.c:



```
855 void Temp_SetProc(void)
856 {
857     u8 theRoll_num = 3;
858     static u16 cont = 0;
859 
860     if(device_info.t_step == 10)   cont = 1;  //Stepping increases with long press
861 
862     if(EFFECTIVE_KEY_TIMER > 0) {
863         if(gCont  != 0)  { //setup interface comes out from the right to the left, only once
864             //设置界面从右往左出来
865             gCont = 0;
866             Set_LongKeyFlag(0);
867             Clear_Screen(); //refresh
868 
869             Show_Set(); //display temperature
870         }
871         switch(Get_gKey()) {
872         case KEY_V1: //decrease, scroll down, and the present display disappears //the setup values appear
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]) { //greater than 100, setup can start
876                 gTempset_showctrl = 1;
877                 theRoll_num = Roll_Num(device_info.t_step,1);//calculate the setup value of the number
                                                                   of rolling needed
878                 while(Show_TempReverse(theRoll_num,16,1));//vertical scrolling
879                 device_info.t_work = Calculation_TWork(1);//calculate the value after the reduction
880                 if(device_info.t_work < gSet_table[1]) device_info.t_work = gSet_table[1];
                //if it is less than 100℃, set it to be 100℃
```

5. TEMP_SET status



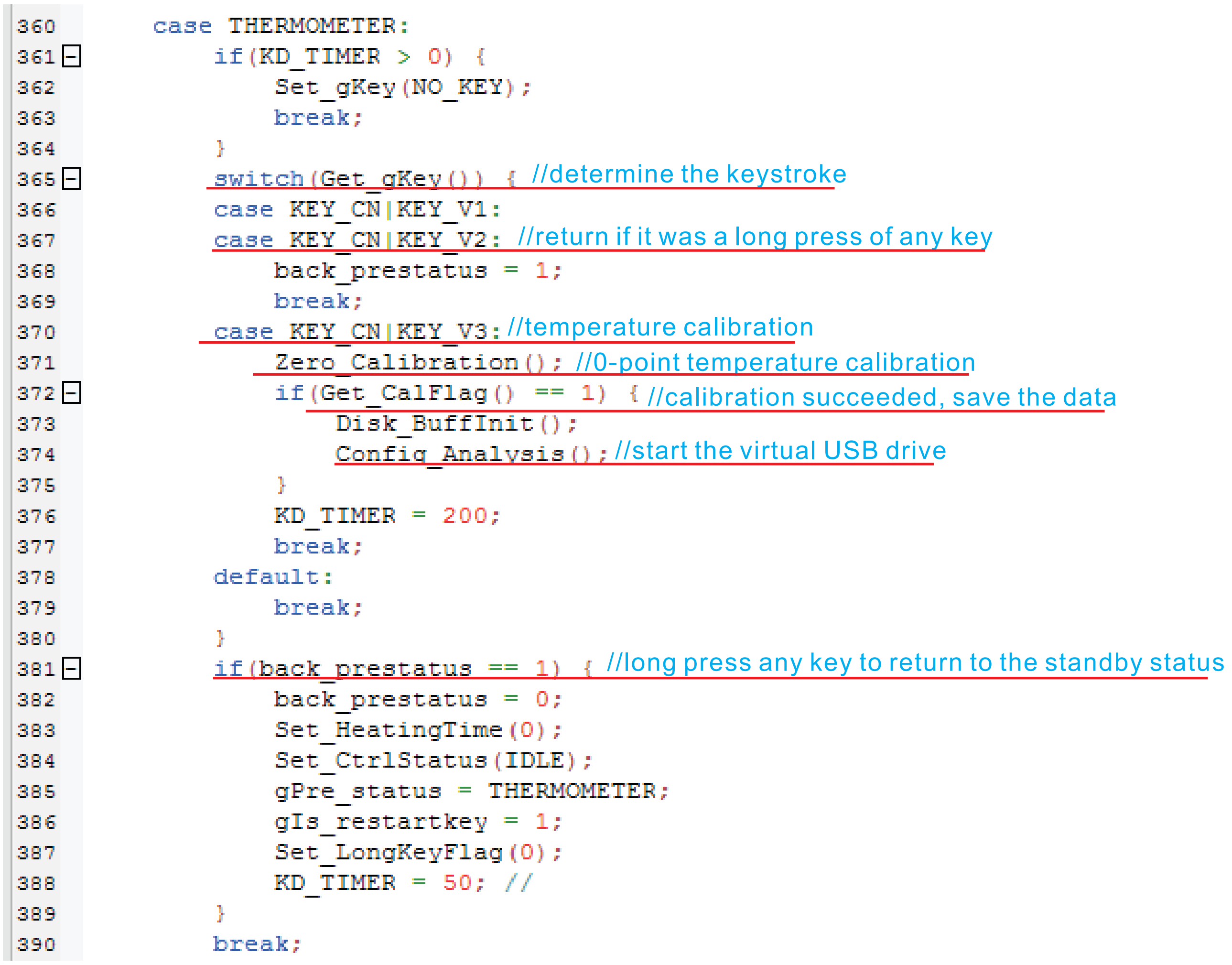
```
                gTempset_showctrl = 2;
                while(Show_TempReverse(theRoll_num,16,1)); //vertical scrolling
                Show_Triangle(1,0); //display key direction
            }
            if(device_info.t_work == gSet_table[1])  Show_Triangle(0,1);//show key direction
            break;
        case KEY_V2://increase, scroll up, and the present display disappears //the setup values appear
            //减少,往上翻,当前的显示消失
            //设置值出现
            if(device_info.t_work < gSet_table[0]) {//the actual temperature is less than 400, and the
                gTempset_showctrl = 1;                                                temperature can increase
                theRoll_num = Roll_Num(device_info.t_step,0); //calculate the rolling character
                while(Show_TempReverse(theRoll_num,16,0)); //vertical scrolling
                device_info.t_work = Calculation_TWork(0);//calculate the value after the increase
                if(device_info.t_work > gSet_table[0]) device_info.t_work = gSet_table[0];
                gTempset_showctrl = 2;//if it is greater than 400 after the increase, set the value to 400℃
                while(Show_TempReverse(theRoll_num,16,0)); //vertical scrolling
                Show_Triangle(2,0); //show key direction
            }

            if(device_info.t_work == gSet_table[0])  Show_Triangle(0,2);//show key direction
            break;
        default:
            break;
        }
        if(Get_gKey() > NO_KEY) { //calculation of the step length of increment
            if(cont > 0 && EFFECTIVE_KEY_TIMER > 260) {
                device_info.t_step = 10 +  cont * 50;
                cont++;
            }
            EFFECTIVE_KEY_TIMER = 300;
            Set_gKey(NO_KEY);
        }
        if(cont > 0 && EFFECTIVE_KEY_TIMER <= 260) {
            device_info.t_step = 10;
            cont = 1;
        }
```

6.THERMOMETER status

CTRL.c ,Lines 360 to 390

The thermometer is of the thermocouple type; the resulting temperature is only a rough estimate, not very accurate.



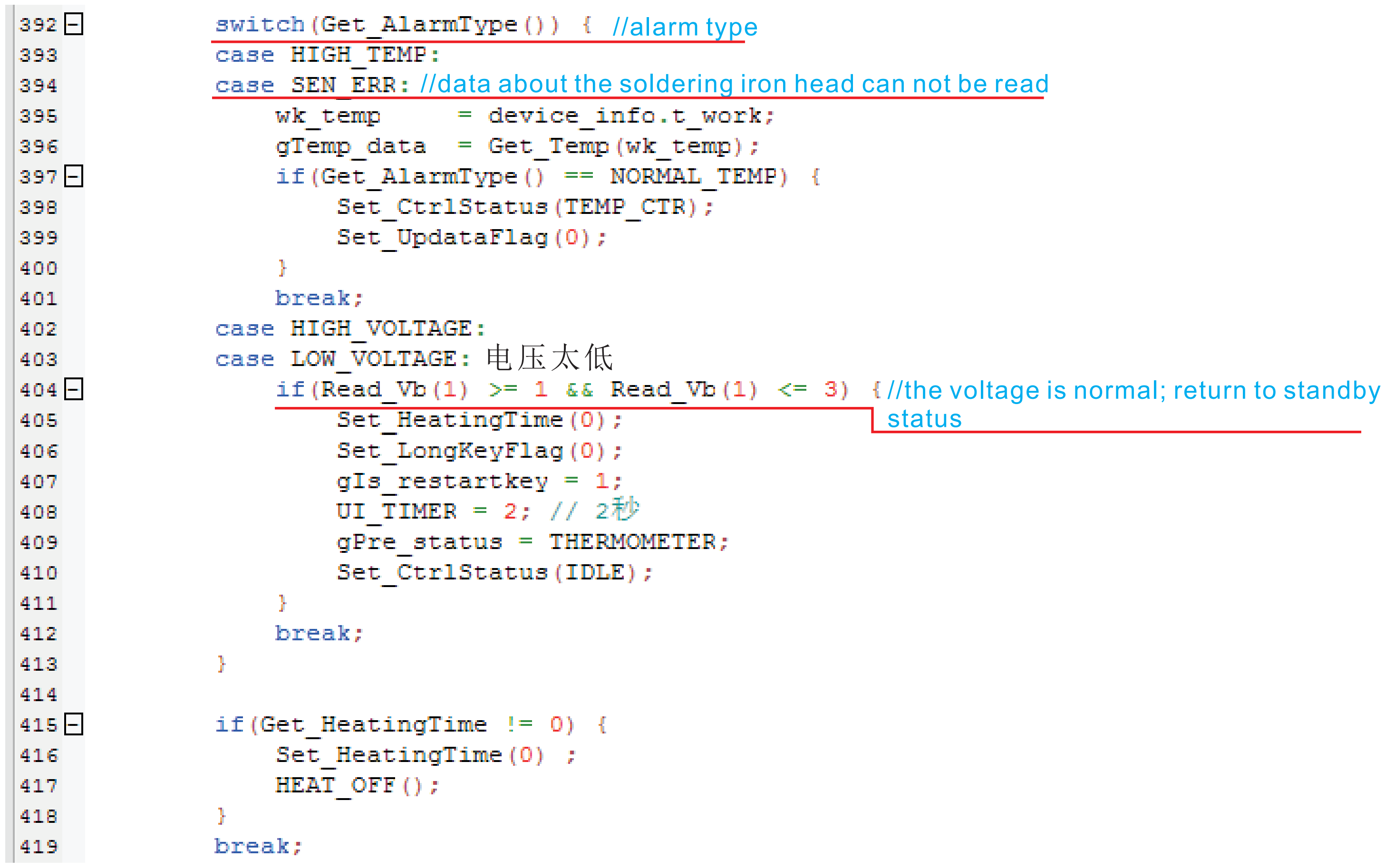
```
360        case THERMOMETER:
361            if(KD_TIMER > 0) {
362                Set_gKey(NO_KEY);
363                break;
364            }
365            switch(Get_gKey()) { //determine the keystroke
366            case KEY_CN|KEY_V1:
367            case KEY_CN|KEY_V2: //return if it was a long press of any key
368                back_prestatus = 1;
369                break;
370            case KEY_CN|KEY_V3: //temperature calibration
371                Zero_Calibration(); //0-point temperature calibration
372                if(Get_CalFlag() == 1) { //calibration succeeded, save the data
373                    Disk_BuffInit();
374                    Config_Analysis(); //start the virtual USB drive
375                }
376                KD_TIMER = 200;
377                break;
378            default:
379                break;
380            }
381            if(back_prestatus == 1) { //long press any key to return to the standby status
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;
```

II. TS100 status

7.ALARM status

CTRL.c,Lines 392 to 419

The main function is to responds to the various alarm types.



```
392         switch(Get_AlarmType()) {  //alarm type
393         case HIGH_TEMP:
394         case SEN_ERR: //data about the soldering iron head can not be read
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) {//the voltage is normal; return to standby
                                                        status
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) ;
417             HEAT_OFF();
418         }
419         break;
```

III. Switching between the various statuses

Initialized status

UI.c, Line 123

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

Alarm status (ALARM) (high voltage)

UI.c, Line 126

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

USB drive configuration status (CONFIG) (voltage of 5V)

UI.c, Line 128

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

Standby status (IDLE) (normal voltage)

Below is status switching from current status to other status.

Standby status (IDLE)

CTRL.c, Line 226

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

Thermometer status (THERMOMETER) (press B key)

CTRL.c, Line 216

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

Temperature control status (TEMP_CTR) (press A key)

CTRL.c, Line 240

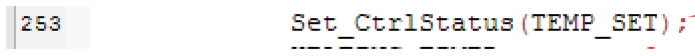
```
240                 Set_CtrlStatus(ALARM);
```

Alarm status (ALARM) (errors in soldering iron head, voltage, temperature, etc.)

III. Switching between the various statuses

Temperature control status (TEMP_CTR)

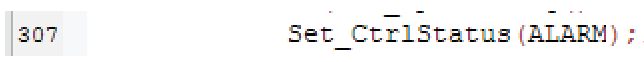
CTRL.c, Line 253



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

Setup status (TEMP_SET) (long press any key)

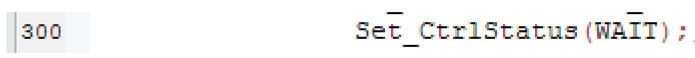
CTRL.c, Line 307



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

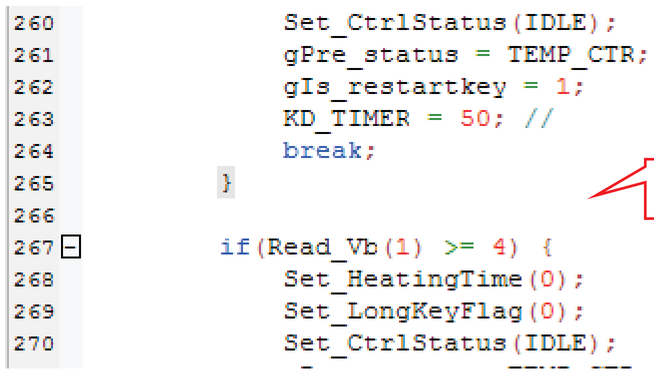
Alarm status (ALARM) (errors in soldering iron head, voltage, temperature, etc.)

CTRL.c, Line 300



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

Sleep status (WAIT) (waiting for a period of time)



```
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);
```

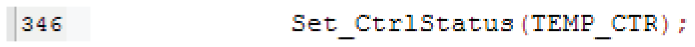
CTRL.c, Lines 260 to 270

Standby status (IDLE) (pressing two keys at the same time)

III. Switching between the various statuses

Sleep status (WAIT)

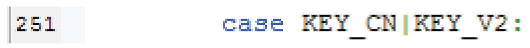
CTRL.c, Line 346



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

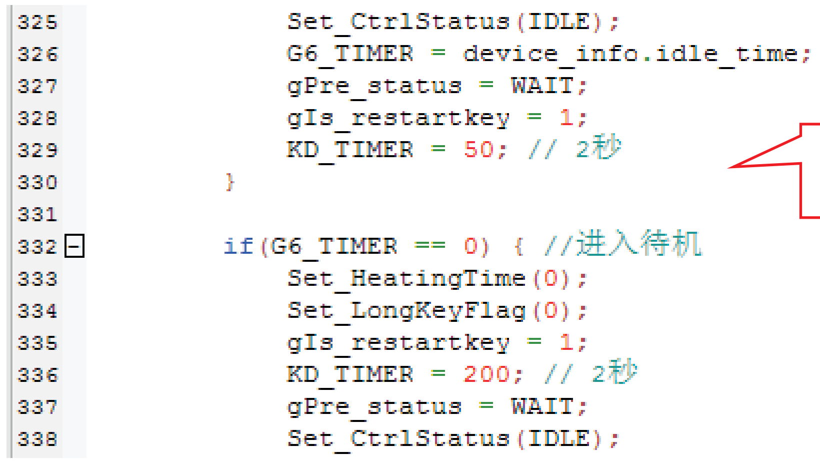
Temperature control status (TEMP_CTR) (moving soldering iron)

CTRL.c, Line 251



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

Alarm status (ALARM) (errors in soldering iron head, voltage, temperature, etc.)



```
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);
```

CTRL.c, Lines 325 to 338

Standby status (IDLE) (static through standby time)

III. Switching between the various statuses

Setup status (TEMP_SET)

CTRL.c, Line 356

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

Temperature control status (TEMP_CTR) (No keystrokes for a few seconds)

Alarm status (ALARM)

CTRL.c, Line 410

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

Standby status (IDLE) (after alarm is disarmed)

CTRL.c, Line 398

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

Temperature control status (TEMP_CTR) (after alarm is disarmed)

Thermometer (THERMOMETER)

CTRL.c Line 384

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

Standby status (IDLE) (long press any key)

Configuration status (CONFIG)

Enter the configuration file mode; it is not possible to switch to any other statuses.