

STM32F103 机智云

云

开发手册 V1.0

移植部分再来看本教程

4.3.3 移植

1) 添加文件

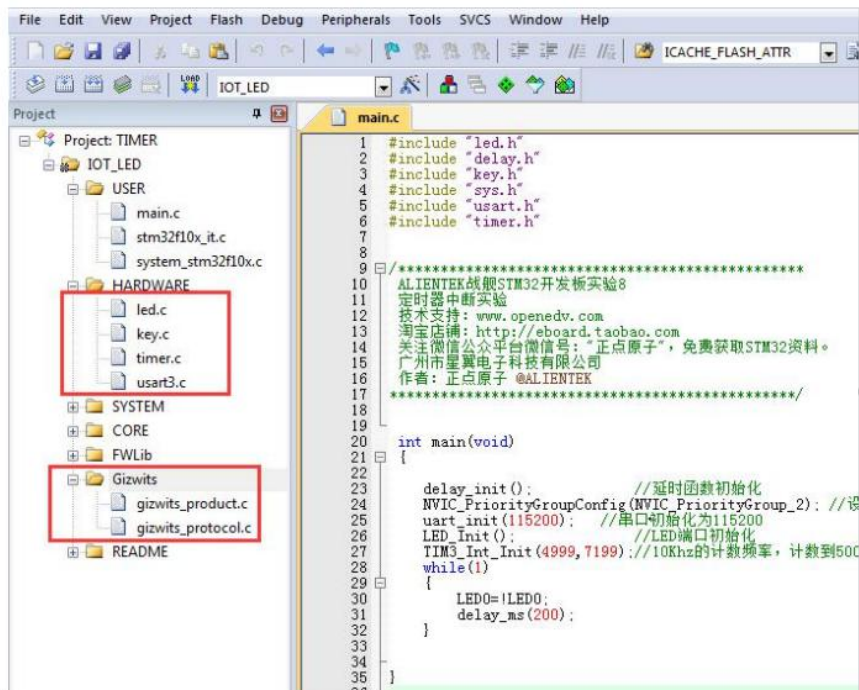
我们把之前下载的通用平台 SDK 下 “Gizwits” 文件夹复制到我们的工程下，如图 4.3.3.1 所示。

名称	修改日期
CORE	2017/5/31 15:07
Gizwits	2017/6/2 10:58
HARDWARE	2017/5/31 15:07
OBJ	2017/6/1 14:54
STM32F10x_FWLib	2017/5/31 15:07
SYSTEM	2017/5/31 15:07
USER	2017/6/2 11:08
keilkill.bat	2016/11/18 11:52

图 4.3.3.1 添加 Gizwits 文件

由于 WIFI 的通讯是插在 ATK MODULE 的接口上，而 ATK MODULE 接口是连接到板子的串口 3 上，在 HARDWARE 文件夹中我们创建 USART3 文件夹，并且在文件夹内创建 usart3.c 和 usart3.h 文件 如图 4.3.3.2 所示。

前面步骤完成后，需要将其添加到工程分组中，添加完成以后如图 4.3.3.4 所示。



然后就是区分内容了

1.

Gizwits	2020/3/31 21:10	文件夹	
Hal	2020/3/31 21:10	文件夹	
Inc	2020/3/31 21:10	文件夹	
MDK-ARM	2020/3/31 21:10	文件夹	
Src	2020/3/31 21:10	文件夹	
Utils	2020/3/31 21:10	文件夹	
.mxproject	2019/3/2 1:36	MXPROJECT 文件	4 KB
Changelog.txt	2019/3/2 1:36	文本文档	1 KB

将 Utils 文件夹下内容添加至工程 库函数也要包含

2.gitwits_product.c

首先注释开头几行 如果使用可写类型 include 相应.h 文件

添加 extern u8 wifi_sta;

extern dataPoint_t currentDataPoint;

```

12  *          链接|增值|升阶|中立|安全|自有|自出|生念
13  *          www.gizwits.com
14  *
15  *****
16
17  #include <stdio.h>
18  #include <string.h>
19
20  #include "gizwits_product.h"
21  #include "common.h"
22  #include "usart3.h"
23  #include "led.h"
24
25  static uint32_t timerMsCount;
26  uint8_t aRxBuffer;
27  extern u8 wifi_sta;
28  /** User area the current device state structure*/
29  extern dataPoint_t currentDataPoint;
30  //extern keysTypeDef_t keys;
31
32  //extern TIM_HandleTypeDef htim2;
33  //extern UART_HandleTypeDef huart1;
34  //extern UART_HandleTypeDef huart2;
35
36  /**@} */

```

可写内容的修改

```

65  #else
66  moduleInfo_t *ptModuleInfo = (moduleInfo_t *)gizdata;
67  #endif
68
69  if((NULL == info) || (NULL == gizdata))
70  {
71      return -1;
72  }
73
74  for(i=0; i<info->num; i++)
75  {
76      switch(info->event[i])
77      {
78          case EVENT_LEDOnoff:
79              currentDataPoint.valueLEDOnoff = dataPointPtr->valueLEDOnoff;
80              GIZWITS_LOG("Evt: EVENT_LEDOnoff %d \n", currentDataPoint.valueLEDOnoff);
81              if(0x01 == currentDataPoint.valueLEDOnoff)
82              {
83                  LED1=0; //user handle
84              }
85              else
86              {
87                  //user handle
88                  LED1=1;
89              }
90              break;
91
92
93
94
95              case WIFI_SOFTAP:
96                  break;
97              case WIFI_AIRLINK:
98                  break;
99              case WIFI_STATION:
100                 break;
101              case WIFI_CON_ROUTER:
102
103                 break;
104             --- WIFI_DISCON_ROUTER.

```

注释掉 userHandle()和 PUTCHAR_PROTOTYPE 和 uartInit()、HAL_TIM_PeriodElapsedCallback、timerInit()、HAL_UART_RxCpltCallback()；

```

168  L */
169  void userHandle(void)
170  {
171  L /*
172  L
173  L     */
174  L
175  L }
176  L

237  L * @retval None
238  L */
239  //void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
240  //{
241  //    if(htim==&htim2)
242  //    {
243  //        keyHandle((keysTypedef_t *)&keys);
244  //        gizTimerMs();
245  //    }
246  //}
247
248  L /**
249  L  * @brief Period elapsed callback in non blocking mode
250  L  * @param htim : TIM handle
251  L  * @retval None
252  L  */
253  //void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
254  //{
255  //    if(htim==&htim2)
256  //    {
257  //        keyHandle((keysTypedef_t *)&keys);
258  //        gizTimerMs();
259  //    }
260  //}
261
262  L /**
263  L  * @brief Timer TIM3 init function
264  L  * @param none
265  L  * @return none
266  L  */
267  void timerInit(void)
268  {
269  //    HAL_TIM_Base_Start_IT(&htim2);
270  //}
271
272  L /**
273  L  * @brief This function handles USART IDLE interrupt.
274  L  */
275  //void HAL_UART_RxCpltCallback(UART_HandleTypeDef *UartHandle)
276  //{
277  //    if(UartHandle->Instance == USART2)
278  //    {
279  //        gizPutData((uint8_t *)&aRxBuffer, 1);
280  //        HAL_UART_Receive_IT(&huart2, (uint8_t *)&aRxBuffer, 1); //开启下一次接收中断
281  //    }
282  //}
283
284  L /**
285  L  * @brief
286  L  */
287  //void
288  //{
289  //}
290  L /**
291  L  * @brief
292  L  */
293  //void
294  //{
295  //}
296  L /**
297  L  * @brief
298  L  */
299  //void
300  //{
301  //}
302  L /**
303  L  * @brief
304  L  */
305  //void
306  //{
307  //}
308  L /**
309  L  * @brief
310  L  */
311  //void
312  //{
313  //}
314  L /**
315  L  * @brief
316  L  */
317  //void
318  //{
319  //}
320  L /**
321  L  * @brief
322  L  */
323  //void
324  //{
325  //}
326  L /**
327  L  * @brief
328  L  */
329  //void
330  //{
331  //}
332  L /**
333  L  * @brief
334  L  */
335  //void
336  //{
337  //}
338  L /**
339  L  * @brief
340  L  */
341  //void
342  //{
343  //}
344  L /**
345  L  * @brief
346  L  */
347  //void
348  //{
349  //}
350  L /**
351  L  * @brief
352  L  */
353  //void
354  //{
355  //}
356  L /**
357  L  * @brief
358  L  */
359  //void
360  //{
361  //}
362  L /**
363  L  * @brief
364  L  */
365  //void
366  //{
367  //}
368  L /**
369  L  * @brief
370  L  */
371  //void
372  //{
373  //}
374  L /**
375  L  * @brief
376  L  */
377  //void
378  //{
379  //}
380  L /**
381  L  * @brief
382  L  */
383  //void
384  //{
385  //}
386  L /**
387  L  * @brief
388  L  */
389  //void
390  //{
391  //}
392  L /**
393  L  * @brief
394  L  */
395  //void
396  //{
397  //}
398  L /**
399  L  * @brief
400  L  */
401  //void
402  //{
403  //}
404  L /**
405  L  * @brief
406  L  */
407  //void
408  //{
409  //}
410  L /**
411  L  * @brief
412  L  */
413  //void
414  //{
415  //}
416  L /**
417  L  * @brief
418  L  */
419  //void
420  //{
421  //}
422  L /**
423  L  * @brief
424  L  */
425  //void
426  //{
427  //}
428  L /**
429  L  * @brief
430  L  */
431  //void
432  //{
433  //}
434  L /**
435  L  * @brief
436  L  */
437  //void
438  //{
439  //}
440  L /**
441  L  * @brief
442  L  */
443  //void
444  //{
445  //}
446  L /**
447  L  * @brief
448  L  */
449  //void
450  //{
451  //}
452  L /**
453  L  * @brief
454  L  */
455  //void
456  //{
457  //}
458  L /**
459  L  * @brief
460  L  */
461  //void
462  //{
463  //}
464  L /**
465  L  * @brief
466  L  */
467  //void
468  //{
469  //}
470  L /**
471  L  * @brief
472  L  */
473  //void
474  //{
475  //}
476  L /**
477  L  * @brief
478  L  */
479  //void
480  //{
481  //}
482  L /**
483  L  * @brief
484  L  */
485  //void
486  //{
487  //}
488  L /**
489  L  * @brief
490  L  */
491  //void
492  //{
493  //}
494  L /**
495  L  * @brief
496  L  */
497  //void
498  //{
499  //}
500  L /**
501  L  * @brief
502  L  */
503  //void
504  //{
505  //}
506  L /**
507  L  * @brief
508  L  */
509  //void
510  //{
511  //}
512  L /**
513  L  * @brief
514  L  */
515  //void
516  //{
517  //}
518  L /**
519  L  * @brief
520  L  */
521  //void
522  //{
523  //}
524  L /**
525  L  * @brief
526  L  */
527  //void
528  //{
529  //}
530  L /**
531  L  * @brief
532  L  */
533  //void
534  //{
535  //}
536  L /**
537  L  * @brief
538  L  */
539  //void
540  //{
541  //}
542  L /**
543  L  * @brief
544  L  */
545  //void
546  //{
547  //}
548  L /**
549  L  * @brief
550  L  */
551  //void
552  //{
553  //}
554  L /**
555  L  * @brief
556  L  */
557  //void
558  //{
559  //}
560  L /**
561  L  * @brief
562  L  */
563  //void
564  //{
565  //}
566  L /**
567  L  * @brief
568  L  */
569  //void
570  //{
571  //}
572  L /**
573  L  * @brief
574  L  */
575  //void
576  //{
577  //}
578  L /**
579  L  * @brief
580  L  */
581  //void
582  //{
583  //}
584  L /**
585  L  * @brief
586  L  */
587  //void
588  //{
589  //}
590  L /**
591  L  * @brief
592  L  */
593  //void
594  //{
595  //}
596  L /**
597  L  * @brief
598  L  */
599  //void
600  //{
601  //}
602  L /**
603  L  * @brief
604  L  */
605  //void
606  //{
607  //}
608  L /**
609  L  * @brief
610  L  */
611  //void
612  //{
613  //}
614  L /**
615  L  * @brief
616  L  */
617  //void
618  //{
619  //}
620  L /**
621  L  * @brief
622  L  */
623  //void
624  //{
625  //}
626  L /**
627  L  * @brief
628  L  */
629  //void
630  //{
631  //}
632  L /**
633  L  * @brief
634  L  */
635  //void
636  //{
637  //}
638  L /**
639  L  * @brief
640  L  */
641  //void
642  //{
643  //}
644  L /**
645  L  * @brief
646  L  */
647  //void
648  //{
649  //}
650  L /**
651  L  * @brief
652  L  */
653  //void
654  //{
655  //}
656  L /**
657  L  * @brief
658  L  */
659  //void
660  //{
661  //}
662  L /**
663  L  * @brief
664  L  */
665  //void
666  //{
667  //}
668  L /**
669  L  * @brief
670  L  */
671  //void
672  //{
673  //}
674  L /**
675  L  * @brief
676  L  */
677  //void
678  //{
679  //}
680  L /**
681  L  * @brief
682  L  */
683  //void
684  //{
685  //}
686  L /**
687  L  * @brief
688  L  */
689  //void
690  //{
691  //}
692  L /**
693  L  * @brief
694  L  */
695  //void
696  //{
697  //}
698  L /**
699  L  * @brief
700  L  */
701  //void
702  //{
703  //}
704  L /**
705  L  * @brief
706  L  */
707  //void
708  //{
709  //}
710  L /**
711  L  * @brief
712  L  */
713  //void
714  //{
715  //}
716  L /**
717  L  * @brief
718  L  */
719  //void
720  //{
721  //}
722  L /**
723  L  * @brief
724  L  */
725  //void
726  //{
727  //}
728  L /**
729  L  * @brief
730  L  */
731  //void
732  //{
733  //}
734  L /**
735  L  * @brief
736  L  */
737  //void
738  //{
739  //}
740  L /**
741  L  * @brief
742  L  */
743  //void
744  //{
745  //}
746  L /**
747  L  * @brief
748  L  */
749  //void
750  //{
751  //}
752  L /**
753  L  * @brief
754  L  */
755  //void
756  //{
757  //}
758  L /**
759  L  * @brief
760  L  */
761  //void
762  //{
763  //}
764  L /**
765  L  * @brief
766  L  */
767  //void
768  //{
769  //}
770  L /**
771  L  * @brief
772  L  */
773  //void
774  //{
775  //}
776  L /**
777  L  * @brief
778  L  */
779  //void
780  //{
781  //}
782  L /**
783  L  * @brief
784  L  */
785  //void
786  //{
787  //}
788  L /**
789  L  * @brief
790  L  */
791  //void
792  //{
793  //}
794  L /**
795  L  * @brief
796  L  */
797  //void
798  //{
799  //}
800  L /**
801  L  * @brief
802  L  */
803  //void
804  //{
805  //}
806  L /**
807  L  * @brief
808  L  */
809  //void
810  //{
811  //}
812  L /**
813  L  * @brief
814  L  */
815  //void
816  //{
817  //}
818  L /**
819  L  * @brief
820  L  */
821  //void
822  //{
823  //}
824  L /**
825  L  * @brief
826  L  */
827  //void
828  //{
829  //}
830  L /**
831  L  * @brief
832  L  */
833  //void
834  //{
835  //}
836  L /**
837  L  * @brief
838  L  */
839  //void
840  //{
841  //}
842  L /**
843  L  * @brief
844  L  */
845  //void
846  //{
847  //}
848  L /**
849  L  * @brief
850  L  */
851  //void
852  //{
853  //}
854  L /**
855  L  * @brief
856  L  */
857  //void
858  //{
859  //}
860  L /**
861  L  * @brief
862  L  */
863  //void
864  //{
865  //}
866  L /**
867  L  * @brief
868  L  */
869  //void
870  //{
871  //}
872  L /**
873  L  * @brief
874  L  */
875  //void
876  //{
877  //}
878  L /**
879  L  * @brief
880  L  */
881  //void
882  //{
883  //}
884  L /**
885  L  * @brief
886  L  */
887  //void
888  //{
889  //}
890  L /**
891  L  * @brief
892  L  */
893  //void
894  //{
895  //}
896  L /**
897  L  * @brief
898  L  */
899  //void
900  //{
901  //}
902  L /**
903  L  * @brief
904  L  */
905  //void
906  //{
907  //}
908  L /**
909  L  * @brief
910  L  */
911  //void
912  //{
913  //}
914  L /**
915  L  * @brief
916  L  */
917  //void
918  //{
919  //}
920  L /**
921  L  * @brief
922  L  */
923  //void
924  //{
925  //}
926  L /**
927  L  * @brief
928  L  */
929  //void
930  //{
931  //}
932  L /**
933  L  * @brief
934  L  */
935  //void
936  //{
937  //}
938  L /**
939  L  * @brief
940  L  */
941  //void
942  //{
943  //}
944  L /**
945  L  * @brief
946  L  */
947  //void
948  //{
949  //}
950  L /**
951  L  * @brief
952  L  */
953  //void
954  //{
955  //}
956  L /**
957  L  * @brief
958  L  */
959  //void
960  //{
961  //}
962  L /**
963  L  * @brief
964  L  */
965  //void
966  //{
967  //}
968  L /**
969  L  * @brief
970  L  */
971  //void
972  //{
973  //}
974  L /**
975  L  * @brief
976  L  */
977  //void
978  //{
979  //}
980  L /**
981  L  * @brief
982  L  */
983  //void
984  //{
985  //}
986  L /**
987  L  * @brief
988  L  */
989  //void
990  //{
991  //}
992  L /**
993  L  * @brief
994  L  */
995  //void
996  //{
997  //}
998  L /**
999  L  * @brief
1000  L  */

```

```

277  */
278  //void uartInit(void)
279  //{
280  //  HAL_UART_Receive_IT(&huart2, (uint8_t *)&aRxBuffer, 1); //开启下一次接收中断
281  //}
282
283  /**

```

修改 mcuRestart ()

```

211  */
212  void mcuRestart(void)
213  {
214      __set_FAULTMASK(1);
215      HAL_NVIC_SystemReset();
216  }
217
218  /**@ */
219

```

```

211  */
212  void mcuRestart(void)
213  {
214      __set_FAULTMASK(1);
215      NVIC_SystemReset();
216  }
217
218  /**@ */

```

修改 uartWrite () 函数

```

291  */
292  int32_t uartWrite(uint8_t *buf, uint32_t len)
293  {
294      uint8_t crc[1] = {0x55};
295      uint32_t i = 0;
296
297      if(NULL == buf)
298      {
299          return -1;
300      }
301
302      for(i=0; i<len; i++)
303      {
304          //  HAL_UART_Transmit_IT(&huart2, (uint8_t *)&buf[i], 1);
305          //  while (huart2.gState != HAL_UART_STATE_READY) //Loop until the end of transmission
306
307          //  if(i >=2 && buf[i] == 0xFF)
308          //  {
309              //  HAL_UART_Transmit_IT(&huart2, (uint8_t *)&crc, 1);
310              //  while (huart2.gState != HAL_UART_STATE_READY) //Loop until the end of transmission
311          //  }
312          USART_SendData(USART3, buf[i]);
313          while(USART_GetFlagStatus(USART3, USART_FLAG_TC) == RESET); //循环发送, 直到发送完毕
314          if(i >=2 && buf[i] == 0xFF)
315          {
316              USART_SendData(USART3, 0x55);
317              while(USART_GetFlagStatus(USART3, USART_FLAG_TC) == RESET); //循环发送, 直到发送完毕
318          }
319      }
320  }
321

```


3.product.h

改开始几行即可

```
15 | *****
16 | #ifndef _GIZWITS_PRODUCT_H
17 | #define _GIZWITS_PRODUCT_H
18 |
19 | #ifdef __cplusplus
20 | extern "C" {
21 | #endif
22 |
23 | #include <stdint.h>
24 | #include <stm32f10x.h>
25 | #include "gizwits_protocol.h"
26 |
27 | /**
28 |  * MCH software version
```

4. gizwits_protocol.h

```
275 |
276 | /** User Area Device State Structure */
277 | typedef struct {
278 |     bool valueLEDonoff;
279 | } dataPoint_t;
280 |
```

5.main 函数

定义 userHandle() 初始化各外设和协议

```
11  /* 用户区当前设备状态结构体*/
12  dataPoint_t currentDataPoint;
13
14  //协议初始化
15  void Gizwits_Init(void)
16  {
17
18      TIM3_Int_Init(9, 7199); //1MS系统定时
19      usart3_init(9600); //WIFI初始化
20      memset((uint8_t*)&currentDataPoint, 0, sizeof(dataPoint_t)); //设备状态结构体初始化
21      gizwitsInit(); //缓冲区初始化
22  }
23
24  //数据采集
25  void userHandle(void)
26  {
27
28      //判断当前LED1开关量
29      if(LED1==0)
30          currentDataPoint.valueLEDOnoff = 1;
31      else
32          currentDataPoint.valueLEDOnoff = 0;
33  }
34
35  //主函数
36  int main(void)
37  {
38      int key;
39      delay_init(); //延时函数初始化
40      NVIC_PriorityGroupConfig(NVIC_PriorityGroup_2); //设置NVIC中断分组2: 2位抢占优先级
41      uart_init(115200); //串口初始化为115200
42      LED_Init(); //LED端口初始化
43      KEY_Init(); //按键初始化
44      Gizwits_Init(); //协议初始化
45      printf("-----机智云-协议移植LED测试实验-----\r\n");
46      printf("KEY1: AirLink连接模式\t KEY_UP: 复位\r\n\r\n");
47      while(1)
48      {
49          userHandle(); //用户采集
50
51          gizwitsHandle((dataPoint_t *)&currentDataPoint); //协议处理
52      }
```

6.下载验证

同原文档

作者: yyc

特别鸣谢: 爱吃栗子饼