

《嵌入式系统原理与实践》作业

10213903403 岳锦鹏
10225001410 朱宇笑

2024 年 12 月 30 日

目录

第七次实验

- 端口修改
- 代码修改
- 实验结果

第七次实验

端口修改

代码修改

实验结果

端口设置如图。

STM32CubeMX exp7.ioc: STM32G473RCTx

File Window Help Hello Yuxiao GENERATE CODE

Home > STM32G473RCTx > exp7.ioc - Pinout & Configuration >

Pinout & Configuration Clock Configuration Project Manager Tools

GPIO Mode and Configuration

Group By Peripherals

Search Signals Search (Ctrl+F)

Show only Modified Pins

Pin No.	Signal on	GPIO out...	GPIO mode	GPIO Pul...	Maximum...	Fast Mode	User Label	Modified
PA0	n/a	High	Output P...	Pull-up	Low	n/a	SegLed...	<input checked="" type="checkbox"/>
PA1	n/a	High	Output P...	Pull-up	Low	n/a		<input checked="" type="checkbox"/>
PA2	n/a	High	Output P...	Pull-up	Low	n/a		<input checked="" type="checkbox"/>
PA3	n/a	High	Output P...	Pull-up	Low	n/a		<input checked="" type="checkbox"/>
PA4	n/a	High	Output P...	Pull-up	Low	n/a		<input checked="" type="checkbox"/>
PA5	n/a	High	Output P...	Pull-up	Low	n/a		<input checked="" type="checkbox"/>
PA6	n/a	High	Output P...	Pull-up	Low	n/a		<input checked="" type="checkbox"/>
PA7	n/a	High	Output P...	Pull-up	Low	n/a		<input checked="" type="checkbox"/>
PB0	n/a	n/a	Input mode	Pull-up	n/a	n/a	KeyLine	<input checked="" type="checkbox"/>
PB1	n/a	n/a	Input mode	Pull-up	n/a	n/a		<input checked="" type="checkbox"/>
PB2	n/a	n/a	Input mode	Pull-up	n/a	n/a		<input checked="" type="checkbox"/>
PB3	n/a	n/a	Input mode	Pull-up	n/a	n/a		<input checked="" type="checkbox"/>

Select Pins from table to configure them. Multiple selection is Allowed.

Pinout view System view

STM32G473RCTx LQFP64

第七次实验

端口修改

代码修改

实验结果

添加 tim.h:

```
20  /* Define to prevent recursive inclusion -----*/ 39   /* USER CODE BEGIN Private defines */
21  #ifndef __TIM_H__ 40
22  #define __TIM_H__ 41   /* USER CODE END Private defines */
23
24  #ifdef __cplusplus 42
25  extern "C" { 43   void MX_TIM3_Init(void);
26  #endif 44   void MX_TIM4_Init(void);
27
28  /* Includes -----*/ 45
29  #include "main.h" 46   void HAL_TIM_MspPostInit(TIM_HandleTypeDef *htim);
30
31  /* USER CODE BEGIN Includes */ 47
32
33  /* USER CODE END Includes */ 48   /* USER CODE BEGIN Prototypes */
34
35  extern TIM_HandleTypeDef htim3; 49
36
37  extern TIM_HandleTypeDef htim4; 50   /* USER CODE END Prototypes */
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52  #ifdef __cplusplus
53  }
54  #endif
55
56  #endif /* __TIM_H__ */
```

添加 tim.c:

```
20  /* Includes -----*/      36  /* USER CODE END TIM3_Init_0 */  
21  #include "tim.h"          37  
22  /* USER CODE BEGIN 0 */     38  TIM_ClockConfigTypeDef sClockSourceConfig = {0};  
23  /* USER CODE BEGIN 0 */     39  TIM_MasterConfigTypeDef sMasterConfig = {0};  
24  
25  /* USER CODE END 0 */      40  
26  
27  TIM_HandleTypeDef htim3;    41  /* USER CODE BEGIN TIM3_Init_1 */  
28  TIM_HandleTypeDef htim4;    42  
29  
30  /* TIM3 init function */  43  /* USER CODE END TIM3_Init_1 */  
31  void MX_TIM3_Init(void)    44  htim3.Instance = TIM3;  
32  {                         45  htim3.Init.Prescaler = 0;  
33  
34  /* USER CODE BEGIN TIM3_Init_0 */ 46  htim3.Init.CounterMode =  
35                                         47  → TIM_COUNTERMODE_CENTERALIGNED1;  
                                         48  htim3.Init.Period = 9;  
                                         49  htim3.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;  
                                         → htim3.Init.AutoReloadPreload =  
                                         → TIM_AUTORELOAD_PRELOAD_ENABLE;
```

添加 tim.c:

```
58     if (HAL_TIM_ConfigClockSource(&htim3,  
59         &sClockSourceConfig) != HAL_OK)  
60     {  
61         Error_Handler();  
62     sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;  
63     sMasterConfig.MasterSlaveMode =  
64         TIM_MASTERSLAVEMODE_DISABLE;  
65     if (HAL_TIMEx_MasterConfigSynchronization(&htim3,  
66         &sMasterConfig) != HAL_OK)  
67     {  
68         Error_Handler();  
69     /* USER CODE BEGIN TIM3_Init_2 */  
70     /* USER CODE END TIM3_Init_2 */  
71     }  
72     }  
73     /* TIM4 init function */  
74     void MX_TIM4_Init(void)  
75     {  
76         /* USER CODE BEGIN TIM4_Init_0 */  
77     /* USER CODE END TIM4_Init_0 */  
78     TIM_ClockConfigTypeDef sClockSourceConfig = {0};  
79     TIM_MasterConfigTypeDef sMasterConfig = {0};  
80     TIM_OC_InitTypeDef sConfigOC = {0};  
81     }  
82     }  
83     }  
84     }  
85     }  
86     }  
87     }  
88     }  
89     }  
90     }  
91     }  
92     }  
93     }  
94     }  
95     }  
96     }  
97     }  
98     }  
99     }  
100    }  
101    }  
102    }  
103    }  
104    }  
105    }  
106    }  
107    }  
108    }  
109    }  
110    }  
111    }  
112    }  
113    }  
114    }  
115    }  
116    }  
117    }  
118    }  
119    }  
120    }  
121    }  
122    }  
123    }  
124    }  
125    }  
126    }  
127    }  
128    }  
129    }  
130    }  
131    }  
132    }  
133    }  
134    }  
135    }  
136    }  
137    }  
138    }  
139    }  
140    }  
141    }  
142    }  
143    }  
144    }  
145    }  
146    }  
147    }  
148    }  
149    }  
150    }  
151    }  
152    }  
153    }  
154    }  
155    }  
156    }  
157    }  
158    }  
159    }  
160    }  
161    }  
162    }  
163    }  
164    }  
165    }  
166    }  
167    }  
168    }  
169    }  
170    }  
171    }  
172    }  
173    }  
174    }  
175    }  
176    }  
177    }  
178    }  
179    }  
180    }  
181    }  
182    }  
183    }  
184    }  
185    }  
186    }  
187    }  
188    }  
189    }  
190    }  
191    }  
192    }  
193    }  
194    }  
195    }  
196    }  
197    }  
198    }  
199    }  
200    }  
201    }  
202    }  
203    }  
204    }  
205    }  
206    }  
207    }  
208    }  
209    }  
210    }  
211    }  
212    }  
213    }  
214    }  
215    }  
216    }  
217    }  
218    }  
219    }  
220    }  
221    }  
222    }  
223    }  
224    }  
225    }  
226    }  
227    }  
228    }  
229    }  
230    }  
231    }  
232    }  
233    }  
234    }  
235    }  
236    }  
237    }  
238    }  
239    }  
240    }  
241    }  
242    }  
243    }  
244    }  
245    }  
246    }  
247    }  
248    }  
249    }  
250    }  
251    }  
252    }  
253    }  
254    }  
255    }  
256    }  
257    }  
258    }  
259    }  
260    }  
261    }  
262    }  
263    }  
264    }  
265    }  
266    }  
267    }  
268    }  
269    }  
270    }  
271    }  
272    }  
273    }  
274    }  
275    }  
276    }  
277    }  
278    }  
279    }  
280    }  
281    }  
282    }  
283    }  
284    }  
285    }  
286    }  
287    }  
288    }  
289    }  
290    }  
291    }  
292    }  
293    }  
294    }  
295    }  
296    }  
297    }  
298    }  
299    }  
300    }  
301    }  
302    }  
303    }  
304    }  
305    }  
306    }  
307    }  
308    }  
309    }  
310    }  
311    }  
312    }  
313    }  
314    }  
315    }  
316    }  
317    }  
318    }  
319    }  
320    }  
321    }  
322    }  
323    }  
324    }  
325    }  
326    }  
327    }  
328    }  
329    }  
330    }  
331    }  
332    }  
333    }  
334    }  
335    }  
336    }  
337    }  
338    }  
339    }  
340    }  
341    }  
342    }  
343    }  
344    }  
345    }  
346    }  
347    }  
348    }  
349    }  
350    }  
351    }  
352    }  
353    }  
354    }  
355    }  
356    }  
357    }  
358    }  
359    }  
360    }  
361    }  
362    }  
363    }  
364    }  
365    }  
366    }  
367    }  
368    }  
369    }  
370    }  
371    }  
372    }  
373    }  
374    }  
375    }  
376    }  
377    }  
378    }  
379    }  
380    }  
381    }  
382    }  
383    }  
384    }  
385    }  
386    }  
387    }  
388    }  
389    }  
390    }  
391    }  
392    }  
393    }  
394    }  
395    }  
396    }  
397    }  
398    }  
399    }  
400    }  
401    }  
402    }  
403    }  
404    }  
405    }  
406    }  
407    }  
408    }  
409    }  
410    }  
411    }  
412    }  
413    }  
414    }  
415    }  
416    }  
417    }  
418    }  
419    }  
420    }  
421    }  
422    }  
423    }  
424    }  
425    }  
426    }  
427    }  
428    }  
429    }  
430    }  
431    }  
432    }  
433    }  
434    }  
435    }  
436    }  
437    }  
438    }  
439    }  
440    }  
441    }  
442    }  
443    }  
444    }  
445    }  
446    }  
447    }  
448    }  
449    }  
450    }  
451    }  
452    }  
453    }  
454    }  
455    }  
456    }  
457    }  
458    }  
459    }  
460    }  
461    }  
462    }  
463    }  
464    }  
465    }  
466    }  
467    }  
468    }  
469    }  
470    }  
471    }  
472    }  
473    }  
474    }  
475    }  
476    }  
477    }  
478    }  
479    }  
480    }  
481    }  
482    }  
483    }  
484    }  
485    }  
486    }  
487    }  
488    }  
489    }  
490    }  
491    }  
492    }  
493    }  
494    }  
495    }  
496    }  
497    }  
498    }  
499    }  
500    }  
501    }  
502    }  
503    }  
504    }  
505    }  
506    }  
507    }  
508    }  
509    }  
510    }  
511    }  
512    }  
513    }  
514    }  
515    }  
516    }  
517    }  
518    }  
519    }  
520    }  
521    }  
522    }  
523    }  
524    }  
525    }  
526    }  
527    }  
528    }  
529    }  
530    }  
531    }  
532    }  
533    }  
534    }  
535    }  
536    }  
537    }  
538    }  
539    }  
540    }  
541    }  
542    }  
543    }  
544    }  
545    }  
546    }  
547    }  
548    }  
549    }  
550    }  
551    }  
552    }  
553    }  
554    }  
555    }  
556    }  
557    }  
558    }  
559    }  
560    }  
561    }  
562    }  
563    }  
564    }  
565    }  
566    }  
567    }  
568    }  
569    }  
570    }  
571    }  
572    }  
573    }  
574    }  
575    }  
576    }  
577    }  
578    }  
579    }  
580    }  
581    }  
582    }  
583    }  
584    }  
585    }  
586    }  
587    }  
588    }  
589    }  
590    }  
591    }  
592    }  
593    }  
594    }  
595    }  
596    }  
597    }  
598    }  
599    }  
600    }  
601    }  
602    }  
603    }  
604    }  
605    }  
606    }  
607    }  
608    }  
609    }  
610    }  
611    }  
612    }  
613    }  
614    }  
615    }  
616    }  
617    }  
618    }  
619    }  
620    }  
621    }  
622    }  
623    }  
624    }  
625    }  
626    }  
627    }  
628    }  
629    }  
630    }  
631    }  
632    }  
633    }  
634    }  
635    }  
636    }  
637    }  
638    }  
639    }  
640    }  
641    }  
642    }  
643    }  
644    }  
645    }  
646    }  
647    }  
648    }  
649    }  
650    }  
651    }  
652    }  
653    }  
654    }  
655    }  
656    }  
657    }  
658    }  
659    }  
660    }  
661    }  
662    }  
663    }  
664    }  
665    }  
666    }  
667    }  
668    }  
669    }  
670    }  
671    }  
672    }  
673    }  
674    }  
675    }  
676    }  
677    }  
678    }  
679    }  
680    }  
681    }  
682    }  
683    }  
684    }  
685    }  
686    }  
687    }  
688    }  
689    }  
690    }  
691    }  
692    }  
693    }  
694    }  
695    }  
696    }  
697    }  
698    }  
699    }  
700    }  
701    }  
702    }  
703    }  
704    }  
705    }  
706    }  
707    }  
708    }  
709    }  
710    }  
711    }  
712    }  
713    }  
714    }  
715    }  
716    }  
717    }  
718    }  
719    }  
720    }  
721    }  
722    }  
723    }  
724    }  
725    }  
726    }  
727    }  
728    }  
729    }  
730    }  
731    }  
732    }  
733    }  
734    }  
735    }  
736    }  
737    }  
738    }  
739    }  
740    }  
741    }  
742    }  
743    }  
744    }  
745    }  
746    }  
747    }  
748    }  
749    }  
750    }  
751    }  
752    }  
753    }  
754    }  
755    }  
756    }  
757    }  
758    }  
759    }  
760    }  
761    }  
762    }  
763    }  
764    }  
765    }  
766    }  
767    }  
768    }  
769    }  
770    }  
771    }  
772    }  
773    }  
774    }  
775    }  
776    }  
777    }  
778    }  
779    }  
780    }  
781    }  
782    }  
783    }  
784    }  
785    }  
786    }  
787    }  
788    }  
789    }  
790    }  
791    }  
792    }  
793    }  
794    }  
795    }  
796    }  
797    }  
798    }  
799    }  
800    }  
801    }  
802    }  
803    }  
804    }  
805    }  
806    }  
807    }  
808    }  
809    }  
810    }  
811    }  
812    }  
813    }  
814    }  
815    }  
816    }  
817    }  
818    }  
819    }  
820    }  
821    }  
822    }  
823    }  
824    }  
825    }  
826    }  
827    }  
828    }  
829    }  
830    }  
831    }  
832    }  
833    }  
834    }  
835    }  
836    }  
837    }  
838    }  
839    }  
840    }  
841    }  
842    }  
843    }  
844    }  
845    }  
846    }  
847    }  
848    }  
849    }  
850    }  
851    }  
852    }  
853    }  
854    }  
855    }  
856    }  
857    }  
858    }  
859    }  
860    }  
861    }  
862    }  
863    }  
864    }  
865    }  
866    }  
867    }  
868    }  
869    }  
870    }  
871    }  
872    }  
873    }  
874    }  
875    }  
876    }  
877    }  
878    }  
879    }  
880    }  
881    }  
882    }  
883    }  
884    }  
885    }  
886    }  
887    }  
888    }  
889    }  
890    }  
891    }  
892    }  
893    }  
894    }  
895    }  
896    }  
897    }  
898    }  
899    }  
900    }  
901    }  
902    }  
903    }  
904    }  
905    }  
906    }  
907    }  
908    }  
909    }  
910    }  
911    }  
912    }  
913    }  
914    }  
915    }  
916    }  
917    }  
918    }  
919    }  
920    }  
921    }  
922    }  
923    }  
924    }  
925    }  
926    }  
927    }  
928    }  
929    }  
930    }  
931    }  
932    }  
933    }  
934    }  
935    }  
936    }  
937    }  
938    }  
939    }  
940    }  
941    }  
942    }  
943    }  
944    }  
945    }  
946    }  
947    }  
948    }  
949    }  
950    }  
951    }  
952    }  
953    }  
954    }  
955    }  
956    }  
957    }  
958    }  
959    }  
960    }  
961    }  
962    }  
963    }  
964    }  
965    }  
966    }  
967    }  
968    }  
969    }  
970    }  
971    }  
972    }  
973    }  
974    }  
975    }  
976    }  
977    }  
978    }  
979    }  
980    }  
981    }  
982    }  
983    }  
984    }  
985    }  
986    }  
987    }  
988    }  
989    }  
990    }  
991    }  
992    }  
993    }  
994    }  
995    }  
996    }  
997    }  
998    }  
999    }  
1000    }
```

添加 tim.c:

```
85  /* USER CODE BEGIN TIM4_Init 1 */          106   }
86
87  /* USER CODE END TIM4_Init 1 */          107   sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
88  htim4.Instance = TIM4;                   108   sMasterConfig.MasterSlaveMode =
89  htim4.Init.Prescaler = 170-1;           109   → TIM_MASTERSLAVEMODE_DISABLE;
90  htim4.Init.CounterMode = TIM_COUNTERMODE_UP; 110
91  htim4.Init.Period = 99;                 111
92  htim4.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1; 112
93  htim4.Init.AutoReloadPreload =          113   sConfigOC.OCMode = TIM_OCMODE_PWM1;
94  → TIM_AUTORELOAD_PRELOAD_ENABLE;       114   sConfigOC.Pulse = 50;
95  if (HAL_TIM_Base_Init(&htim4) != HAL_OK)  115   sConfigOC.OCPolarity = TIM_OCPOLARITY_HIGH;
96  {                                         116   sConfigOC.OCFastMode = TIM_OCFAST_DISABLE;
97      Error_Handler();                  117   if (HAL_TIM_PWM_ConfigChannel(&htim4, &sConfigOC,
98  }                                         118   → TIM_CHANNEL_1) != HAL_OK)
99  sClockSourceConfig.ClockSource =        119   {
100 → TIM_CLOCKSOURCE_INTERNAL;           120      Error_Handler();
101 if (HAL_TIM_ConfigClockSource(&htim4, 121  /* USER CODE BEGIN TIM4_Init 2 */
102 → &sClockSourceConfig) != HAL_OK)     122
103 {                                     123  /* USER CODE END TIM4_Init 2 */
104     Error_Handler();                124  HAL_TIM_MspPostInit(&htim4);
105 }                                     125
106   Error_Handler();                  126 }
```

添加 tim.c:

```
128 void HAL_TIM_Base_MspInit(TIM_HandleTypeDef*          149      HAL_GPIO_Init(GPIOD, &GPIO_InitStruct);
    __ tim_baseHandle)           150
{                                         151      /* TIM3 interrupt Init */
130                                         152      HAL_NVIC_SetPriority(TIM3_IRQn, 0, 0);
131     GPIO_InitTypeDef GPIO_InitStruct = {0}; 153      HAL_NVIC_EnableIRQ(TIM3_IRQn);
132     if(tim_baseHandle->Instance==TIM3)        154      /* USER CODE BEGIN TIM3_MspInit 1 */
133     {                                         155
134         /* USER CODE BEGIN TIM3_MspInit 0 */       156      /* USER CODE END TIM3_MspInit 1 */
135                                         157
136         /* USER CODE END TIM3_MspInit 0 */       158
137         /* TIM3 clock enable */                 159
138         __HAL_RCC_TIM3_CLK_ENABLE();           160      /* USER CODE BEGIN TIM4_MspInit 0 */
139                                         161
140         __HAL_RCC_GPIOD_CLK_ENABLE();          162      /* USER CODE END TIM4_MspInit 0 */
141         /**TIM3 GPIO Configuration           163      /* TIM4 clock enable */
142         PD2      -----> TIM3_ETR           164      __HAL_RCC_TIM4_CLK_ENABLE();
143         */                                         165      /* USER CODE BEGIN TIM4_MspInit 1 */
144     GPIO_InitStruct.Pin = GPIO_PIN_2;           166
145     GPIO_InitStruct.Mode = GPIO_MODE_AF_PP;    167      /* USER CODE END TIM4_MspInit 1 */
146     GPIO_InitStruct.Pull = GPIO_NOPULL;        168
147     GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW; 169
148     GPIO_InitStruct.Alternate = GPIO_AF2_TIM3;
```

添加 tim.c:

```
170 void HAL_TIM_MspPostInit(TIM_HandleTypeDef* timHandle)84
171 {
172     GPIO_InitTypeDef GPIO_InitStruct = {0};
173     if(timHandle->Instance==TIM4)185
174     {
175         /* USER CODE BEGIN TIM4_MspPostInit 0 */
176
177         /* USER CODE END TIM4_MspPostInit 0 */
178
179         __HAL_RCC_GPIOA_CLK_ENABLE();
180         /**TIM4 GPIO Configuration
181          PA11      -----> TIM4_CH1
182
183     */
184
185     GPIO_InitStruct.Pin = GPIO_PIN_11;
186     GPIO_InitStruct.Mode = GPIO_MODE_AF_PP;
187     GPIO_InitStruct.Pull = GPIO_NOPULL;
188     GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
189     GPIO_InitStruct.Alternate = GPIO_AF10_TIM4;
190     HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);
191
192     /* USER CODE BEGIN TIM4_MspPostInit 1 */
193
194     /* USER CODE END TIM4_MspPostInit 1 */
195
196 }
```

添加 tim.c:

```
198 void HAL_TIM_Base_MspDeInit(TIM_HandleTypeDef*  
  __ tim_baseHandle)  
199 {  
200     if(tim_baseHandle->Instance==TIM3)  
201     {  
202         /* USER CODE BEGIN TIM3_MspDeInit 0 */  
203         /* USER CODE END TIM3_MspDeInit 0 */  
204         /* Peripheral clock disable */  
205         __HAL_RCC_TIM3_CLK_DISABLE();  
206  
207         /**TIM3 GPIO Configuration  
208          PD2      -----> TIM3_ETR  
209          */  
210         HAL_GPIO_DeInit(GPIOD, GPIO_PIN_2);  
211  
212         /* TIM3 interrupt Deinit */  
213  
214  
215         HAL_NVIC_DisableIRQ(TIM3_IRQn);  
216         /* USER CODE BEGIN TIM3_MspDeInit 1 */  
217  
218         /* USER CODE END TIM3_MspDeInit 1 */  
219     }  
220     else if(tim_baseHandle->Instance==TIM4)  
221     {  
222         /* USER CODE BEGIN TIM4_MspDeInit 0 */  
223  
224         /* USER CODE END TIM4_MspDeInit 0 */  
225         /* Peripheral clock disable */  
226         __HAL_RCC_TIM4_CLK_DISABLE();  
227         /* USER CODE BEGIN TIM4_MspDeInit 1 */  
228  
229         /* USER CODE END TIM4_MspDeInit 1 */  
230     }  
231 }
```

main.c:

```
25  /* USER CODE BEGIN Includes */          54  structTime stTime = {  
26  #include "variable.h"                  55  .mSecond = 50,  
27  #include "directkey.h"                 56  .mMinute = 45,  
28  #include "MatrixKey.h"                57  .mHour = 8,  
29  #include "SegLed.h"                   58  .mTimeCount = 0,  
30  /* USER CODE END Includes */           59  .bSecondIsOk = 0,  
                                         60  .mTenMilCount = 0,  
                                         61  .bTenMillIsOk = 0  
49  /* USER CODE BEGIN PV */            62  };  
50  stSysTickTimer sSysTickTimer = {        63  /* USER CODE END PV */  
51  0, 0, 0, 0                           64  /* Private function prototypes -----*/  
52  };                                     65  void SystemClock_Config(void);  
53  uint8_t tempValue;
```

main.c:

```
76  /**
77   * @brief  The application entry point.
78   * @retval int
79   */
80 int main(void)
81 {
82
83     /* USER CODE BEGIN 1 */
84     uint8_t KeyValue = 0;
85     /* USER CODE END 1 */
86
87     /* MCU Configuration-----*/
88
89     /* Reset of all peripherals, Initializes the Flash
    ↔ interface and the Systick. */
90     HAL_Init();
91
92     /* USER CODE BEGIN Init */
93
94     /* USER CODE END Init */
95
96     /* Configure the system clock */
97     SystemClock_Config();
98
99     /* USER CODE BEGIN SysInit */
100
101    /* USER CODE END SysInit */
102
103    /* Initialize all configured peripherals */
104    MX_GPIO_Init();
105    MX_TIM3_Init();
106
107    /* USER CODE BEGIN 2 */
108    // FlashLeds_GPIO_Port->ODR |= 0xff01;
109    HAL_TIM_Base_Start_IT(&htim3);
110    TimeToBuff(&stTime);
111    /* USER CODE END 2 */
```

main.c:

```
112 /* Infinite loop */  
113 /* USER CODE BEGIN WHILE */  
114 while (1)  
115 {  
116     /* USER CODE END WHILE */  
117  
118     /* USER CODE BEGIN 3 */  
119     if (stTime.bTenMilIsOk) {  
120         stTime.bTenMilIsOk = 0;  
121     }  
122     if (stTime.bSecondIsOk) {  
123         stTime.bSecondIsOk = 0;  
124         if (++stTime.mSecond >= 60) {  
125             stTime.mSecond = 0;  
126             if (++stTime.mMinute >= 60) {  
127                 stTime.mMinute = 0;  
128                 if (++stTime.mHour >= 24) {  
129                     stTime.mHour = 0;  
130                 }  
131             }  
132         }  
133     }  
134     TimeToBuff(&stTime);  
135     if (sSysTickTimer.bTenMilSecOk) {  
136         sSysTickTimer.bTenMilSecOk = 0;  
137         KeyValue = MatrixKeyScan();  
138         if (KeyValue != NO_KEY) {  
139             for (int i = 0; i < 16; i++) {  
140                 if (KeyValue == KeyTable[i]) {  
141                     tempValue = i;  
142                 }  
143             }  
144             // DispToBuff(tempValue);  
145         }  
146     }  
147     if (sSysTickTimer.bTimeOk) {  
148         sSysTickTimer.bTimeOk = 0;  
149         // TimeToBuff();  
150         HAL_GPIO_TogglePin(LED_GPIO_Port, LED_Pin);  
151     }  
152 }  
153 /* USER CODE END 3 */  
154 }
```

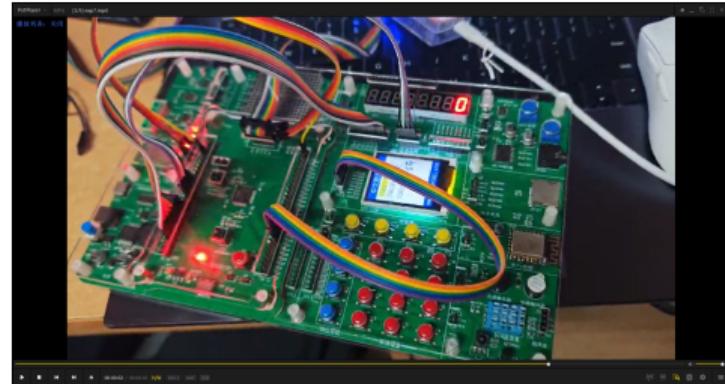
第七次实验

端口修改

代码修改

实验结果

数码管显示 08-45-50，循环闪烁，逻辑与正常定时器相同，但时间变化很慢。



完整视频可以查看：

https://gitea.librastalker.top/423A35C7/STM32CubeMX-Keil_uVision5