Micro-Controller Experiment

Week4

Teacher: 廖裕評 Yu-Ping Liao

TA: 陳大荃 Da-chuan Chen, 陳恩妮 En-ni Chen

Class Rules

1. No drink besides water.
2. Bring a laptop and breadboard if needed.
3. Ask us TAs to sign and borrow development boards. Do not sign or ask others to sign for you without TAs' permission.
4. Arriving 10 minutes after the bell rings will be regarded as absent.
5. If you damage any borrowed equipment, you have to pay for it.

Homework Rules

1. Includes: A. Class content, B. Class exercise, C. Homework (screenshot or video)
2. Editing software: MS PowerPoint
3. File format: PDF
4. Filename: "date_group_studentID_name.pdf", like "0916_第1組_11028XXX_陳OO.pdf"
5. The homework deadline is 23:59 of the day before the next class. If you are late, then your grade will be deducted.

Contact

If you encounter any problems with this class, please get in touch with us with the following

E-mails:

1. Teacher, Prof. Yu-Ping Liao 廖裕評： lyp@cycu.org.tw
2. TA, Da-chuan Chen 陳大荃： dachuan516@gmail.com
3. TA, En-ni Chen 陳恩妮： anna7125867@gmail.com

Or visit 篤信 Lab353 for further questions.

Outline of the Week

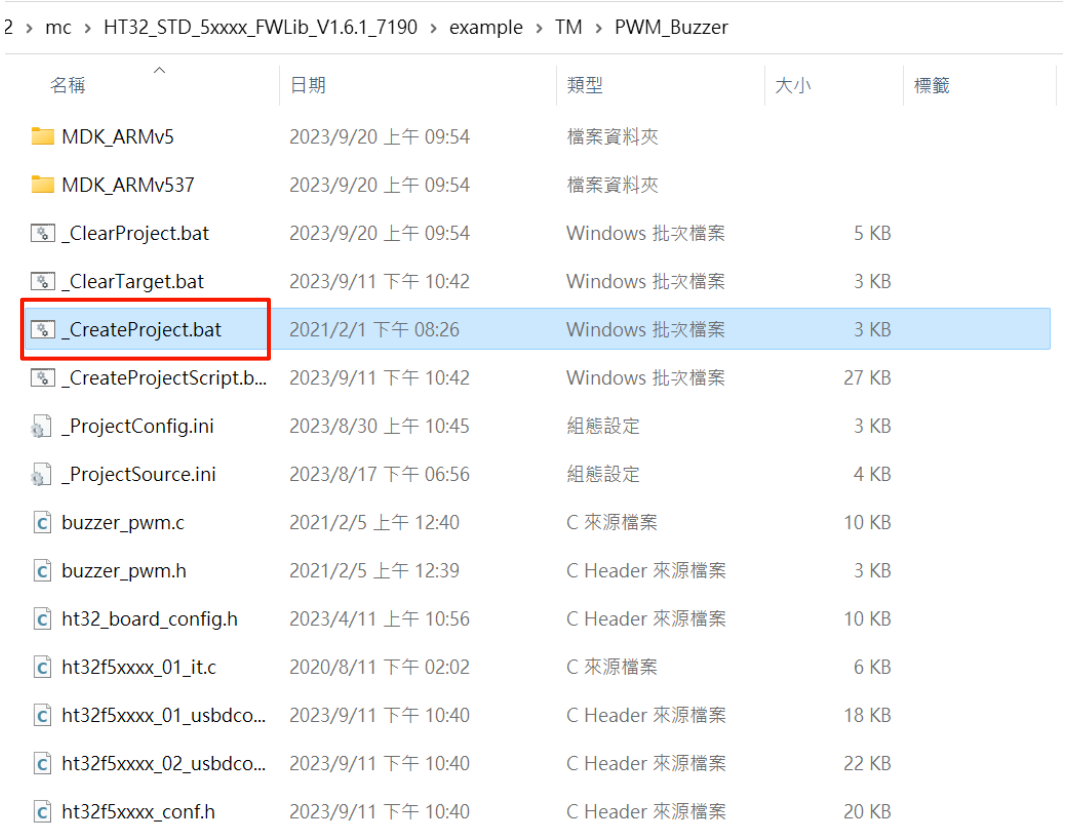
1. Timer Project.
2. Homework 4-1.
3. Homework 4-2.
4. Homework 4-3.

Timer Project

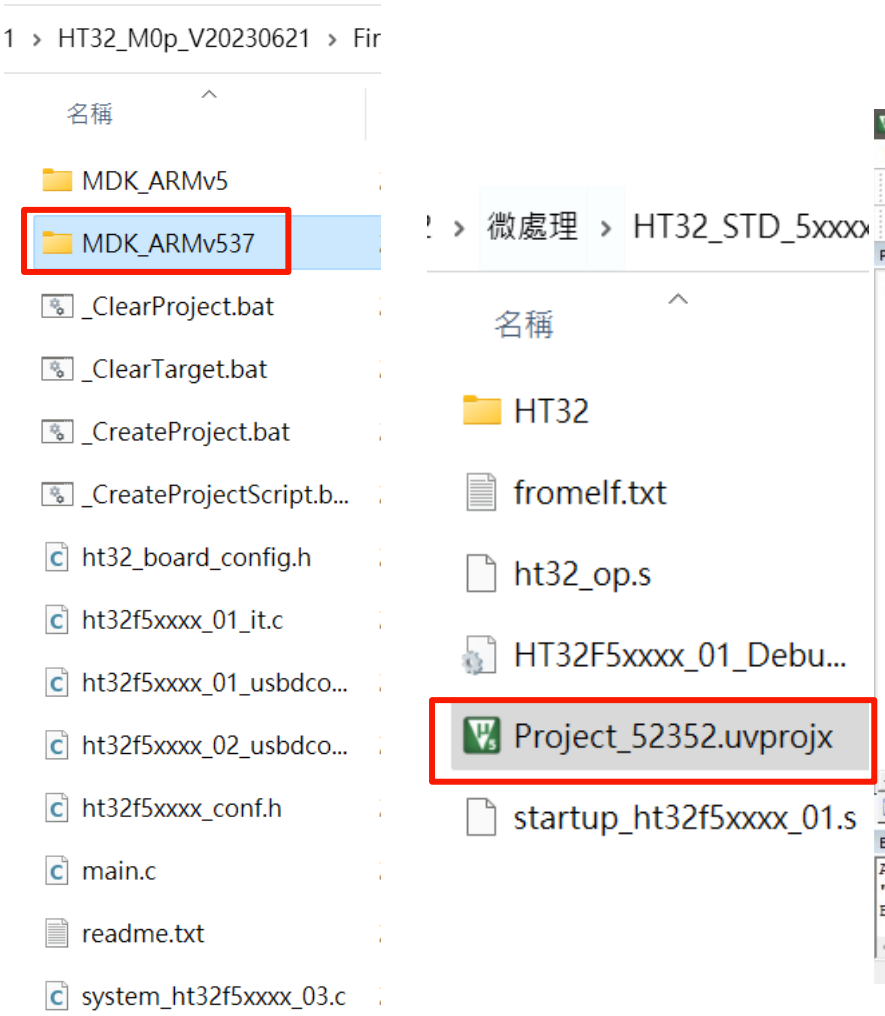
1. Execute “_CreatProject”

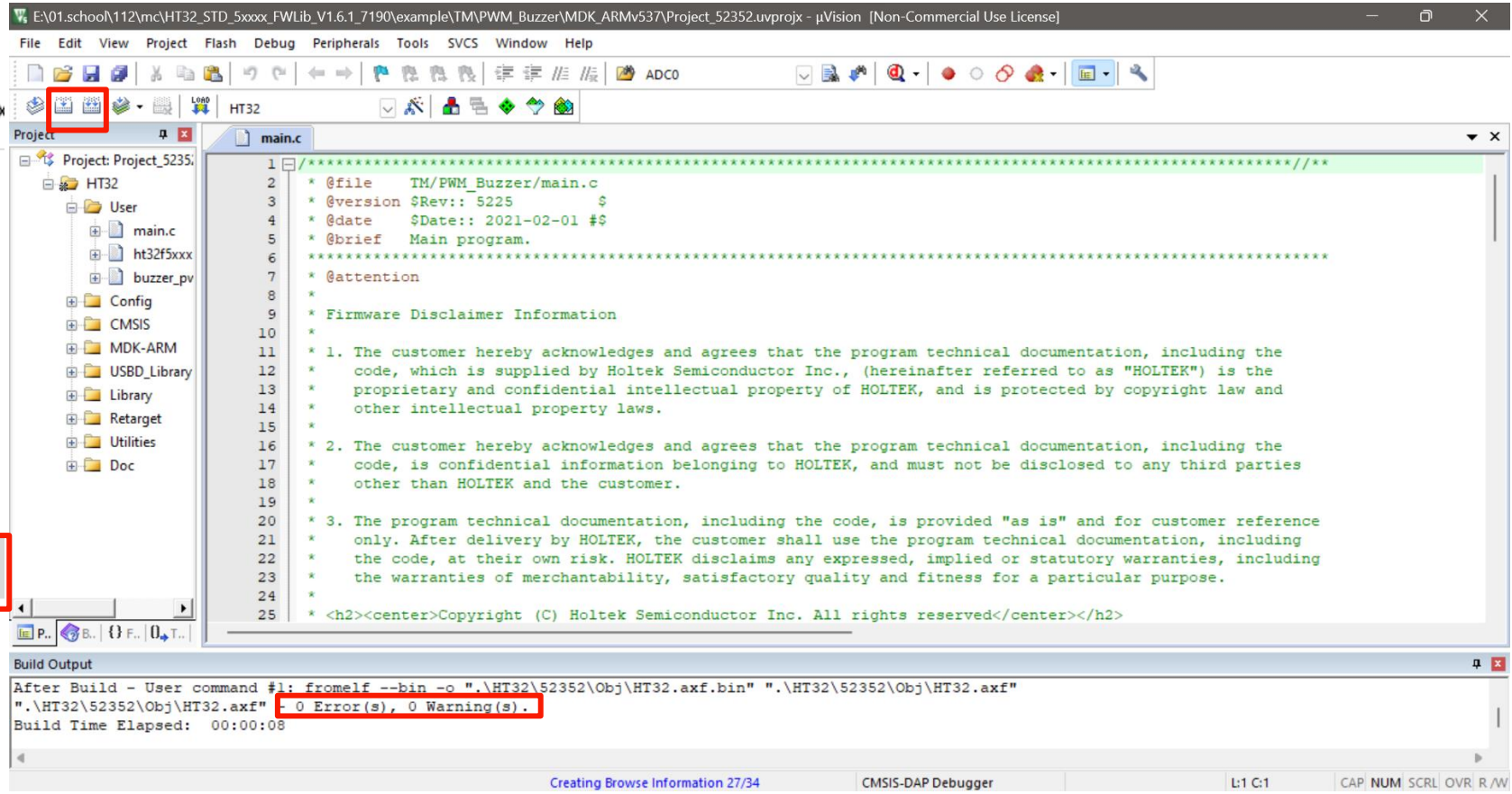
1. Go to “~/HT32_STD_5xxxx_FWLib_V1.5.1_7084/example/TM/PWM_Buzzer”.

2. Double click “_CreateProject.bat”.



2. Launch project





main

```
67  int main(void)
68  {
69    Buzzer_Fun1();
70
71    Buzzer_Fun2();
72
73    Buzzer_Init(0);
74
75    while (1)
76    {
77      Buzzer_PlayTable();
78    }
79  }
```

function1

function2

Initialization Setting

function3

Declare functions in the beginning

```
49  void Buzzer_Fun1(void);
50  void Buzzer_Fun2(void);
51  void Buzzer_PlayTable(void);
```

funciton1

```
85  void Buzzer_Fun1(void)
86  {
87    /* Bee 4 times, 3 kHz, active 50 ms, inactive 50 ms
88    Buzzer_Init(0);
89    Buzzer_Start(4, 3000, 50, 50);
90    while (Buzzer_IsFinish() == FALSE);
91  }
```

```
89    Buzzer_Start(4, 3000, 50, 50);
```

times, frequency, delay time in the beginning(ms), delay time in the end(ms)

funciton2

```
 97  void Buzzer_Fun2(void)
 98  {
 99    /* Bee 2 times, 800 Hz, active 1000 ms, inactive 500 ms
100    Buzzer_Init(0);
101    Buzzer_Start(2, 800, 1000, 500);
102    while (Buzzer_IsFinish() == FALSE);
103  }
```

Initialization Setting valve, If success is not performed

```
101    Buzzer_Start(2, 800, 1000, 500);
```

times, frequency, delay time in the beginning(ms), delay time in the end(ms)

function3

```
109  void Buzzer_PlayTable(void)
110  {
111    static u32 i = 1;
112
113    /* Bee 1 times, gBee_Scale[i] Hz, active 250 ms, inactive 250 ms
114    if (Buzzer_IsFinish() == TRUE && i <= 21)
115    {
116      Buzzer_Start(1, gBee_Scale[i], 250, 250);
117      i++;
118    }
119  }
```

declare variable

loop setting

do the frequency array

```
54   u16 gBee_Scale[] =
55   {
56       0,
57     262,  294,  330,  349,  392,  440,  494,
58     523,  587,  659,  698,  784,  880,  988,
59    1046, 1175, 1318, 1397, 1568, 1760, 1976
60   };
```

the frequency array

Initialization Setting

```
94  void Buzzer_Init(u32 uFrequency)

16      TM_TimeBaseInitTypeDef TimeBaseInit;
17
18      TimeBaseInit.Prescaler =  BEE_TM_PRE - 1;
19      TimeBaseInit.CounterReload = uReload;
20      TimeBaseInit.RepetitionCounter = 0;
21      TimeBaseInit.CounterMode = TM_CNT_MODE_UP;
22      TimeBaseInit.PSCReloadTime = TM_PSC_RLD_IMMEDIATE;
23      TM_TimeBaseInit(HTCFG_BUZZER_PORT, &TimeBaseInit);
24
25      /* Clear Update Event Interrupt flag since the "TM_TimeBaseInit()" writes the UEV1G bit
26      TM_ClearFlag(HTCFG_BUZZER_PORT, TM_FLAG_UEV);
27    }
```

Setting the prescaler value

Timer reload value

Repeat counter's value

Counter's mode

The pre-scaler is reloaded immediately after the trigger

Timer interrupt status

Timer Prescaler Resgister

Timer Prescaler Register – PSCR

This register specifies the timer prescaler value to generate the counter clock.

Offset: 0x084

Reset value: 0x0000_0000

	31	30	29	28	27	26	25	24
	Reserved							
Type/Reset								
	23	22	21	20	19	18	17	16
	Reserved							
Type/Reset								
	15	14	13	12	11	10	9	8
	PSCV							
Type/Reset	RW 0	RW 0	RW 0	RW 0	RW 0	RW 0	RW 0	RW 0
	7	6	5	4	3	2	1	0
	PSCV							
Type/Reset	RW 0	RW 0	RW 0	RW 0	RW 0	RW 0	RW 0	RW 0

Timer Counter Configuration Register

Register Descriptions

Timer Counter Configuration Register – CNTCFR

This register specifies the MCTM counter configuration.

Offset: 0x000

Reset value: 0x0000_0000

	31	30	29	28	27	26	25	24
	Reserved							DIR
Type/Reset								RW 0

	23	22	21	20	19	18	17	16
	Reserved							CMSEL
Type/Reset							RW 0	RW 0

	15	14	13	12	11	10	9	8
	Reserved							CKDIV
Type/Reset							RW 0	RW 0

	7	6	5	4	3	2	1	0
	Reserved						UGDIS	UEV1DIS
Type/Reset							RW 0	RW 0

Timer Counter Configuration Register

Register Descriptions

Timer Counter Configuration Register – CNTCFR

This register specifies the MCTM counter configuration.

Offset: 0x000

Reset value: 0x0000_0000

	31	30	29	28	27	26	25	24
	Reserved							DIR
Type/Reset								RW 0

	23	22	21	20	19	18	17	16
	Reserved							CMSEL
Type/Reset							RW 0	RW 0

	15	14	13	12	11	10	9	8
	Reserved							CKDIV
Type/Reset							RW 0	RW 0

	7	6	5	4	3	2	1	0
	Reserved						UGDIS	UEV1DIS
Type/Reset							RW 0	RW 0

Timer Repetition Register

Timer Repetition Register – REPR

This register specifies the timer repetition counter value.

Offset: 0x08C

Reset value: 0x0000_0000

	31	30	29	28	27	26	25	24
	Reserved							
Type/Reset								
	23	22	21	20	19	18	17	16
	Reserved							
Type/Reset								
	15	14	13	12	11	10	9	8
	Reserved							
Type/Reset								
	7	6	5	4	3	2	1	0
	REPV							
Type/Reset	RW 0	RW 0	RW 0	RW 0	RW 0	RW 0	RW 0	RW 0

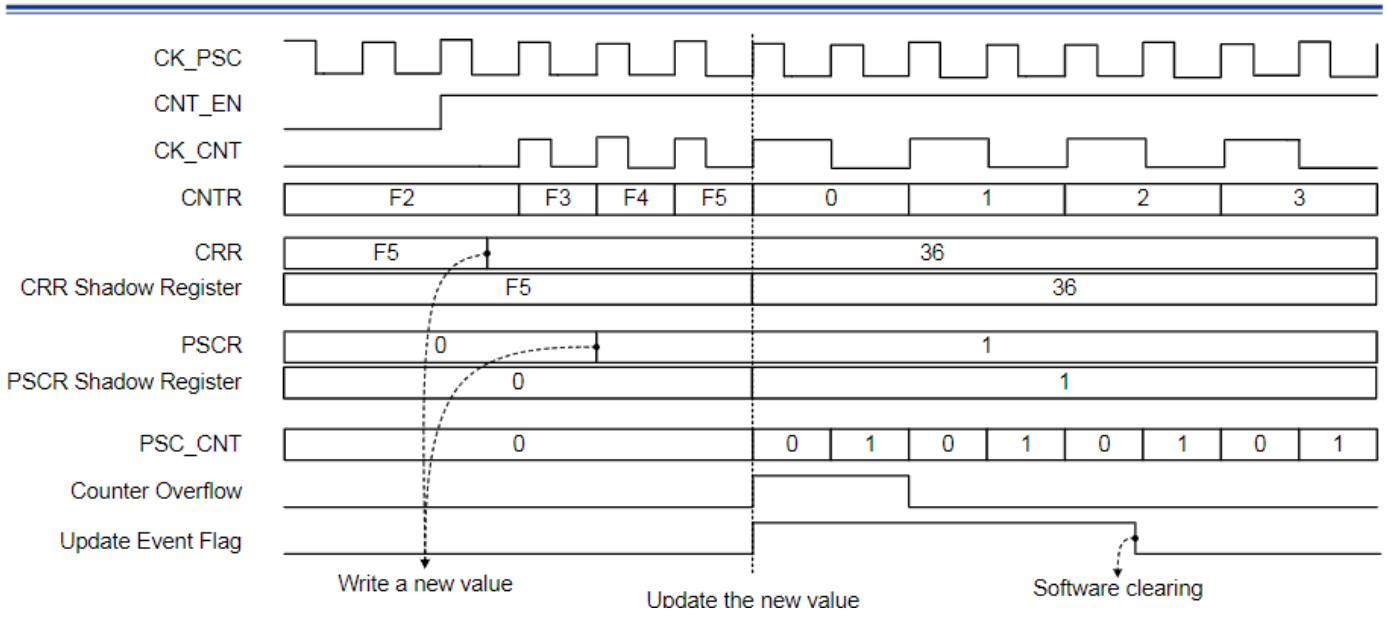
Counter mode

Functional Descriptions

Counter Mode

Up-Counting

The counter counts continuously from 0 to the counter-reload value, which is defined in the CRR register. Once the counter reaches the counter-reload value, the Timer Module generates an overflow event and the counter restarts to count once again from 0. This action will continue repeatedly. When the update event is generated by setting the UEVG bit in the EVGR register to 1, the counter value will also be initialized to 0.



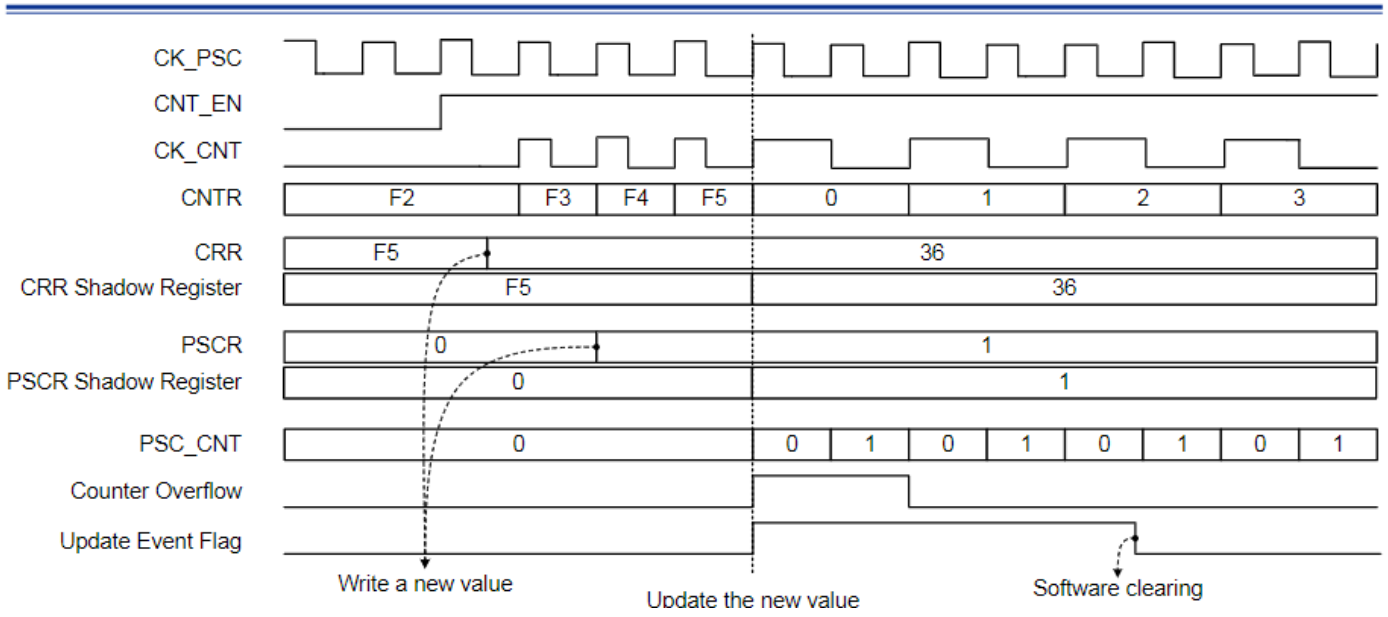
Counter mode

Functional Descriptions

Counter Mode

Up-Counting

The counter counts continuously from 0 to the counter-reload value, which is defined in the CRR register. Once the counter reaches the counter-reload value, the Timer Module generates an overflow event and the counter restarts to count once again from 0. This action will continue repeatedly. When the update event is generated by setting the UEVG bit in the EVGR register to 1, the counter value will also be initialized to 0.



Buzzer_Start

```
167  void Buzzer_Start(u8 uBeeTimes, u16 uFrequency, u16 uActive_ms, u16 uInActive_ms)
168  {
169      u32 uReload  =  (BEE_SYS_CLK / BEE_TM_PRE / uFrequency) - 1;
170      u32 uCompare =  ((uReload + 1) * (100 - BEE_DUTY)) / 100;
171
172      gBee_Time = uBeeTimes;
173      gBee_Active = BEE_TIME_MS(uActive_ms, uFrequency);
174      gBee_InActive = BEE_TIME_MS(uInActive_ms, uFrequency);
175      gBee_Count = gBee_Active + gBee_InActive;
176
177      TM_SetCounterReload(HTCFG_BUZZER_PORT, uReload);
178      TM_SetCaptureCompare(HTCFG_BUZZER_PORT, HTCFG_BUZZER_CH, uCompare);
179
180      TM_IntConfig(HTCFG_BUZZER_PORT, TM_INT_UEV, ENABLE);
181      TM_ChannelConfig(HTCFG_BUZZER_PORT, HTCFG_BUZZER_CH, TM_CHCTL_ENABLE);
182      TM_Cmd(HTCFG_BUZZER_PORT, ENABLE);
183  }
```

Parameter Calculation

Set Counter Reload

Set Capture Compare

Interrupt Enable

Channel Enable

MCTM Enable

Add USART to check value

Add the function

```
67  int main(void)
68  {
69    RETARGET_Configuration();
70    Buzzer_Fun1();
71
72    Buzzer_Fun2();
73
74    Buzzer_Init(0);
75
76    while (1)
77    {
78      Buzzer_PlayTable();
79    }
80  }
```

F12

Serial Port Setting

```
118 void RETARGET_Configuration(void)
119 {
120 #ifdef RETARGET_IS_UART
121   /* !!! NOTICE !!!
122      Notice that the local variable (structure) did not have a
123      Please confirm that there are no missing members in the p
124   */
125   USART_InitTypeDef USART_InitStructure;
126   #ifdef RETARGET_UxART_BAUDRATE
127   USART_InitStructure.USART_BaudRate = 115200;
128   #else
129   USART_InitStructure.USART_BaudRate = 115200;
130   #endif
131   USART_InitStructure.USART_WordLength = USART_WORDLENGTH_8B;
132   USART_InitStructure.USART_StopBits = USART_STOPBITS_1;
133   USART_InitStructure.USART_Parity = USART_PARITY_NO;
134   USART_InitStructure.USART_Mode = USART_MODE_NORMAL;
```

Edit Buzzer_Init(0)

```
67  int main(void)
68  {
69    RETARGET_Configuration();
70    Buzzer_Fun1();
71
72    Buzzer_Fun2();
73
74    Buzzer_Init(0);
75
76    while (1)
77    {
78      Buzzer_PlayTable();
79    }
```

F12

```
116   TM_TimeBaseInitTypeDef TimeBaseInit;
117
118   TimeBaseInit.Prescaler =  BEE_TM_PRE - 1;
119   TimeBaseInit.CounterReload = uReload;
120   TimeBaseInit.RepetitionCounter = 0;
121   TimeBaseInit.CounterMode = TM_CNT_MODE_UP;
122   TimeBaseInit.PSCReloadTime = TM_PSC_RLD_IMMEDIATE;
123   TM_TimeBaseInit(HTCFG_BUZZER_PORT, &TimeBaseInit);
124
125   /* Clear Update Event Interrupt flag since the "TM_TimeBaseInit()" writes the UEV1G bit */
126   TM_ClearFlag(HTCFG_BUZZER_PORT, TM_FLAG_UEV);
127   printf("\r Prescaler = %4d , CounterReload = %8d\n", TimeBaseInit.Prescaler,TimeBaseInit.CounterReload);
128 }
```

Outputs the value in the register to the Tera Term display.

Scale Frequency Table

高音	Do	Do#	Re	Re#	Mi	Fa	Fa#	So	So#	La	La#	Si
頻率	1048	1108	1176	1244	1320	1396	1480	1568	1660	1760	1856	1976
中音	Do	Do#	Re	Re#	Mi	Fa	Fa#	So	So#	La	La#	Si
頻率	524	554	588	622	660	698	740	784	830	880	928	988
低音	Do	Do#	Re	Re#	Mi	Fa	Fa#	So	So#	La	La#	Si
頻率	262	277	294	311	330	349	370	392	415	440	464	494

Check the pin

```
67  int main(void)
68  {
69    RETARGET_Configuration();
70    Buzzer_Fun1();
71
72    Buzzer_Fun2();
73
74    Buzzer_Init(0);
75
76    while (1)
77    {
78      Buzzer_PlayTable();
79    }
```

F12

```
114 #if defined(USE_HT32F52352_SK)
115   #define  _HTCFG_BUZZER_GPIOX                    A
116   #define  _HTCFG_BUZZER_GPION                    10
117   #define  HTCFG_BUZZER_IPN                       MCTM0
118   #define  HTCFG_BUZZER_CHN                       1
119 #endif
```

```
 94  void Buzzer_Init(u32 uFrequency)
 95  {
 96    u32 uReload;
 97
 98    { /* Enable peripheral clock
 99      CKCU_PeripClockConfig_TypeDef CKCUClock = {{ 0 }};
100      CKCUClock.Bit.AFIO = 1;
101      CKCUClock.Bit.HTCFG_BUZZER_IPN = 1;
102      CKCU_PeripClockConfig(CKCUClock, ENABLE);
103    }
```

F12

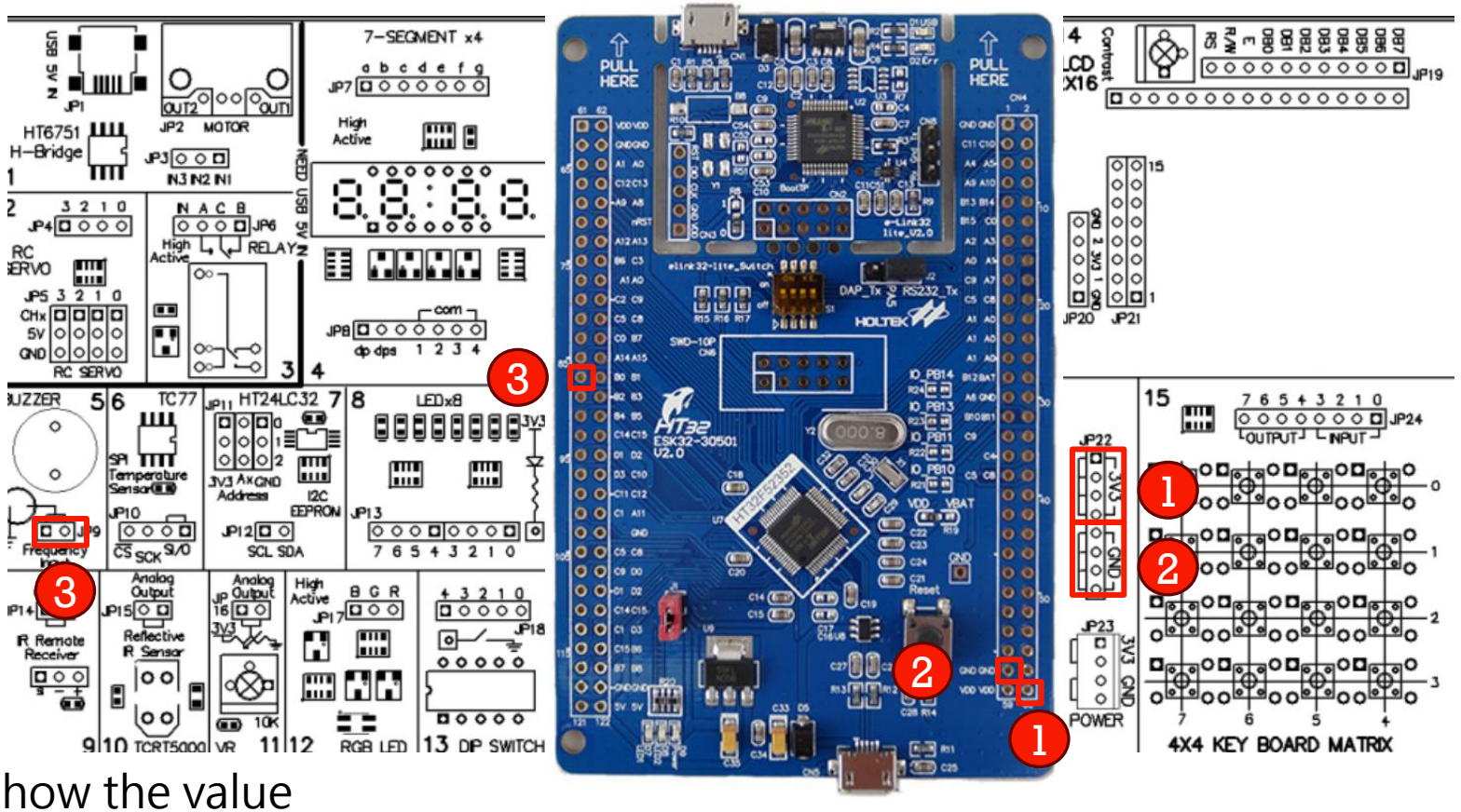
Homework W4-1.

https://github.com/CYCU-AIoT-System-Lab/Microcontroller-Experiment/blob/main/w4/TM-PWM_Buzzer-Experiment_Steps.md

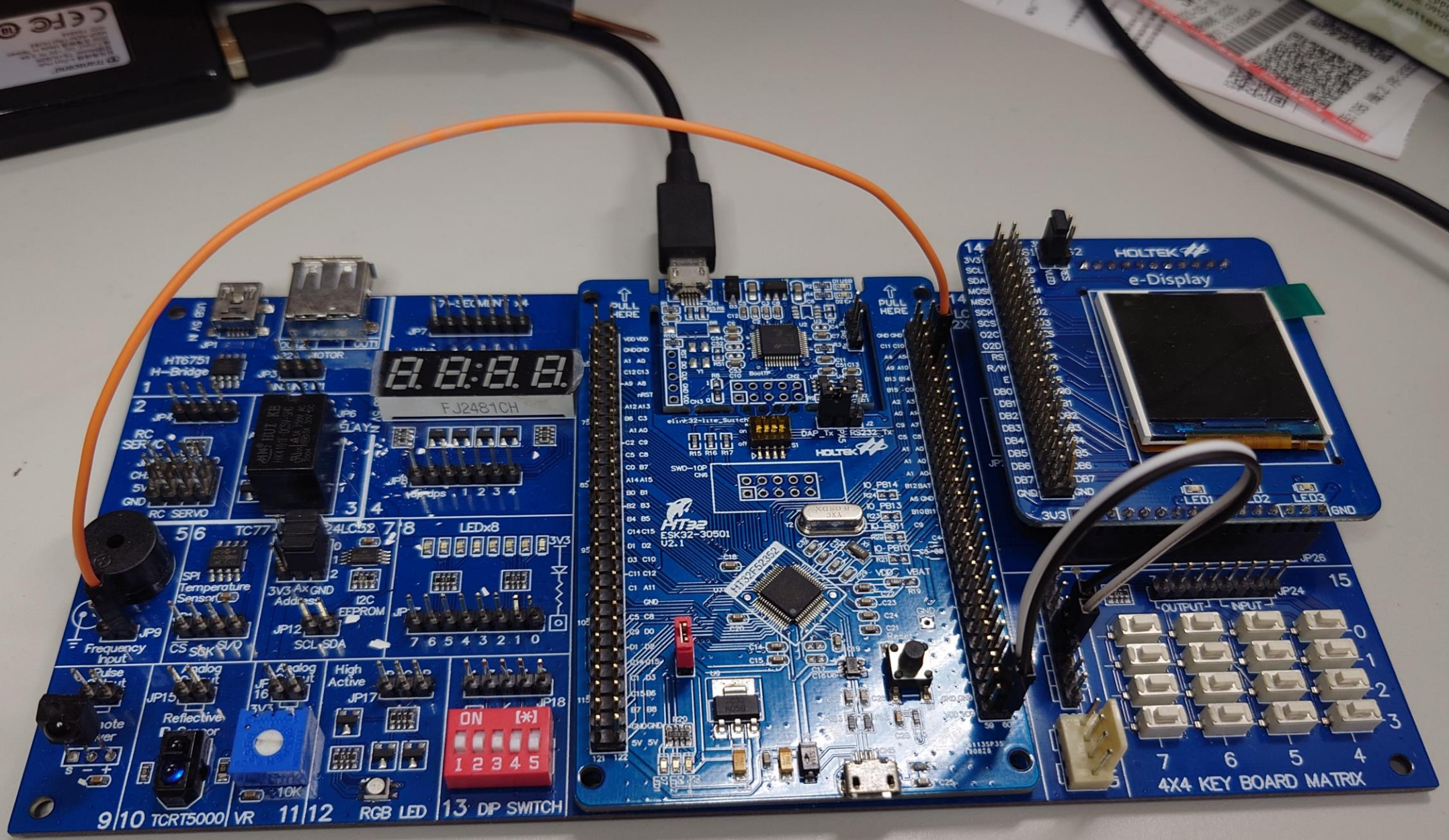
Execute the example

- Objective: Use timer to control buzzer and display in Tera Term.
- Hint:

1. Wiring



2. TERA TERM show the value



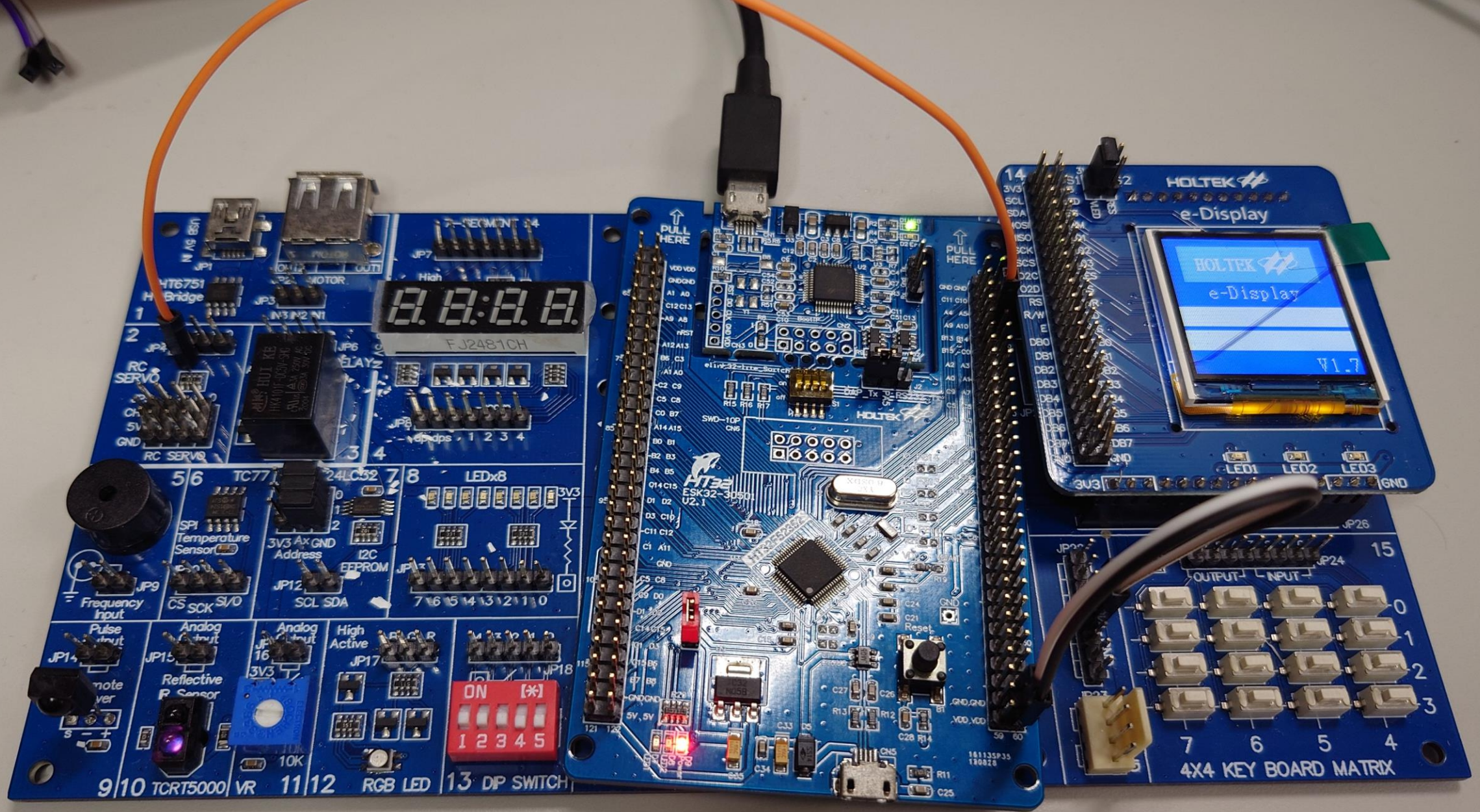
https://youtu.be/jCMfqXhKw2I

Homework W4-2

Play a part of For Alice music

- Objective: Edit the frequency array to play a part of “For Alice”.
- Hint:

1. Use pin **B0.**
2. Look for numbered musical notation of **“For Alice"**.
3. Find the corresponding scale frequency.
4. Edit gBee_Scale.



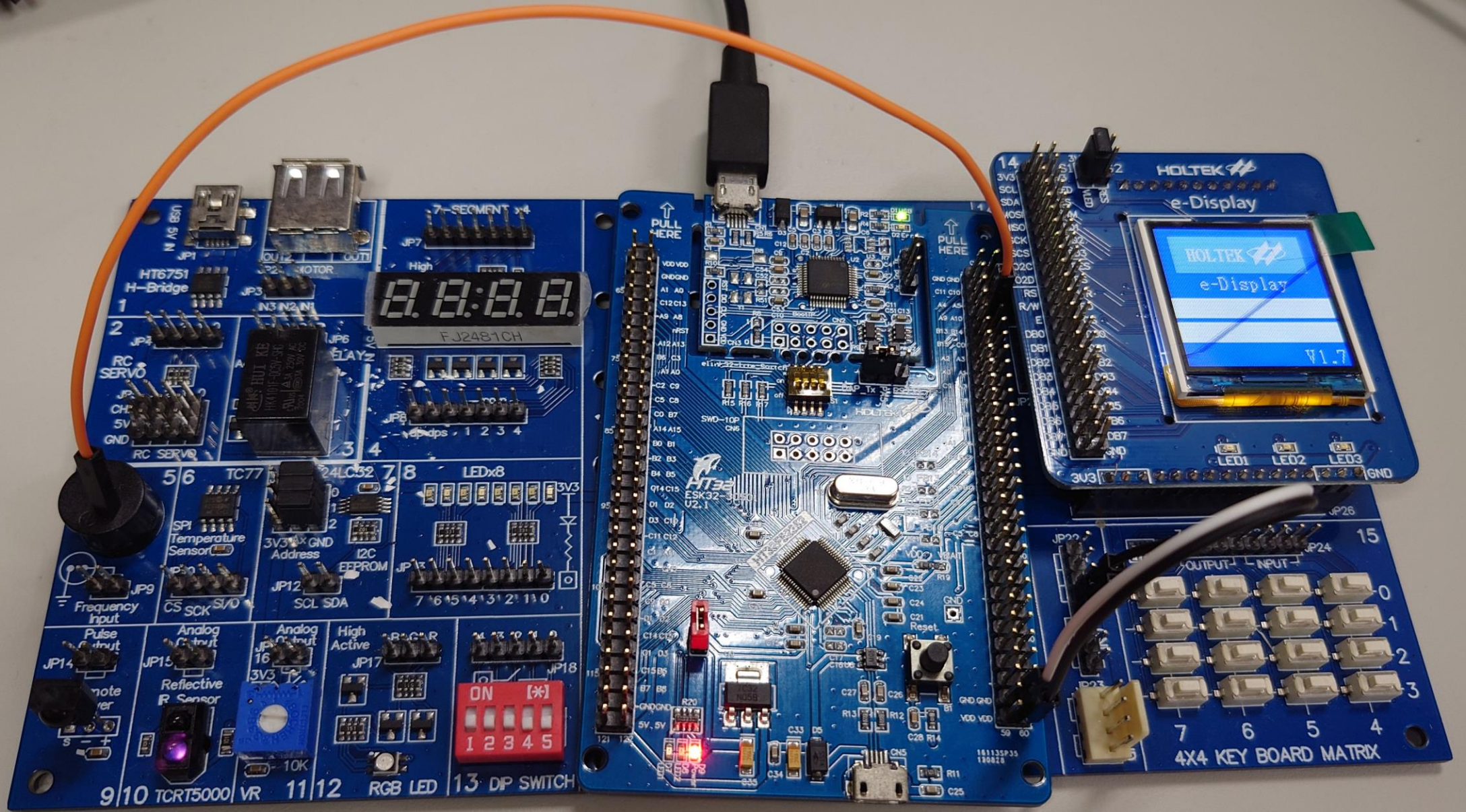
https://youtu.be/g1pHioxOzoQ

Homework W4-3

Play music faster.

- Objective: Edit delay time.
- Hint:

1. Use pin **B0.**
2. Edit Buzzer_PlayTable function



https://youtu.be/9lugvTlxR5s

Class Dismissed