實驗七 STM32 Clock and Timer

# 實驗目的

* 瞭解STM32的各種clock source使用與修改
* 瞭解STM32的timer使用原理
* 瞭解STM32的PWM使用原理與應用

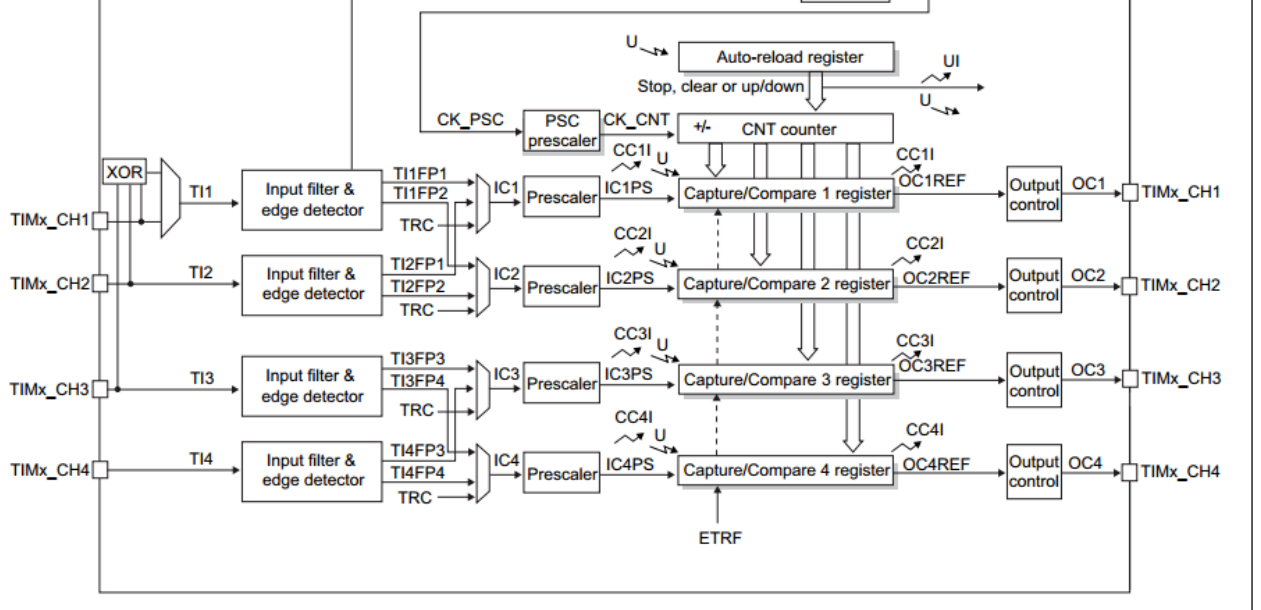
# 實驗原理

## Timer and Counter

請參考上課009-MCSL-CounterTimer講義。

## Timer PWM output mode

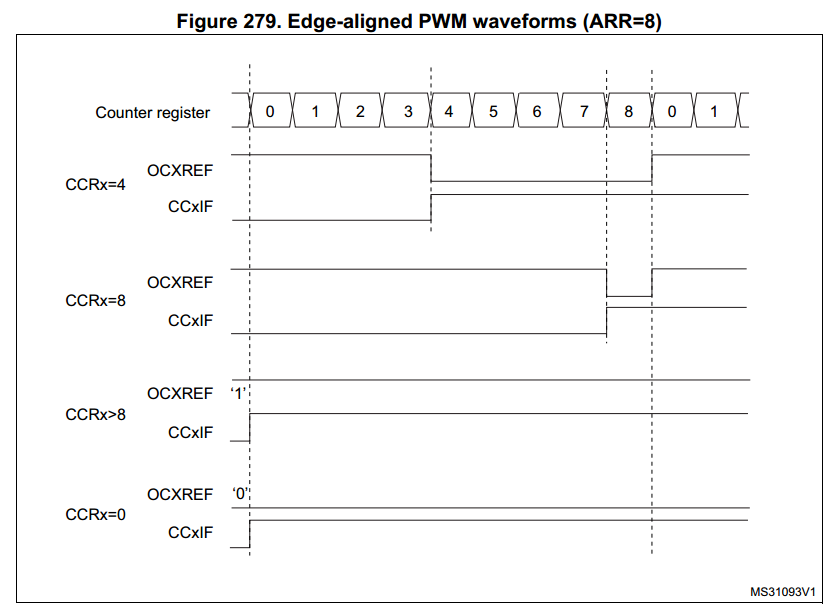
在STM32系統中要利用Timer產生PWM輸出，主要通過capture/compare mode register(TIMx\_CCMR1)與TIMx\_CCRx registers設定並利用TIMx\_CCER啟動之。



而一般PWM有分mode1與mode2兩種模式，而在計數器上數模式時其對應的輸出為

* PWM mode1: Channel is active as long as TIMx\_CNT<TIMx\_CCR1 else inactive.
* PWM mode2: Channel is inactive as long as TIMx\_CNT<TIMx\_CCR1 else active.

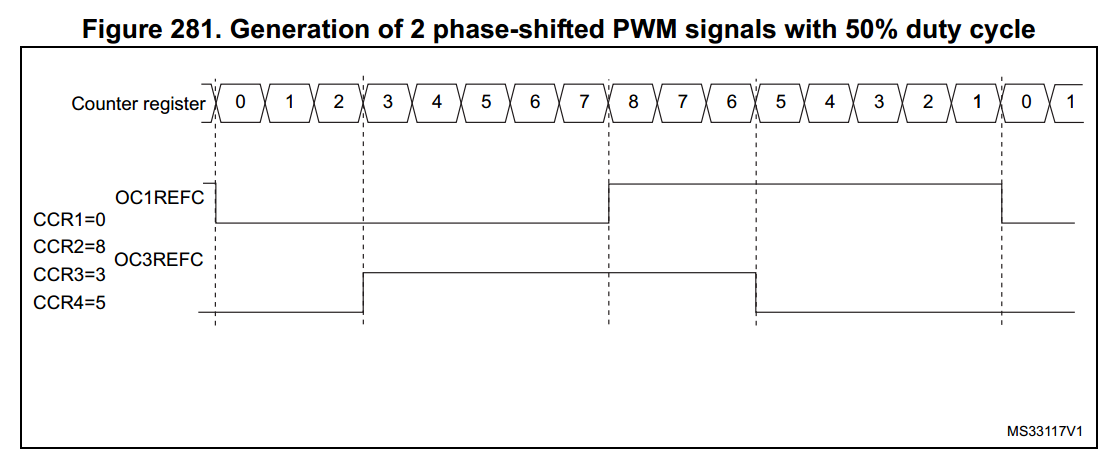
另外依不同的特殊用途又可分Combined PWM mode與Asymmetric PWM mode。



在ARR設定為8時，不同CCR值設定下OCxREF所對應的訊號輸出

以上圖範例來說當CCRx=4,ARR=8可以得到duty cycle(在單位時間內1準位與0準位的比例)為50%的波形，CCRx=8,ARR=8可得duty cycle= 1-1/8=87.5%的波形。

要輸出比較複雜的PWM與不同duty cycle波形也可以利用Asymmetric PWM mode來達成。



其他PWM設定細節用途請參閱Reference manual chapter 27.3.9 PWM mode

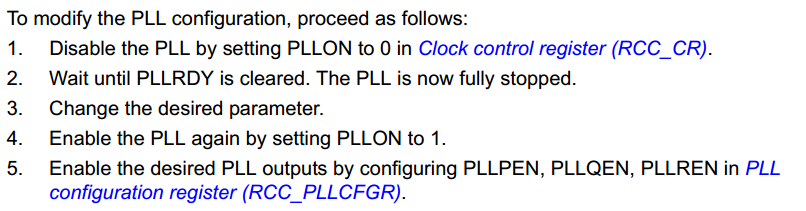
# 實驗步驟

## Modify system initial clock(20%)

* 請利用先前lab所實作的GPIO\_init與delay\_1s，可呼叫之前的assembly function或是用C重新實作，初始化GPIO與delay。
* 修改SYSCLK的clock source以及相關的prescaler使得CPU frequency(HCLK)為1MHz。
* 觀察修改前後LED燈閃爍的頻率。
* 當使用者按下user button便依以下順序改變CPU system clock(HCLK)， 1MHz -> 6MHz -> 10MHz ->16MHz -> 40MHz ->1MHz ->…

|  |
| --- |
| main.c |
| void GPIO\_init();  void 4MHz\_delay\_1s();  void SystemClock\_Config(){  //TODO: Change the SYSCLK source and set the corresponding Prescaler value.  }  int main(){  SystemClock\_Config();  GPIO\_init();  while(1){  if (user\_press\_button())  {  //TODO: Update system clock rate  }  GPIOA->BSRR = (1<<5);  4MHz\_delay\_1s ();  GPIOA->BRR = (1<<5);  4MHz\_delay\_1s ();  }  } |

Note: 有些CPU頻率設定須由PLLCLK內的倍頻器與除頻器達成，此時須將SYSCLK source改成PLLCLK並依以下流程設定RCC\_PLLCFGR register設定。



其中PLL clock頻率計算為f(VCO clock) = f(PLL clock input) × (PLLN / PLLM)

最終可輸出給system clock頻率為f(PLL\_R) = f(VCO clock) / PLLR

## 計時器(30%)

完成以下的main.c中的Timer\_init()與Timer\_start(); 並使用STM32 timer實做一個計時器會從0上數(Upcounting) TIME\_SEC秒的時間。顯示到小數點以下第二位，結束時7-SEG LED停留在TIME\_SEC的數字。(建議使用擁用比較高counter resolution 的TIM2~TIM5 timer)，請使用polling的方式取得 timer CNT register值並換算成時間顯示到7-SEG LED上。

* 1. TIME\_SEC 10000.00 (超過範圍請直接顯示0.00)

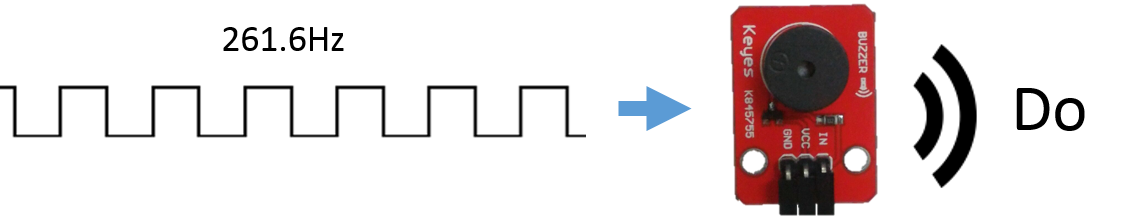
例如 TIME\_SEC 為12.7時的demo影片：<https://goo.gl/F9hh35>

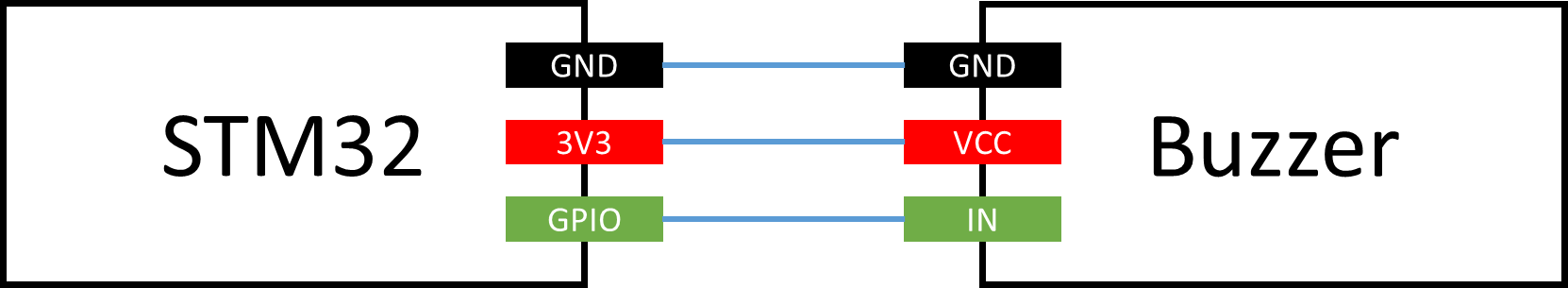
Note: 7-SEG LED驅動請利用之前Lab所實作的GPIO\_init()、max7219\_init()與Display ()函式呈現(須改成可呈現2個小數位)。

|  |
| --- |
| main.c |
| **#include** "stm32l476xx.h"  **#define** TIME\_SEC 12.70  **extern void** GPIO\_init();  **extern void** max7219\_init();  **extern void** Display();  **void** **Timer\_init**( TIM\_TypeDef \*timer)  {  //**TODO**: Initialize timer  }  **void** **Timer\_start**(TIM\_TypeDef \*timer){  //**TODO**: start timer and show the time on the 7-SEG LED.  }  **int** **main**()  {  GPIO\_init();  max7219\_init();  Timer\_init();  Timer\_start();  **while**(1)  {  //**TODO**: Polling the timer count and do lab requirements  }  } |

## Music keypad(35%)

蜂鳴器分為有源(自激式)蜂鳴器和無源(他激式)蜂鳴器。有源蜂鳴器將驅動電路直接設計到蜂鳴器中，因此只需提供直流電壓就可以發出聲音，但其缺點是聲音的頻率無法更改。無源蜂鳴器外部需提供震盪波形才會發出聲音，其聲音的頻率就是輸入波的頻率。我們這次LAB使用的是無源蜂鳴器。





蜂鳴器的VCC接3.3V、GND接GND、IN接GPIO腳位。

請利用timer產生並輸出Duty cycle為**50%的PWM訊號**，並以Lab6中的keypad為鍵盤，當使用者在按下不同keypad按鍵時產生特定頻率(參考下表)的PWM方波給蜂鳴器，沒按鍵或按到沒功能的鍵時請不要發出聲音。本次實驗會需要設定GPIOx\_AFRH、GPIOx\_AFRL、TIMx\_CCER、TIMx\_CCMR1、TIMx\_CCR1…等registers。

Note: 參考[RM0351 Reference manual](http://www.st.com/resource/en/reference_manual/dm00083560.pdf)瞭解這些register的功能完成此次實驗。並利用[STM32L476xx](http://www.st.com/resource/en/datasheet/stm32l476rg.pdf)找到timer channel所對應的腳位。

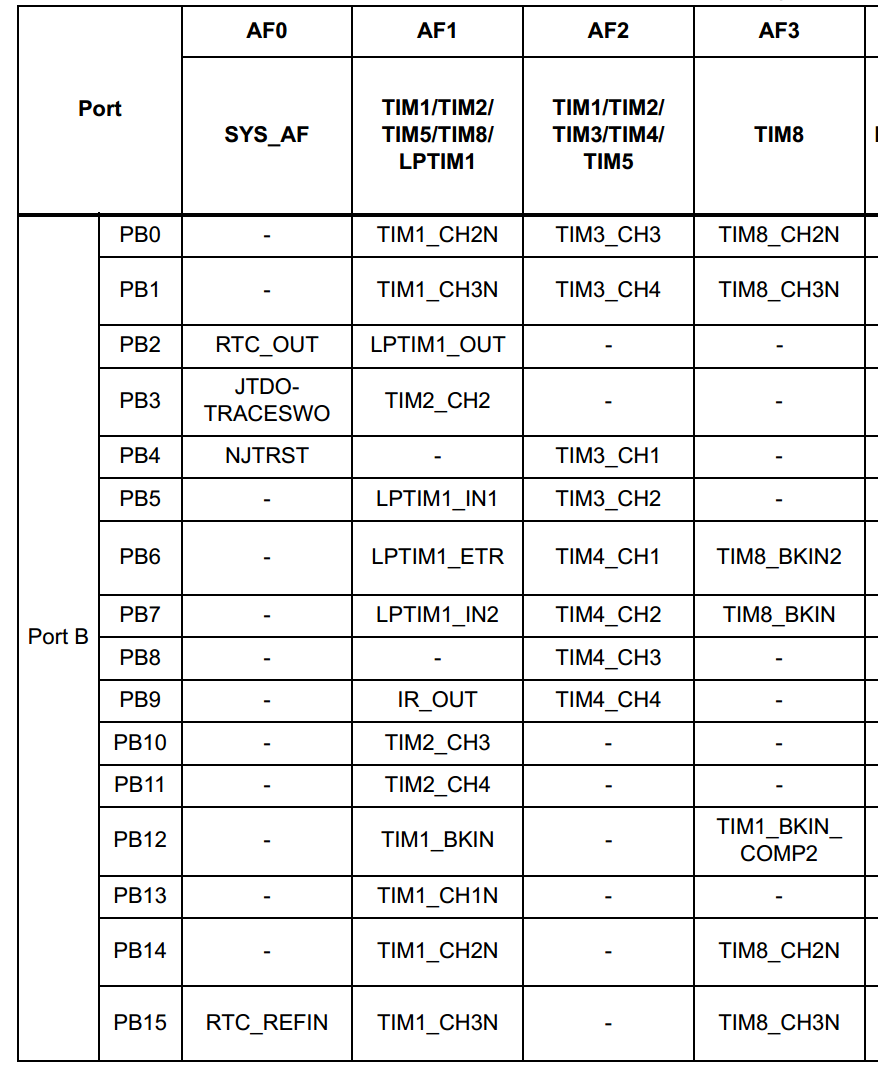
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | X0 | X1 | X2 | X3 |
| Y0 | Do | Re | Mi |  |
| Y1 | Fa | So | La |  |
| Y2 | Si | HDo |  |  |
| Y3 |  |  |  |  |

Keypad 對應音名

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 音名 | Do | Re | Mi | Fa | So | La | Si | HDo |
| 頻率(Hz) | 261.6 | 293.7 | 329.6 | 349.2 | 392.0 | 440.0 | 493.9 | 523.3 |

音名頻率對應表

Note: GPIO Pin設為PWM output時需設定為alternate function(AF) Mode，並根據根據所對應使用的timer設定AFRH與AFRL register，設定方式細節請參考reference manual與datasheet。



PortB AF mode selection table

範例：<https://goo.gl/4MuIFv>

|  |
| --- |
| extern void GPIO\_init();  void GPIO\_init\_AF(){  //TODO: Initial GPIO pin as alternate function for buzzer. You can choose to use C or assembly to finish this function.  }  void Timer\_init(){  //TODO: Initialize timer  }  void PWM\_channel\_init(){  //TODO: Initialize timer PWM channel  }  int main(){  GPIO\_init();  GPIO\_init\_AF();  Timer\_init();  PWM\_channel\_init();  //TODO: Scan the keypad and use PWM to send the corresponding frequency square wave to buzzer.  } |

### Music音色實驗(15%)

在前一實驗中的keypad增加2個功能按鈕用以調整PWM輸出的Duty cycle(範圍10%~90%，每按一次鍵調整5%)，觀察是否會影響蜂鳴器所發出的聲音大小或音色。

Note: 須注意頻率與duty cycle的關係來設定timer ARR與CCR registers。可用LED測試duty cycle是否有改變，成功應會看到LED隨著duty cycle不同而有明暗變化。