

**Bachelor of Science (Honours)**

**in**

**Microelectronics with Embedded Technology**

**(RMB2)**

**Title:**

General Purpose Timer (TIM2 to TIM5)

**Name ：**Ng Yen Aeng

**ID Number ：**15WAU09632

**Lecturer**: Dr. Poh Tze Ven

**Course**: BAME2123 Microcontroller Peripherals

Contents

[CHAPTER 1: INTRODUCTION 4](#_Toc439690373)

[1.1 Objective: 4](#_Toc439690374)

[1.2 Brief Background: 4](#_Toc439690375)

[CHAPTER 2: METHODOLOGY 5](#_Toc439690376)

[2.1 Basic counter mode: 7](#_Toc439690377)

[2.2 Output mode: 8](#_Toc439690378)

[2.3 Input Capture mode: 10](#_Toc439690379)

[CHAPTER 3: RESULT and DISCUSSION 11](#_Toc439690380)

[3.1 Basic counter mode: 11](#_Toc439690381)

[3.2 Output mode: 14](#_Toc439690382)

[3.3 Input Capture mode: 17](#_Toc439690383)

[CHAPTER 4: CONCLUSION 18](#_Toc439690384)

[REFERENCE 18](#_Toc439690385)

[APPENDIX 18](#_Toc439690386)

[Figure 1: Overview of the trigger selection and the master mode selection blocks. 4](#_Toc439690335)

[Figure 2: General-purpose timer block diagram 5](#_Toc439690336)

[Figure 3: Counter timing diagram with the prescaler division changed from 1 to 2. 7](#_Toc439690337)

[Figure 4: example of up-counting mode. 8](#_Toc439690338)

[Figure 5: example of down-counting mode. 8](#_Toc439690339)

[Figure 6: example of center-aligned mode. 8](#_Toc439690340)

[Figure 7: Output compare mode, toggle on OC1 9](#_Toc439690341)

[Figure 8： Edge-aligned PWM waveform example 10](#_Toc439690342)

[Figure 9: Counter = 0, prescaler = 2, preload = 0x200. 11](#_Toc439690343)

[Figure 10 11](#_Toc439690344)

[Figure 11 12](#_Toc439690345)

[Figure 12: configuration of countdown mode 12](#_Toc439690346)

[Figure 13: countdown mode 13](#_Toc439690347)

[Figure 14: configuration of center mode. 13](#_Toc439690348)

[Figure 15: configuration of forcing channel 1 to be high 14](#_Toc439690349)

[Figure 16: result of channel 1 when forced high 14](#_Toc439690350)

[Figure 17: configuration of forcing channel 1 to toggle. 15](#_Toc439690351)

[Figure 18: result of channel 1 when forced to toggle 15](#_Toc439690352)

[Figure 19: result of output compare mode 15](#_Toc439690353)

[Figure 20: PWM mode 1 16](#_Toc439690354)

[Figure 21: PWM mode 1 16](#_Toc439690355)

[Figure 22: PWM mode 2 17](#_Toc439690356)

[Figure 23: result of CCR2 17](#_Toc439690357)

[Figure 24: RCC.h 19](#_Toc439690358)

[Figure 25: RCC.c 19](#_Toc439690359)

[Figure 26: Gpio.c part1 20](#_Toc439690360)

[Figure 27: Gpio.c part 2 20](#_Toc439690361)

[Figure 28: Gpio.c part 3 20](#_Toc439690362)

[Figure 29: Gpio.c part 4 21](#_Toc439690363)

[Figure 30: Gpio.c part 5 21](#_Toc439690364)

[Figure 31: timer.h part 1 22](#_Toc439690365)

[Figure 32: timer.h part 2 22](#_Toc439690366)

[Figure 33: timer.c part 1 23](#_Toc439690367)

[Figure 34:timer.c part 2 23](#_Toc439690368)

[Figure 35: timer.c part 3 23](#_Toc439690369)

[Figure 36: main.c 24](#_Toc439690370)

[Figure 37: main.c 24](#_Toc439690371)

# CHAPTER 1: INTRODUCTION

## 1.1 Objective:

- To study the theory of general purpose timer (TIM2).

- To configure the TIM2 and investigate the result of different configuration on Timer 2 (TIM2).

## 1.2 Brief Background:

The general-purpose timers consist of a 32-bit auto-reload counter driven by a programmable prescaler. They may be used for a variety of purposes, including measuring the pulse lengths of input signals (input capture) or generating output waveforms (output compare and PWM).

Pulse lengths and waveform periods can be modulated from a few microseconds to several milliseconds using the timer prescaler and the RCC clock controller prescalers. The timers are completely independent, and do not share any resources. They can be synchronized together. The TIMx timers are linked together internally for timer synchronization or chaining. When one Timer is configured in Master Mode, it can reset, start, stop or clock the counter of another Timer configured in Slave Mode. Figure 1 presents an overview of the trigger selection and the master mode selection blocks.

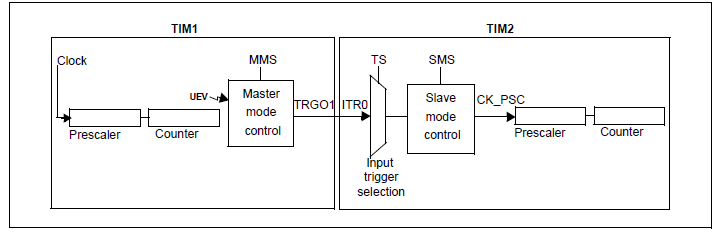


Figure 1: Overview of the trigger selection and the master mode selection blocks.

The main features of general-purpose TIM2 timer included:

1. 32-bits up, down, up/down auto-reload counter.
2. 16-bits programmable prescaler used to divide the counter clock frequency by any factor between 1 and 65536.
3. Up to 4 independent channels for (Input capture, output compare, PWM generation, one-pulse mode output).
4. Synchronization circuit to control the timer with external signals and to interconnect several timers.
5. Interrupt/DMA generation on the following event (update, trigger event, input capture, output compare).
6. Supports incremental (quadrature) encoder and hall-sensor circuitry for positioning purposes.
7. Trigger input for external clock or cycle-by-cycle current management.

The general-purpose timer block diagram was shown in figure 2.

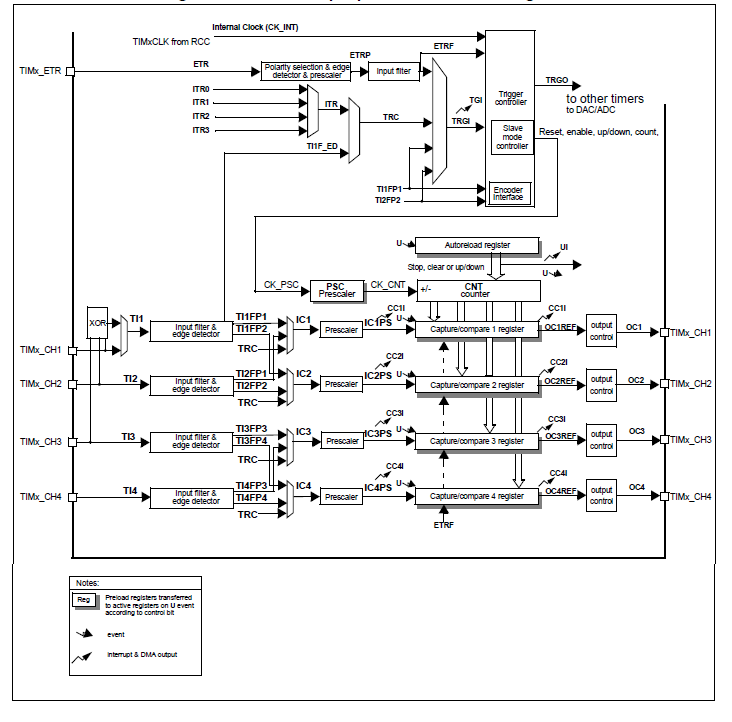


Figure 2: General-purpose timer block diagram

# CHAPTER 2: METHODOLOGY

Three features were configured and investigate in this experiment, which included basic timer mode, output compare mode and input capture mode. Basic timer mode can used to investigate the behavior of counter with different configurations. Output compare mode used to compare the counter value to the register value then create a trigger signal to form a different clock cycle, the result had been capture by using oscilloscope. In input capture mode, the register are used to latch the value of the counter after a transition detected by the corresponding input signal.

## 2.1 Basic counter mode:

The main block of the TIM2 is a 32-bit counter with its related auto-reload register, when the counter count up until it reached the value that loaded into auto-reload register, the counter will restart and count up again (if the timer is not in one-pulse mode). The counter clock can be divided by a prescaler to change the period of the clock. The counter, auto-reload register and the prescaler register can be read or write by the software even when the counter is running. Thus the three time base unit that needed to program included counter register (TIM2\_CNT), prescaler register (TIM2\_PSC) and auto-reload register (TIM2\_ARR).

Preload enable bit (ARPE) in TIM2\_CR1 register control whether the content of auto-reload register transferred into the shadow register permanently or at each update event (UEV). UEV is sent when the counter reached the overflow in upcounting mode or underflow in downcounting mode. Besides that, the counter is clocked by the prescaler output and the prescaler can divide the counter clock frequency by any factor between 1 and 65536.

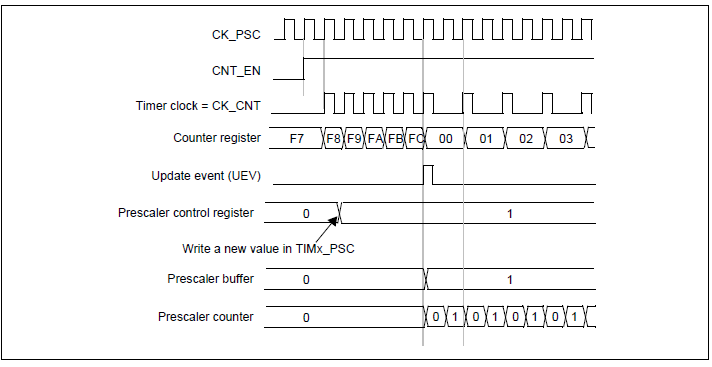


Figure 3: Counter timing diagram with the prescaler division changed from 1 to 2.

To investigate the behaviour of the counter, the following procedure was done:

1. RCC was configure and the clock of TIM2 was enabled.
2. GPIO pin was configure as the output of counter.
3. TIM2\_CNT, TIM2\_PSC and TIM2\_ARR was configured with different data.
4. The result was captured by using oscilloscope.

Besides the time base unit, there are three different mode in counter, which is up-counting, down-counting and centre-aligned mode. Up-counting mode allow counter to count up until value in ARR then restart the counter from 0. Down-counting mode start the counter from the value in ARR and count down to 0 then restart the counter from the value in ARR. Centre aligned mode slightly more complicated, it start counting up from 0 until value of ARR then count down until 0 then start count up again and repeat the counting.

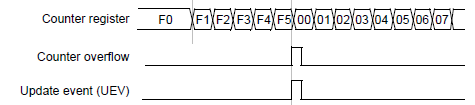


Figure 4: example of up-counting mode.

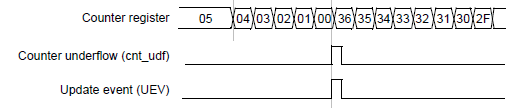


Figure 5: example of down-counting mode.

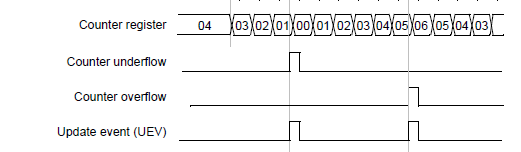


Figure 6: example of center-aligned mode.

## 2.2 Output mode:

There are three different mode that can be configured in output mode, which included forced output mode, output compare mode and pulse width modulation (PWM) mode.

***Forced output mode:***

In output mode, each output compare signal can be forced to active or inactive level directly by software without the comparison between the output compare register and the counter, this is called forced output mode. To force an output compare signal to its active level, the OC1M bits in TIM2\_CCMR1 register was program as 101 thus oc1ref is forced high and OC1 get opposite value to CC1P polarity. Besides that, oc1ref signal can be forced low by writing the OC1M bits to 100 in the TIM1\_CCMR1 register.

However, the comparison between the TIM1\_CCR1 shadow register and the counter is still performed and allows the flag to be set. Thus the interrupt and DMA requests can be sent accordingly.

***Output Compare mode:***

This mode is used to control an output waveform or indicating when a period of time has elapsed. The counter will keep compare to the value of the shadow register, when a match is found, the output compare function will assign the corresponding output pin to a programmable value defined by the output compare mode and the output polarity. Then a flag (CC1IF) in interrupt status register will be set and an interrupt will be generated if the corresponding interrupt mask is set.

In output compare mode, the update event UEV that will be generated when timer overflow has no effect on oc1ref and OC1 output. It is because oc1ref and OC1 output were only affected by comparison between shadow register and the counter. To investigate this mode, the following procedure was done:

1. Counter clock was selected as internal clock.
2. The desired date was written in the TIM2\_ARR and TIM2\_CCR1 registers.
3. TIM2\_CCMR1 register was configured to set channel 1 as output mode.
4. The counter was enabled by setting CEN bit in TIM2\_CR1 register.

TIM2\_CCR1 register can be updated at any time to control the output waveform, the example is shown in figure 7.

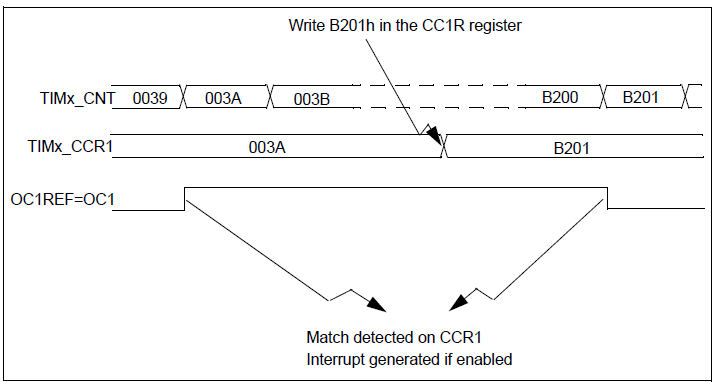


Figure 7: Output compare mode, toggle on OC1

***PWM output mode:***

Pulse width modulation mode allow the generation of a signal with a frequency determined by value of TIM2\_ARR register and a duty cycle determined by the value of the TIM2\_CCR1 register. The PWM mode can be selected independently on each channel by configuring the OCxM bits in the TIM2\_CCMRx register. For this mode, corresponding preload register and auto-reload preload register had to be enabled by setting the OCxPE bit in TIM2\_CCMRx register and ARPE bit in TIM2\_CR1 register respectively.

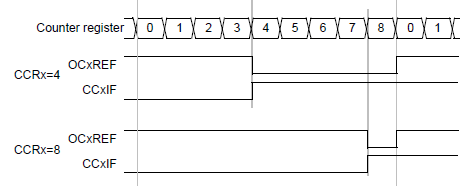


Figure 8： Edge-aligned PWM waveform example

## 2.3 Input Capture mode:

In input capture mode, the CCR registers are used to latch the value of the counter after a transition detected by the corresponding input signal. When a capture occurs, the corresponding CC1IF flag is set and DMA request or interrupt will be sent if they are enabled. While when the CC1IF flag already set and a capture occurred, the over-capture flag CC1OF is set. The CC1IF flag can be cleared by writing 0 to the bit or bt reading the captured data stored in CCR1 register.

The following procedure shows the steps of capturing the counter value in TIM2\_CCR2 when TI1 input rises.

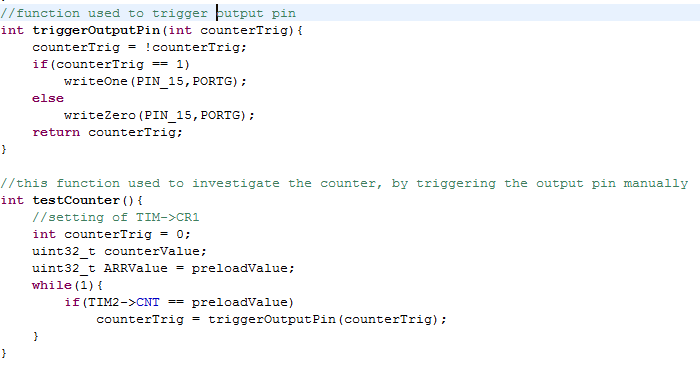
1. Active input had been selected as TI1 (write 01 to CC2S bits in TIM2\_CCMR1 register).
2. Input filter duration was programed with respect to the signal connected to the timer (program the IC2F bits in TIM2\_CCMR1 register).
3. Rising edge was selected on TI1 channel by writing the CC2P and CC2NP bits to 00 in TIM2\_CCER register.
4. Input prescaler was programmed as no prescaler.
5. Capture from the counter into the capture register was enabled by setting the CC2E bit in the TIM2\_CCER register.

Input filter duration is needed because during toggling the input signal is not stable around 5 internal clock cycles, thus a filter duration that is longer than these 5 clock cycles is needed to obtain a clear input signal.

# CHAPTER 3: RESULT and DISCUSSION

This session shows all the result of tests that stated in chapter 2. The tests included basic counter mode, output mode and input capture mode.

## 3.1 Basic counter mode:



The code above shows the program that used to trigger the output pin manually. Where port G pin 15 is the output pin. Figure 10 shown the wavelength of the output pin15 port G that toggled whenever the counter overflow. The initial counter value was set as 0, the prescaler is 2 and the preloaded value is 512 as shown in figure 9.



Figure 9: Counter = 0, prescaler = 2, preload = 0x200.



Figure 10

From the result that shown in oscilloscope, the period of the waveform in Figure 10 is 17.16µs. Then the preload value had changed to 15 and the respective result is shown in Figure 11. The expected period for the waveform in figure 11 is 503ns. While the period that shown in the oscilloscope is 498ns.

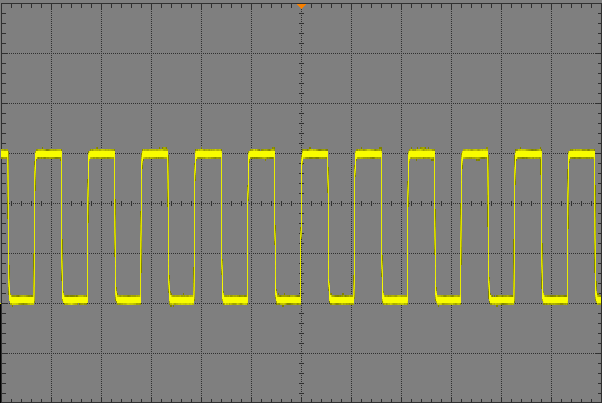


Figure 11

After that, the prescaler was set as 4, 6, 8, 10 and 12 then the result of period that obtained by using oscilloscope had been recorded and tabulated in table 1.

Table 1

|  |  |  |
| --- | --- | --- |
| Prescaler | Period (measured)/ µs | Period (calculated)/ µs |
| 2 | 0.498 | 0.503 |
| 4 | 0.998 | 1.006 |
| 6 | 1.487 | 1.509 |
| 8 | 1.999 | 2.012 |
| 10 | 2.478 | 2.515 |
| 12 | 3.001 | 3.018 |

As shown in table 1, when the prescaler increase the period of the waveform also increase. Which means that when the prescaler increase, the counter of the timer will count slowly as discussed in chapter 2.



Figure 12: configuration of countdown mode

The counter had changed to countdown mode and the period was measured and recorded in table 2.

Table 2

|  |  |  |
| --- | --- | --- |
| Prescaler | Period (measured)/ µs | Period (calculated)/ µs |
| 2 | 0.498 | 0.503 |
| 4 | 0.998 | 1.006 |
| 6 | 1.487 | 1.509 |
| 8 | 1.999 | 2.012 |
| 10 | 2.478 | 2.515 |
| 12 | 3.001 | 3.018 |

The period of the countdown mode is same as the period that measured in count up mode. The only different between countdown and count up mode is the phase different. The result of the waveform in countdown mode is shown in figure 13.



Figure 13: countdown mode



Figure 14: configuration of center mode.

Table 3

|  |  |
| --- | --- |
| Prescaler | Period (measured)/ µs |
| 2 | 0.988 |
| 4 | 1.999 |
| 6 | 3.002 |
| 8 | 4.034 |
| 10 | 4.987 |
| 12 | 6.003 |

The period of the waveform in center-aligned mode is twice larger than the result in count up mode and countdown mode. This is because when the counter count up to preloaded value, the counter start count down until it reach 0, therefore the time taken for the counter to reach preloaded value is around two times larger than the time taken in count up or countdown mode.

## 3.2 Output mode:

***Forced output mode:***

From the manual, the corresponding output pin for the channel 1 of timer 2 is port A pin 0, so port A pin 0 had been configure as alternative function to check the output of the channel 1. The result of forcing output is shown in figure 16 and figure 18.



Figure 15: configuration of forcing channel 1 to be high



Figure 16: result of channel 1 when forced high



Figure 17: configuration of forcing channel 1 to toggle.



Figure 18: result of channel 1 when forced to toggle

***Output compare mode:***



The value of 0x50 (80 in decimal) has been written to the CCR register by using the function that shown above. The figure 19 show the single capture of the waveform in channel 1 when the value of the counter is equal to the value in the CCR register.

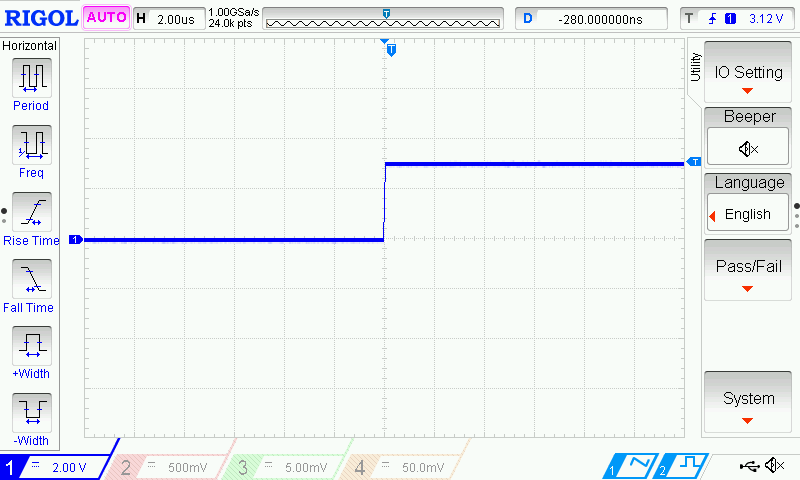


Figure 19: result of output compare mode

***Output PWM mode:***



As shown above, channel 1 of timer 2 had been configured as PWM mode 1. PWM mode 1 means that whenever the value in counter is smaller than the value in CCR1 register, the output of channel 1 is high else the output of channel 1 become low. The waveform that capture this scenario is shown in figure 20 and 21.



Figure 20: PWM mode 1



Figure 21: PWM mode 1

As shown in figure 20 and 21, the period of the cycle is 17.16 µs and the period of low is 14.4µs. The period of high = 17.16 – 14.4, which is equal to 2.76 µs. From the data that calculated and measured, the duty cycle can be calculated. Duty cycle = Thigh / Tcycle = 16% duty cycle.

Where the expected duty cycle is 0x50/0x200 = 15.625% which is closed to the measured duty cycle.



The CCMR1 register then configure to change channel 1 to PWM mode 2. PWM mode 2 shows that whenever the value in counter is smaller than the value in CCR1 register, the output of channel 1 is low else the output of channel 1 become high. The waveform that capture this scenario is shown in figure 22.



Figure 22: PWM mode 2

The Tcycle of this mode is same as the Tcycle of PWM mode 1, while the Thigh and Tlow of PWM mode 2 is equal to Tlow and Thigh of PWM mode 1 respectively.

So duty cyclemode2 = 14.4 / 17.16 = 84%.

Where the expected duty cycle = 84.375%. From the result shown in this session, the configuration of this session is correct as the result of measured and calculated met the expectation.

## 3.3 Input Capture mode:

In this session, channel 2 had been configure to capture the input while channel one had been configure as output compare mode to generate the input signal TI1 for channel 2 to capture the input.

The result of channel 1 that shown in figure 19, is now the input signal to control the channel two to capture the counter value into its CCR2 register.

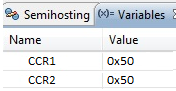
 The signal is low to high edge when the value of counter is equal to value in CCR1. At the same time the CCR2 register of channel 2 capture the value of counter, so the value of CCR2 is equal to value of CCR1.

Figure 23: result of CCR2

# CHAPTER 4: CONCLUSION

General purpose timer is a strong timer which can generate clock with different frequency with users’ desire. Besides that, it can also generate clock with different duty cycle which allow user to do a lot of different task. It also allows user to capture the input so that it can keep track of the time taken to finish a task.

# REFERENCE

[1] ST 2015, ‘General Purpose Timer’, *Reference Manual 0090*, vol. 2015

<<http://www.st.com/web/en/resource/technical/document/reference_manual/DM00031020.pdf/>>

[1] Alternate function mapping, timer

<http://www.sciencezero.org/index.php?title=STM32F429\_Microcontroller/>

# APPENDIX



Figure 24: RCC.h

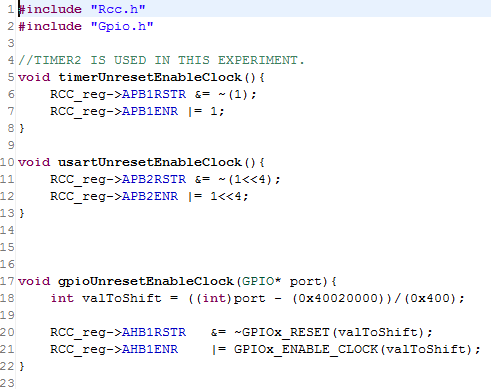


Figure 25: RCC.c

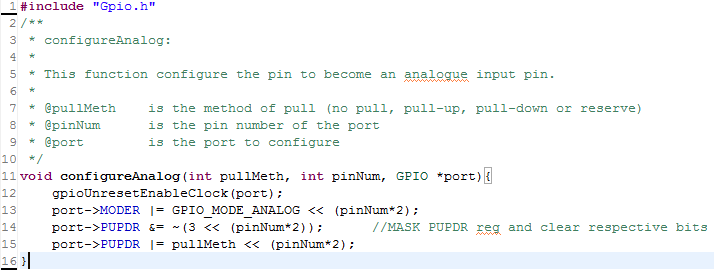


Figure 26: Gpio.c part1

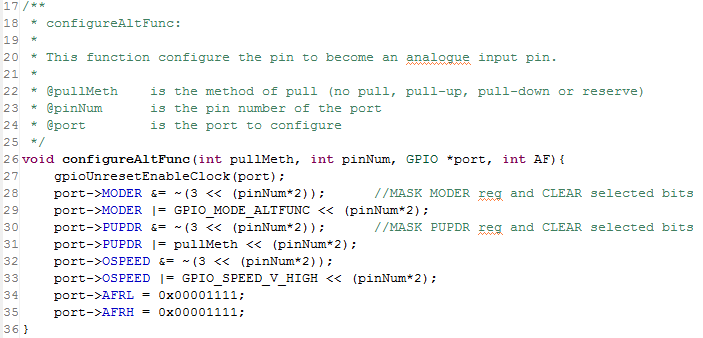


Figure 27: Gpio.c part 2

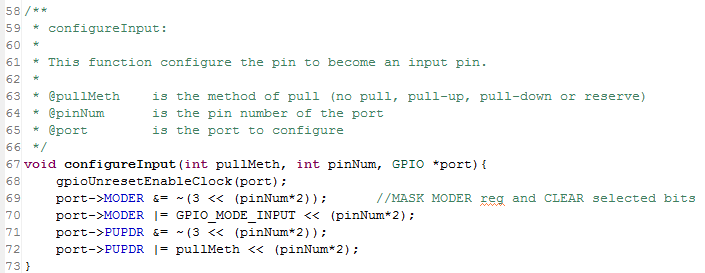


Figure 28: Gpio.c part 3

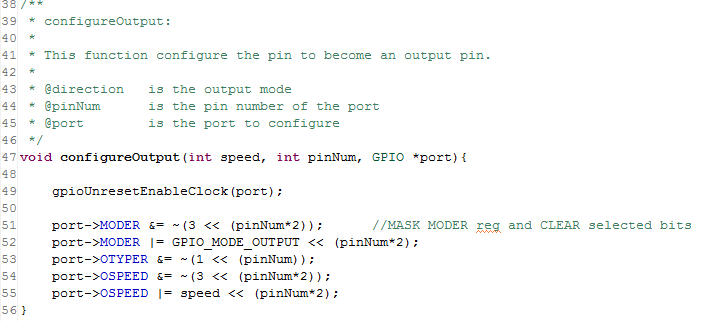


Figure 29: Gpio.c part 4

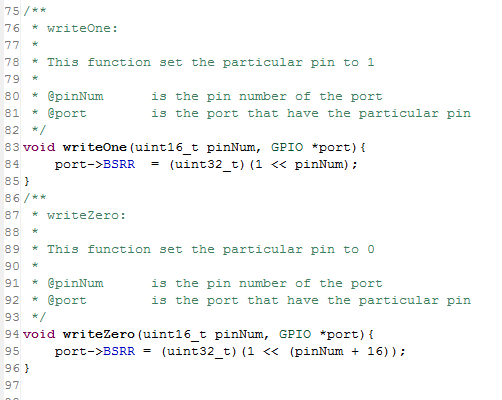


Figure 30: Gpio.c part 5

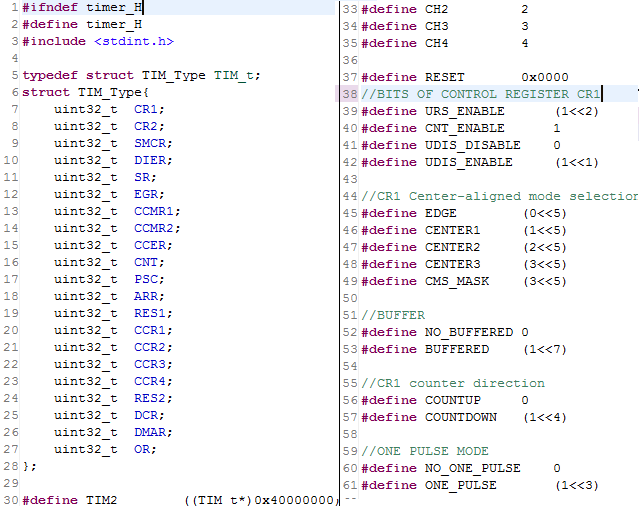


Figure 31: timer.h part 1

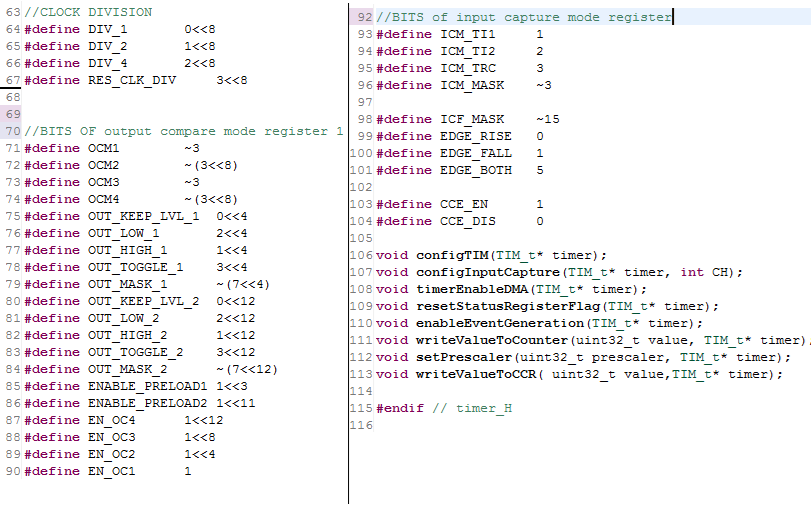


Figure 32: timer.h part 2

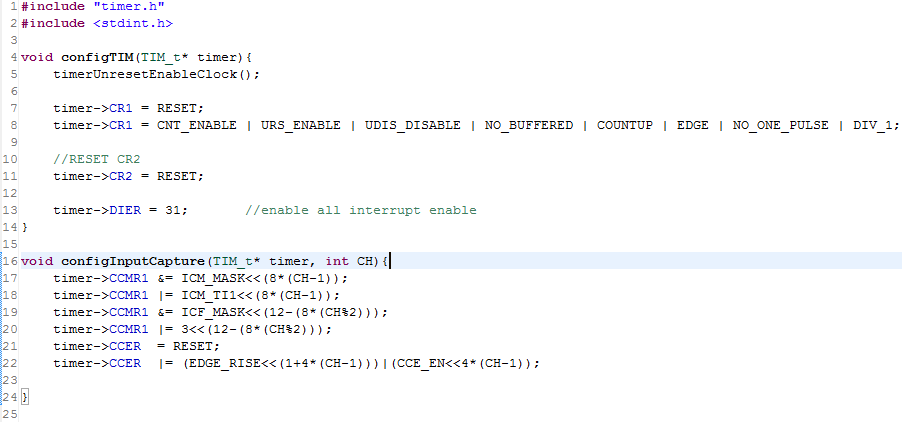


Figure 33: timer.c part 1

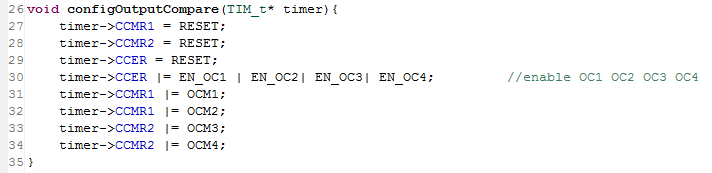


Figure 34:timer.c part 2

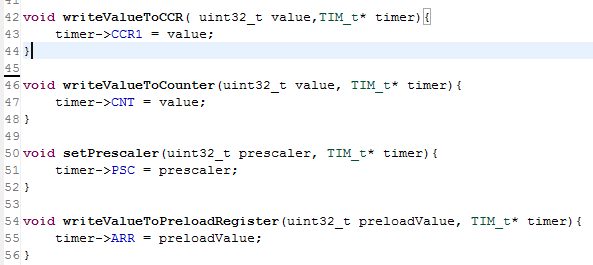


Figure 35: timer.c part 3

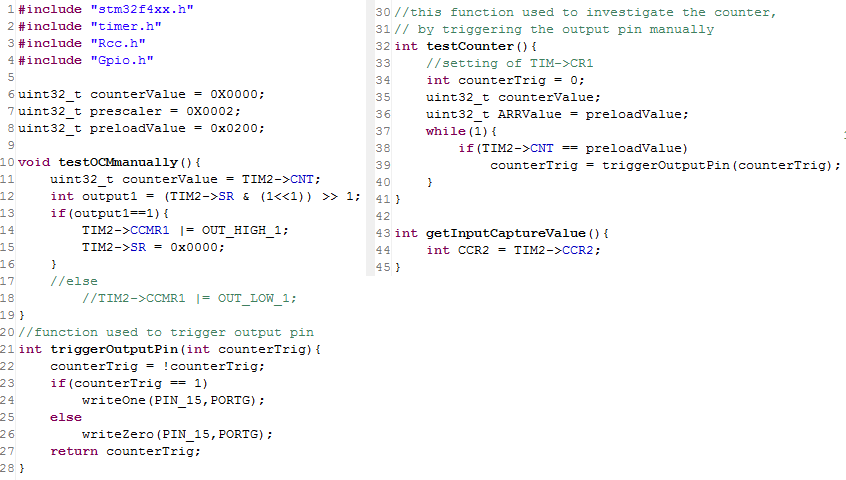


Figure 36: main.c

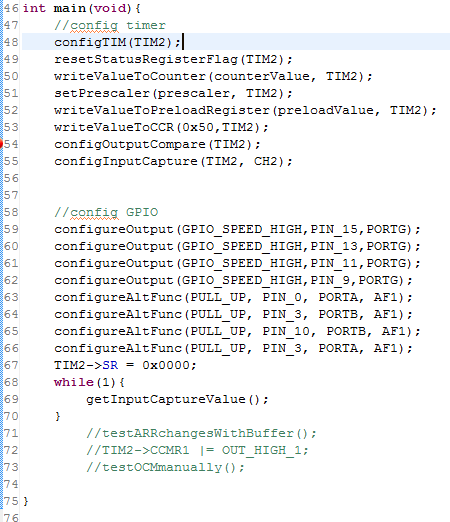


Figure 37: main.c