

微處理機 LAB 7 ADC

Due : 兩週後 早上 8:00

PART 1. (10%) 實作題

Lab 7.1 ADC:

請完成實驗 錄影或拍照紀錄實驗結果並附上程式碼(main.c 及 include 之.h, .c 檔案)

不使用 SysTick 中斷，單純執行一次 ADC 取樣光敏電阻值，並把取樣結果顯示在 7-segment 上。

```
1  #include "stm321476xx.h"
2  #include "helper_functions.h"
3  #include "usart.h"
4  #include "adc.h"
5  #include "led_button.h"
6  #include "7seg.h"
7  #include "stdio.h"
8
9  #define lab10_2
10 #define SEG_gpio GPIOC
11 #define DIN_pin 3
12 #define CS_pin 4
13 #define CLK_pin 5
14 double resistor;
15 int adcVal = 0;
16
17 void GPIO_init(){
18     // Setup FPU
19     SCB->CPACR |= (0xF << 20);
20     __DSB();
21     __ISB();
22
23     RCC->AHB2ENR |= RCC_AHB2ENR_GPIOAEN; // PA5 LED
24     RCC->AHB2ENR |= RCC_AHB2ENR_GPIOCEN; // PC13 Button
25     RCC->AHB2ENR |= RCC_AHB2ENR_ADCEN; // ADC
26
27     // Setup Button
28     GPIOC->MODER &= ~GPIO_MODER_MODE13_Msk;
29     GPIOC->MODER |= (0x3 << GPIO_MODER_MODE13_Pos);
30
31     // Setup PA5 LED
32     GPIOA->MODER &= ~GPIO_MODER_MODE5_Msk;
33     GPIOA->MODER |= (0x1 << GPIO_MODER_MODE5_Pos);
34
35     // PC0 ADC
36     GPIOC->MODER &= ~GPIO_MODER_MODE0_Msk;
37     GPIOC->MODER |= (0x3 << GPIO_MODER_MODE0_Pos);
38     GPIOC->ASCR |= GPIO_ASCR_ASC0;
39
40 }
41
42 void EXTI_Setup(){
43     // Enable SYSCFG CLK
44     RCC->APB2ENR |= RCC_APB2ENR_SYSCFGEN;
45     // Select output bits
46     SYSCFG->EXTICR[3] &= ~SYSCFG_EXTICR4_EXTI13_Msk;
47     SYSCFG->EXTICR[3] |= (2 << SYSCFG_EXTICR4_EXTI13_Pos);
```

```

46     SYSCFG->EXTICR[3] &= ~SYSCFG_EXTICR4_EXTI13_Msk;
47     SYSCFG->EXTICR[3] |= (2 << SYSCFG_EXTICR4_EXTI13_Pos);
48     // Enable interrupt
49     EXTI->IMR1 |= EXTI_IMR1_IM13;
50     // Enable Falling Edge
51     EXTI->RTSR1 |= EXTI_RTSR1_RT13;
52     // Enable NVIC
53     NVIC_EnableIRQ(EXTI15_10_IRQn);
54 }
55
56 void ADCInit(){
57     ADCResolution(ADC1, 0);           // 12 bits
58     ADCContinuousConversion(ADC1, 0); // enable continuous conversion
59     ADCDataAlign(ADC1, 0);            // set right align
60     ADCCommonDualMode(0);             // independent mode
61     ADCCommonClockMode(1);            // hclk / 1
62     ADCCommonPrescaler(0);            // div 1
63     ADCCommonDMAMode(0);              // disable dma
64     ADCCommonDelayTwoSampling(0b0100); // 5 adc clk cycle
65     ADCChannel(ADC1, 1, 1, 2);        // channel 1, rank 1, 12.5 adc clock cycle
66     ADCWakeup(ADC1);
67     ADCInterrupt(ADC1, ADC_IER_EOCIE, 1);
68     NVIC_EnableIRQ(ADC1_2_IRQn);
69     ADCEnable(ADC1);
70 }
71
72 void SysTick_Handler() {
73     if(SysTick->CTRL & SysTick_CTRL_COUNTFLAG_Msk){
74         //ADCStartConversion(ADC1);
75         light();
76         handler();
77         toggle_output(GPIOA,5);
78     }
79     return;
80 }
81
82 void ADC1_2_IRQHandler(){
83     if(ADC1->ISR & ADC_ISR_EOC){
84         ADC1->ISR &= ADC_ISR_EOC;
85         resistor = ADCGetValue(ADC1);
86         resistor = (3.3 / 4095) * resistor;
87         resistor = (3.3 - resistor) * 2000 / resistor;
88     }
89 }
90
91 void startADC() {
92     while (!(ADC1->ISR & ADC_ISR_ADRDY)) ADC1->CR |= ADC_CR_ADEN; // TURN ON
93     ADC1->ISR = ADC_ISR_EOC | ADC_ISR_EOS | ADC_ISR_OVR; // Clear flags
94     ADC1->CR |= ADC_CR_ADSTART; // START CONV
95 }
96
97 void light() {
98     startADC();
99     while (!(ADC1->ISR & ADC_ISR_EOC));
100     adcVal = ADC1->DR;
101 }
102
103 void handler(){
104     double Vsample = (3.3/4095)*adcVal;
105     double current = (3.3 - Vsample)/10000;
106     adcVal = Vsample/current;
107     send_7seg(SEG_gpio, DIN_pin, CS_pin, CLK_pin, SEG_ADDRESS_DIGIT_0, adcVal%10);
108     adcVal/=10;
109     send_7seg(SEG_gpio, DIN_pin, CS_pin, CLK_pin, SEG_ADDRESS_DIGIT_1, adcVal%10);
110     adcVal/=10;
111     send_7seg(SEG_gpio, DIN_pin, CS_pin, CLK_pin, SEG_ADDRESS_DIGIT_2, adcVal%10);
112     adcVal/=10;
113     send_7seg(SEG_gpio, DIN_pin, CS_pin, CLK_pin, SEG_ADDRESS_DIGIT_3, adcVal%10);
114     adcVal/=10;
115     send_7seg(SEG_gpio, DIN_pin, CS_pin, CLK_pin, SEG_ADDRESS_DIGIT_4, adcVal%10);
116     adcVal/=10;

```

```

119 void lab_10_2(){
120     EXTI_Setup();
121     // USARTInit();
122     ADCInit();
123     //SystemClock_Config_Interrupt(4, 1000000);
124     init_led(GPIOA,5);
125     light();
126     handler();
127     while(1){
128         //light();
129         //ADC1_2_IRQHandler();
130         //EXTI15_10_IRQHandler();
131         //TIM2_IRQHandler();
132         //display_two_decimal(SEG_gpio, DIN_pin, CS_pin, CLK_pin, adcVal);
133     }
134 }
135
136 int main(){
137     GPIO_init();
138     if(init_7seg(SEG_gpio, DIN_pin, CS_pin, CLK_pin) != 0){
139         return -1;
140     }
141     send_7seg(SEG_gpio, DIN_pin, CS_pin, CLK_pin, SEG_ADDRESS_DECODE_MODE, 0xFF);
142     send_7seg(SEG_gpio, DIN_pin, CS_pin, CLK_pin, SEG_ADDRESS_SCAN_LIMIT, 0x04);
143     send_7seg(SEG_gpio, DIN_pin, CS_pin, CLK_pin, SEG_ADDRESS_SHUTDOWN, 0x01);
144
145     lab_10_2();
146     while(1){}
147     return 0;
148 }

```

調慢 ADC 取樣間隔到每秒取樣做 ADC 一次，不可改變 sys_clk，然後把取樣結果顯示在 7-segment 上。

```

1  #include "stm321476xx.h"
2  #include "helper_functions.h"
3  #include "usart.h"
4  #include "adc.h"
5  #include "led_button.h"
6  #include "7seg.h"
7  #include "stdio.h"
8
9  #define lab10_2
10 #define SEG_gpio GPIOC
11 #define DIN_pin 3
12 #define CS_pin 4
13 #define CLK_pin 5
14 double resistor;
15 int adcVal = 0;
16
17 void GPIO_init(){
18     // Setup FPU
19     SCB->CPACR |= (0xF << 20);
20     __DSB();
21     __ISB();
22
23     RCC->AHB2ENR |= RCC_AHB2ENR_GPIOAEN; // PA5 LED
24     RCC->AHB2ENR |= RCC_AHB2ENR_GPIOCEN; // PC13 Button
25     RCC->AHB2ENR |= RCC_AHB2ENR_ADCEN; // ADC
26
27     // Setup Button

```

```

27 // Setup Button
28 GPIOC->MODER &= ~GPIO_MODER_MODE13_Msk;
29 GPIOC->MODER |= (0x0 << GPIO_MODER_MODE13_Pos);
30
31 // Setup PA5 LED
32 GPIOA->MODER &= ~GPIO_MODER_MODE5_Msk;
33 GPIOA->MODER |= (0x1 << GPIO_MODER_MODE5_Pos);
34
35 // PC0 ADC
36 GPIOC->MODER &= ~GPIO_MODER_MODE0_Msk;
37 GPIOC->MODER |= (0x3 << GPIO_MODER_MODE0_Pos);
38 GPIOC->ASCR |= GPIO_ASCR_ASC0;
39 }
40
41 void EXTI_Setup(){
42 // Enable SYSCFG CLK
43 RCC->APB2ENR |= RCC_APB2ENR_SYSCFGEN;
44 // Select output bits
45 SYSCFG->EXTICR[3] &= ~SYSCFG_EXTICR4_EXTI13_Msk;
46 SYSCFG->EXTICR[3] |= (2 << SYSCFG_EXTICR4_EXTI13_Pos);
47 // Enable interrupt
48 EXTI->IMR1 |= EXTI_IMR1_IM13;
49 // Enable Falling Edge
50 EXTI->RTSR1 |= EXTI_RTSR1_RT13;
51 // Enable NVIC
52 NVIC_EnableIRQ(EXTI15_10_IRQn);
53 }
54
55 void ADCInit(){
56 ADCResolution(ADC1, 0); // 12 bits
57 ADCContinuousConversion(ADC1, 0); // enable continuous conversion
58 ADCDataAlign(ADC1, 0); // set right align
59 ADCCommonDualMode(0); // independent mode
60 ADCCommonClockMode(1); // hclk / 1
61 ADCCommonPrescaler(0); // div 1
62 ADCCommonDMAMode(0); // disable dma
63 ADCCommonDelayTwoSampling(0b0100); // 5 adc clk cycle
64 ADCChannel(ADC1, 1, 1, 2); // channel 1, rank 1, 12.5 adc clock cycle
65 ADCWakeup(ADC1);
66 ADCInterrupt(ADC1, ADC_IER_EOCIE, 1);
67 NVIC_EnableIRQ(ADC1_2_IRQn);
68 ADCEnable(ADC1);
69 }
70
71 void SysTick_Handler() {
72 if(SysTick->CTRL & SysTick_CTRL_COUNTFLAG_Msk){
73 //ADCStartConversion(ADC1);
74 light();
75 handler();
76 toggle_output(GPIOA,5);
77 }
78 return;
79 }
80
81 void ADC1_2_IRQHandler(){
82 if(ADC1->ISR & ADC_ISR_EOC){
83 ADC1->ISR &= ADC_ISR_EOC;
84 resistor = ADCGetValue(ADC1);
85 resistor = (3.3 / 4095) * resistor;
86 resistor = (3.3 - resistor) * 2000 / resistor;
87 }
88 }
89
90 void startADC() {
91 while (!(ADC1->ISR & ADC_ISR_ADRDY)) ADC1->CR |= ADC_CR_ADEN; // TURN ON
92 ADC1->ISR = ADC_ISR_EOC | ADC_ISR_EOS | ADC_ISR_OVR; // Clear flags
93 ADC1->CR |= ADC_CR_ADSTART; // START CONV
94 }
95
96 void light() {
97 startADC();

```

```

96 void light() {
97     startADC();
98     while (!(ADC1->ISR & ADC_ISR_EOC));
99     adcVal = ADC1->DR;
100 }
101 void handler(){
102     double Vsample = (3.3/4095)*adcVal;
103     double current = (3.3 - Vsample)/10000;
104     adcVal = Vsample/current;
105     send_7seg(SEG_gpio, DIN_pin, CS_pin, CLK_pin, SEG_ADDRESS_DIGIT_0, adcVal%10);
106     adcVal/=10;
107     send_7seg(SEG_gpio, DIN_pin, CS_pin, CLK_pin, SEG_ADDRESS_DIGIT_1, adcVal%10);
108     adcVal/=10;
109     send_7seg(SEG_gpio, DIN_pin, CS_pin, CLK_pin, SEG_ADDRESS_DIGIT_2, adcVal%10);
110     adcVal/=10;
111     send_7seg(SEG_gpio, DIN_pin, CS_pin, CLK_pin, SEG_ADDRESS_DIGIT_3, adcVal%10);
112     adcVal/=10;
113     send_7seg(SEG_gpio, DIN_pin, CS_pin, CLK_pin, SEG_ADDRESS_DIGIT_4, adcVal%10);
114     adcVal/=10;
115 }
116
117 void lab_10_2(){
118     EXTI_Setup();
119     // USARTInit();
120     ADCInit();
121
122     SystemClock_Config_Interrupt(4, 1000000);
123
124     init_led(GPIOA,5);
125     while(1){
126         //light();
127         //ADC1_2_IRQHandler();
128         //EXTI15_10_IRQHandler();
129         //TIM2_IRQHandler();
130         //display_two_decimal(SEG_gpio, DIN_pin, CS_pin, CLK_pin, adcVal);
131     }
132 }
133
134 int main(){
135     GPIO_init();
136     if(init_7seg(SEG_gpio, DIN_pin, CS_pin, CLK_pin) != 0){
137         return -1;
138     }
139     send_7seg(SEG_gpio, DIN_pin, CS_pin, CLK_pin, SEG_ADDRESS_DECODE_MODE, 0xFF);
140     send_7seg(SEG_gpio, DIN_pin, CS_pin, CLK_pin, SEG_ADDRESS_SCAN_LIMIT, 0x04);
141     send_7seg(SEG_gpio, DIN_pin, CS_pin, CLK_pin, SEG_ADDRESS_SHUTDOWN, 0x01);
142
143     lab_10_2();
144     while(1){}
145     return 0;
146 }

```

PART 2. (40%) 實作題

Lab 7.2: ADC 變壓輸入:

請完成實驗 錄影紀錄實驗結果

- 選用三個不同電阻取代光敏電阻，計算出這三個電阻跨壓，不可高於 5V，用 ADC 分別讀取這三個跨壓，，然後把取樣結果顯示在 7-segment 上。
- 計算出電阻值與 ADC 讀值的轉換關係式並寫入程式中，按按鍵(blue button)切換顯示電阻值與 ADC 讀值。

```
1  #include "stm321476xx.h"
2  #include "helper_functions.h"
3  #include "usart.h"
4  #include "adc.h"
5  #include "led_button.h"
6  #include "7seg.h"
7  #include "stdio.h"
8
9  #define BUTTON_gpio GPIOC
10 #define BUTTON_pin 13
11 #define lab10_2
12 #define SEG_gpio GPIOC
13 #define DIN_pin 3
14 #define CS_pin 4
15 #define CLK_pin 5
16
17 double resistor;
18 int adcVal = 0;
19 int state = 0;
20 void GPIO_init(){
21     // Setup FPU
22     SCB->CPACR |= (0xF << 20);
23     __DSB();
24     __ISB();
25
26     RCC->AHB2ENR |= RCC_AHB2ENR_GPIOAEN; // PA5 LED
27     RCC->AHB2ENR |= RCC_AHB2ENR_GPIOCEN; // PC13 Button
28     RCC->AHB2ENR |= RCC_AHB2ENR_ADCEN; // ADC
29
30     // Setup Button
31     GPIOC->MODER &= ~GPIO_MODER_MODE13_Msk;
32     GPIOC->MODER |= (0x0 << GPIO_MODER_MODE13_Pos);
33
34     // Setup PA5 LED
35     GPIOA->MODER &= ~GPIO_MODER_MODE5_Msk;
36     GPIOA->MODER |= (0x1 << GPIO_MODER_MODE5_Pos);
37
38     // PC0 ADC
39     GPIOC->MODER &= ~GPIO_MODER_MODE0_Msk;
40     GPIOC->MODER |= (0x3 << GPIO_MODER_MODE0_Pos);
41     GPIOC->ASCR |= GPIO_ASCR_ASC0;
42 }
43
44 void EXTI_Setup(){
45     // Enable SYSCFG CLK
46     RCC->APB2ENR |= RCC_APB2ENR_SYSCFGEN;
47     // Select output bits
48     SYSCFG->EXTICR[3] &= ~SYSCFG_EXTICR4_EXTI13_Msk;
49     SYSCFG->EXTICR[3] |= (2 << SYSCFG_EXTICR4_EXTI13_Pos);
50     // Enable interrupt
51     EXTI->IMR1 |= EXTI_IMR1_IM13;
```



```

51     EXTI->IMR1 |= EXTI_IMR1_IM13;
52     // Enable Falling Edge
53     EXTI->RTSR1 |= EXTI_RTSR1_RT13;
54     // Enable NVIC
55     NVIC_EnableIRQ(EXTI15_10_IRQn);
56 }
57
58 void ADCInit(){
59     ADCResolution(ADC1, 0);           // 12 bits
60     ADCContinuousConversion(ADC1, 0); // enable continuous conversion
61     ADCDataAlign(ADC1, 0);            // set right align
62     ADCCommonDualMode(0);             // independent mode
63     ADCCommonClockMode(1);            // hclk / 1
64     ADCCommonPrescaler(0);            // div 1
65     ADCCommonDMAMode(0);              // disable dma
66     ADCCommonDelayTwoSampling(0b0100); // 5 adc clk cycle
67     ADCChannel(ADC1, 1, 1, 2);        // channel 1, rank 1, 12.5 adc clock cycle
68     ADCWakeup(ADC1);
69     ADCInterrupt(ADC1, ADC_IER_EOCIE, 1);
70     NVIC_EnableIRQ(ADC1_2_IRQn);
71     ADCEnable(ADC1);
72 }
73
74 void SysTick_Handler() {
75     if(SysTick->CTRL & SysTick_CTRL_COUNTFLAG_Msk){
76         //ADCStartConversion(ADC1);
77         light();
78         handler();
79         toggle_output(GPIOA,5);
80     }
81     return;
82 }
83
84 void ADC1_2_IRQHandler(){
85     if(ADC1->ISR & ADC_ISR_EOC){
86         ADC1->ISR &= ADC_ISR_EOC;
87         resistor = ADCGetValue(ADC1);
88         resistor = (3.3 / 4095) * resistor;
89         resistor = (3.3 - resistor) * 2000 / resistor;
90     }
91 }
92
93 void startADC() {
94     while (!(ADC1->ISR & ADC_ISR_ADRDY)) ADC1->CR |= ADC_CR_ADEN; // TURN ON
95     ADC1->ISR = ADC_ISR_EOC | ADC_ISR_EOS | ADC_ISR_OVR; // Clear flags
96     ADC1->CR |= ADC_CR_ADSTART; // START CONV
97 }
98
99 void light() {
100     startADC();
101     while (!(ADC1->ISR & ADC_ISR_EOC));
102     adcVal = ADC1->DR;
103 }
104
105 void handler(){
106     double Vsample = (3.3/4095)*adcVal;
107     double current = Vsample/10000;
108     if(state==1){
109         display_two_decimal(SEG_gpio, DIN_pin, CS_pin, CLK_pin, (3.3 - Vsample));
110     }else{
111         int k=0;
112         int temp;
113         adcVal = (3.3 - Vsample)/current;
114         while(adcVal !=0){
115             adcVal = adcVal/10;
116             k++;
117         }
118         if(k>8){
119             k=0;
120         }
121         adcVal = (3.3 - Vsample)/current;

```

```

121     adcVal = (3.3 - Vsample)/current;
122     display_number(SEG_gpio, DIN_pin, CS_pin, CLK_pin, adcVal, k);
123 }
124 }
125
126 void lab_10_2(){
127     //EXTI_Setup();
128     // USARTInit();
129     ADCInit();
130     SystemClock_Config_Interrupt(4, 1000000);
131     init_led(GPIOA,5);
132     while(1){
133         //Light();
134         //ADC1_2_IRQHandler();
135         //EXTI15_10_IRQHandler();
136         //TIM2_IRQHandler();
137         //display_two_decimal(SEG_gpio, DIN_pin, CS_pin, CLK_pin, adcVal);
138
139         int ch_state = 0;
140         int button_press_persecond_cycle = 10;
141         int debounce_cycle = 100;
142         int debounce_threshold = 70;
143         int last_botton_pos =1;
144
145         for(int i=0; i<button_press_persecond_cycle; i++){
146             int pos_cnt = 0; //count
147             for(int a=0; a<debounce_cycle; a++){
148                 if(read_gpio(BUTTON_gpio, BUTTON_pin) == 0){
149                     pos_cnt++;
150                 }
151             }
152             if(pos_cnt > debounce_threshold){
153                 if(last_botton_pos == 1){
154                     last_botton_pos = 0;
155                 }
156             }else{
157                 if(last_botton_pos == 0){
158                     if(state == 0){
159                         state = 1;
160                     }else{
161                         state = 0;
162                     }
163                     last_botton_pos = 1;
164                 }
165             }
166         }
167     }
168 }
169
170 int main(){
171     GPIO_init();
172     if(init_7seg(SEG_gpio, DIN_pin, CS_pin, CLK_pin) != 0){
173         return -1;
174     }
175     send_7seg(SEG_gpio, DIN_pin, CS_pin, CLK_pin, SEG_ADDRESS_DECODE_MODE, 0xFF);
176     send_7seg(SEG_gpio, DIN_pin, CS_pin, CLK_pin, SEG_ADDRESS_SCAN_LIMIT, 0xFF);
177     send_7seg(SEG_gpio, DIN_pin, CS_pin, CLK_pin, SEG_ADDRESS_SHUTDOWN, 0x01);
178
179     lab_10_2();
180     while(1){}
181     return 0;
182 }

```


PART 3. (50%) 實作題

Lab 7.3: 練習馬達控制

請完成實驗 錄影及截圖紀錄實驗結果並附上程式碼(main.c 及 include 之.h, .c 檔案)

利用 SysTick timer 和 SG90 伺服馬達，每 1 秒順時鐘轉動 45 度，4 秒達到 180 度後，停頓一秒後每秒往逆時鐘方向旋轉 45 度直到回到原點。

```
1  #include "stm321476xx.h"
2  #include "timer.h"
3
4  int times = 1;
5  void SysTick_Handler() {
6      if(SysTick->CTRL & SysTick_CTRL_COUNTFLAG_Msk){
7          times ++;
8          //----- timer_init(TIM2, 1000, 20000); -----//
9          TIM2->PSC = (uint32_t)(1000-1);          // PreScalser
10         TIM2->ARR = (uint32_t)(150000-1);          // Reload value
11         TIM2->EGR |= TIM_EGR_UG;                  // Reinitialize the counter
12
13         //----- timer_start(TIM2); -----//
14         TIM2->CR1 |= TIM_CR1_CEN;
15         if (times % 4 == 1 || times % 4 == 2)
16             TIM2->CCR1 = 20;
17         else
18             TIM2->CCR1 = (uint32_t)(1);
19     }
20 }
21
22 int main()
23 {
24     // Configure SysTick clk to 10 Mhz and interrupt every 4s
25     SystemClock_Config_Interrupt(10, 40000000);
26
27     //----- GPIO_init_AF(); -----//
28     RCC->AHB2ENR |= RCC_AHB2ENR_GPIOAEN;
29     // Set to Alternate function mode
30     GPIOA->MODER &= ~GPIO_MODER_MODE0_Msk;
31     GPIOA->MODER |= (2 << GPIO_MODER_MODE0_Pos);
32     // Set AFRL
33     GPIOA->AFR[0] &= ~GPIO_AFR[0]_AFSEL0_Msk;
34     GPIOA->AFR[0] |= (1 << GPIO_AFR[0]_AFSEL0_Pos);
35
36     //----- timer_enable(TIM2); -----//
37     RCC->APB1ENR1 |= RCC_APB1ENR1_TIM2EN;
38
39     // ----- timer_init(TIM2, 1000, 20000); -----//
40     TIM2->PSC = (uint32_t)(1000-1);          // PreScalser
41     TIM2->ARR = (uint32_t)(1000-1);          // Reload value
42     TIM2->EGR |= TIM_EGR_UG;                  // Reinitialize the counter
43
44     //----- sg90_init(); -----//
45     // PA0 for PWM
46     // PWM mode 1
47     TIM2->CCMR1 &= ~TIM_CCMR1_OC1M_Msk;
48     TIM2->CCMR1 |= (6 << TIM_CCMR1_OC1M_Pos);
49     // OCPreLoad_Enable
```

```

49 // OCPreload_Enable
50 TIM2->CCMR1 &= ~TIM_CCMR1_OC1PE_Msk;
51 TIM2->CCMR1 |= (1 << TIM_CCMR1_OC1PE_Pos);
52 // Active high for channel 1 polarity
53 TIM2->CCER &= ~TIM_CCER_CC1P_Msk;
54 TIM2->CCER |= (0 << TIM_CCER_CC1P_Pos);
55 // Enable for channel 1 output
56 TIM2->CCER &= ~TIM_CCER_CC1E_Msk;
57 TIM2->CCER |= (1 << TIM_CCER_CC1E_Pos);
58
59 //----- timer_init(TIM2, 1000, 20000); -----//
60 TIM2->PSC = (uint32_t)(10000-1); // PreScaler
61 TIM2->ARR = (uint32_t)(1000-1); // Reload value
62 TIM2->EGR |= TIM_EGR_UG; // Reinitialize the counter
63
64 //----- timer_start(TIM2); -----//
65 TIM2->CR1 |= TIM_CR1_CEN;
66
67 while (1){
68
69 }
70 return 0;
71 }

```

本作業參考自: DCP1155 Microprocessor System Lab 2016

曹孝櫟教授 國立交通大學 資訊工程學系 Lab7