# 實驗五 實驗結報

0316213 蒲郁文 & 0316323 薛世恩

## 實驗名稱

MAX7219 & 7-Seg LED

## 實驗目的

熟悉 output 跟 MAX7219 & 7-Seg LED

## 實驗步驟

### show1~9 A~F

```
.syntax
unified
    .cpu cortex-m4
    .thumb


.data
    arr: .byte 0b01111110, 0b00110000, 0b01101101, 0b01111001,
0b00110011, 0b01011011, 0b01011111, 0b01110000, 0b01111111,
0b01111011, 0b01110111, 0b00011111, 0b01001110, 0b00111101,
0b01001111, 0b01000111
    @ arr: a0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, b, C, d, E, F


.text
    .global main
    .equ RCC_AHB2ENR,  0x4002104C

    .equ DECODE_MODE,  0x09
    .equ DISPLAY_TEST, 0x0F
    .equ SCAN_LIMIT,   0x0B
    .equ INTENSITY,    0x0A
    .equ SHUTDOWN,     0x0C

    .equ MAX7219_DIN,  0x20 @ PA5
    .equ MAX7219_CS,   0x40 @ PA6
    .equ MAX7219_CLK,  0x80 @ PA7

    .equ GPIOA_BASE,   0x48000000
    .equ BSRR_OFFSET,  0x18 @ set bit
    .equ BRR_OFFSET,   0x28 @ clear bit

main:
    bl   gpio_init
    bl   max7219_init

display_0_to_f:
    mov  r2, 0x0
    ldr  r3, =arr
    b    loop

gpio_init:
    mov  r0, 0b00000000000000000000000000000001
    ldr  r1, =RCC_AHB2ENR
    str  r0, [r1]

    ldr  r1, =GPIOA_BASE @ GPIOA_MODER
    ldr  r2, [r1]
    and  r2, 0b11111111111111110000001111111111
    orr  r2, 0b00000000000000000101010000000000
    str  r2, [r1]


    add  r1, 0x4 @ GPIOA_OTYPER
    ldr  r2, [r1]
    and  r2, 0b11111111111111111111111100011111
    str  r2, [r1]


    add  r1, 0x4 @ GPIOA_SPEEDER
    ldr  r2, [r1]
    and  r2, 0b11111111111111110000001111111111
    orr  r2, 0b00000000000000000101010000000000
    str  r2, [r1]


    bx   lr


loop:
    mov  r0, 0x1
    ldrb r1, [r3, r2]
    bl   max7219_send


    ldr  r0, =4000000 @ delay 1s
    movs r0, r0
    bl   delay


    add  r2, 0x1
    cmp  r2, 0x10
    bne  loop
    mov  r2, 0x0
    b    loop


max7219_send:
    @ input parameter: r0 is ADDRESS , r1 is DATA
    push {r0, r1, r2, r3, r4, r5, r6, r7, r8, lr}
    lsl  r0, r0, 0x8
    add  r0, r1
    ldr  r1, =GPIOA_BASE
    ldr  r2, =MAX7219_CS
    ldr  r3, =MAX7219_DIN
    ldr  r4, =MAX7219_CLK
    ldr  r5, =BSRR_OFFSET
    ldr  r6, =BRR_OFFSET
    ldr  r7, =0x0F @ currently sending r7-th bit


max7219_send_loop:
    mov  r8, 0x1
    lsl  r8, r8, r7
    str  r4, [r1, r6] @ clk -> 0
    tst  r0, r8 @ ANDS but discard result
    beq  max7219_send_clear_bit
    str  r3, [r1, r5] @ din -> 1
    b    max7219_send_check_done


max7219_send_clear_bit:
    str  r3, [r1, r6] @ din -> 0


max7219_send_check_done:
    str  r4, [r1, r5] @ clk -> 1
    subs r7, 0x1
    bge  max7219_send_loop
    str  r2, [r1, r6] @ cs -> 0
    str  r2, [r1, r5] @ cs -> 1
    pop  {r0, r1, r2, r3, r4, r5, r6, r7, r8, pc}


max7219_init:
    push {r0, r1, r2, lr}


    ldr  r0, =DECODE_MODE
    ldr  r1, =0x0
    bl   max7219_send


    ldr  r0, =DISPLAY_TEST
    ldr  r1, =0x0
    bl   max7219_send


    ldr  r0, =SCAN_LIMIT
    ldr  r1, =0x0
    bl   max7219_send


    ldr  r0, =INTENSITY
    ldr  r1, =0xA
    bl   max7219_send


    ldr  r0, =SHUTDOWN
    ldr  r1, =0x1
    bl   max7219_send


    pop  {r0, r1, r2, pc}


delay:
    beq  delay_end
    subs r0, 0x4
    b    delay

delay_end:
    bx   lr
```

## show student ID

```
    .syntax
unified
    .cpu cortex-m4
    .thumb

.data
    arr: .byte 0x0, 0x3, 0x1, 0x6, 0x2, 0x1, 0x3
    @ arr: .byte 0x0, 0x3, 0x1, 0x6, 0x3, 0x2, 0x3

.text
    .global main

    .equ RCC_AHB2ENR,  0x4002104C

    .equ DECODE_MODE,  0x09
    .equ DISPLAY_TEST, 0x0F
    .equ SCAN_LIMIT,   0x0B
    .equ INTENSITY,    0x0A
    .equ SHUTDOWN,     0x0C
    .equ MAX7219_DIN,  0x20 @ PA5
    .equ MAX7219_CS,   0x40 @ PA6
    .equ MAX7219_CLK,  0x80 @ PA7


    .equ GPIOA_BASE,   0x48000000
    .equ BSRR_OFFSET,  0x18 @ set bit
    .equ BRR_OFFSET,   0x28 @ clear bit


main:
    bl   gpio_init
    bl   max7219_init


display_arr:
    mov  r0, 0x7
    mov  r2, 0x0
    ldr  r3, =arr
    b    loop


gpio_init:
    mov  r0, 0b00000000000000000000000000000001
    ldr  r1, =RCC_AHB2ENR
    str  r0, [r1]


    ldr  r1, =GPIOA_BASE @ GPIOA_MODER
    ldr  r2, [r1]
    and  r2, 0b11111111111111110000001111111111
    orr  r2, 0b00000000000000000101010000000000
    str  r2, [r1]


    add  r1, 0x4 @ GPIOA_OTYPER
    ldr  r2, [r1]
    and  r2, 0b11111111111111111111111100011111
    str  r2, [r1]


    add  r1, 0x4 @ GPIOA_SPEEDER
    ldr  r2, [r1]
    and  r2, 0b11111111111111111000000111111111
    orr  r2, 0b00000000000000000101010000000000
    str  r2, [r1]


    bx   lr


loop:
    ldrb r1, [r3, r2]
    bl   max7219_send


    sub  r0, 0x1
    add  r2, 0x1
    cmp  r2, 0x8
    bne  loop


    mov  r0, 0x7
    mov  r2, 0x0
    b    loop


max7219_send:
    @ input parameter: r0 is ADDRESS , r1 is DATA
    push {r0, r1, r2, r3, r4, r5, r6, r7, r8, lr}
    lsl  r0, r0, 0x8
    add  r0, r1
    ldr  r1, =GPIOA_BASE
    ldr  r2, =MAX7219_CS
    ldr  r3, =MAX7219_DIN
    ldr  r4, =MAX7219_CLK
    ldr  r5, =BSRR_OFFSET
    ldr  r6, =BRR_OFFSET
    ldr  r7, =0x0F @ currently sending r7-th bit


max7219_send_loop:
    mov  r8, 0x1
    lsl  r8, r8, r7
    str  r4, [r1, r6] @ clk -> 0
    tst  r0, r8 @ ANDS but discard result
    beq  max7219_send_clear_bit
    str  r3, [r1, r5] @ din -> 1
    b    max7219_send_check_done


max7219_send_clear_bit:
    str  r3, [r1, r6] @ din -> 0


max7219_send_check_done:
    str  r4, [r1, r5] @ clk -> 1
    subs r7, 0x1
    bge  max7219_send_loop
    str  r2, [r1, r6] @ cs -> 0
    str  r2, [r1, r5] @ cs -> 1
    pop  {r0, r1, r2, r3, r4, r5, r6, r7, r8, pc}


max7219_init:
    push {r0, r1, r2, lr}


    ldr  r0, =DECODE_MODE
    ldr  r1, =0xFF
    bl   max7219_send


    ldr  r0, =DISPLAY_TEST
    ldr  r1, =0x0
    bl   max7219_send


    ldr  r0, =SCAN_LIMIT
    ldr  r1, =0x6
    bl   max7219_send


    ldr  r0, =INTENSITY
    ldr  r1, =0xA
    bl   max7219_send


    ldr  r0, =SHUTDOWN
    ldr  r1, =0x1
    bl   max7219_send


    pop  {r0, r1, r2, pc}


delay:
    beq  delay_end
    subs r0, 0x4
    b    delay


delay_end:
    bx   lr
```

## show 費伯納數列

```
.syn
tax
unif
ied
```

```
    .cpu cortex-m4
    .thumb

.data
    ans: .asciz "0112358132134558914423337761098715972584418167651094617711286574636875025121393196418317811514229832040134626921783093524578570288792274651493035224157817390881696324598 6:1"
    len: .byte 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x1, 0x2, 0x2, 0x2, 0x2, 0x2, 0x3, 0x3, 0x3, 0x3, 0x3, 0x4, 0x4, 0x4, 0x4, 0x5, 0x5, 0x5, 0x5, 0x5, 0x6, 0x6, 0x6, 0x6, 0x6, 0x7, 0x7, 0x7, 0x7, 0x7, 0x8, 0x8, 0x8, 0x8, 0x2

.text
    .global main

    .equ RCC_AHB2ENR,  0x4002104C

    .equ DECODE_MODE,  0x09
    .equ DISPLAY_TEST, 0x0F
    .equ SCAN_LIMIT,   0x0B
    .equ INTENSITY,    0x0A
    .equ SHUTDOWN,     0x0C

    .equ MAX7219_DIN,  0x20 @ PA5
    .equ MAX7219_CS,   0x40 @ PA6
    .equ MAX7219_CLK,  0x80 @ PA7

    .equ GPIOA_BASE,   0x48000000
    .equ GPIOC_BASE,   0x48000800
    .equ BSRR_OFFSET,  0x18 @ set bit
    .equ BRR_OFFSET,   0x28 @ clear bit
```

```
main:
    bl   gpio_init
    bl   max7219_init


display_ans:
    ldr  r2, =len
    ldr  r3, =ans
    b    loop_init


gpio_init:
    mov  r0, 0b00000000000000000000000000000101
    ldr  r1, =RCC_AHB2ENR
    str  r0, [r1]


    ldr  r1, =GPIOA_BASE @ GPIOA_MODER
    ldr  r2, [r1]
    and  r2, 0b11111111111111110000001111111111
    orr  r2, 0b00000000000000000101010000000000
    str  r2, [r1]


    add  r1, 0x4 @ GPIOA_OTYPER
    ldr  r2, [r1]
    and  r2, 0b11111111111111111111111100011111
    str  r2, [r1]


    add  r1, 0x4 @ GPIOA_SPEEDER
    ldr  r2, [r1]
    and  r2, 0b11111111111111110000001111111111
    orr  r2, 0b00000000000000000101010000000000
    str  r2, [r1]
    ldr  r1, =GPIOC_BASE @ GPIOC_MODER
    ldr  r2, [r1]
    and  r2, 0b11110011111111111111111111111111
    str  r2, [r1]


    add  r10, r1, 0x10
    bx   lr


loop_init:
    mov  r0, 0x1
    mov  r1, 0x0F
    bl   max7219_send


    mov  r0, 0x2
    mov  r1, 0x0F
    bl   max7219_send


    mov  r0, 0x3
    mov  r1, 0x0F
    bl   max7219_send


    mov  r0, 0x4
    mov  r1, 0x0F
    bl   max7219_send


    mov  r0, 0x5
    mov  r1, 0x0F
    bl   max7219_send


    mov  r0, 0x6
    mov  r1, 0x0F
    bl   max7219_send


    mov  r0, 0x7
    mov  r1, 0x0F
    bl   max7219_send


    mov  r0, 0x8
    mov  r1, 0x0F
    bl   max7219_send


    ldrb r0, [r2]


loop_inner:
    ldrb r1, [r3]
    sub  r1, 0x30 @ char - '0' = digit
    bl   max7219_send


    subs r0, 0x1
    add  r3, 0x1
    bne  loop_inner


loop:
    ldr  r0, =len
    add  r0, 40
    cmp  r2, r0
    it   eq
    bleq loop_last
    add  r2, 0x1


    mov  r11, 0x1
    mov  r12, 0x1
    bl   check_button_init


    b    loop_init


loop_last:
    sub  r2, 0x1
    sub  r3, 0x2
    bx   lr


max7219_send:
    @ input parameter: r0 is ADDRESS , r1 is DATA
    push {r0, r1, r2, r3, r4, r5, r6, r7, r8, lr}
    lsl  r0, r0, 0x8
    add  r0, r1
    ldr  r1, =GPIOA_BASE
    ldr  r2, =MAX7219_CS
    ldr  r3, =MAX7219_DIN
    ldr  r4, =MAX7219_CLK
    ldr  r5, =BSRR_OFFSET
    ldr  r6, =BRR_OFFSET
    ldr  r7, =0x0F @ currently sending r7-th bit


max7219_send_loop:
    mov  r8, 0x1
    lsl  r8, r8, r7
    str  r4, [r1, r6] @ clk -> 0
    tst  r0, r8 @ ANDS but discard result
    beq  max7219_send_clear_bit
    str  r3, [r1, r5] @ din -> 1
    b    max7219_send_check_done


max7219_send_clear_bit:
    str  r3, [r1, r6] @ din -> 0


max7219_send_check_done:
    str  r4, [r1, r5] @ clk -> 1
    subs r7, 0x1
    bge  max7219_send_loop
    str  r2, [r1, r6] @ cs -> 0
    str  r2, [r1, r5] @ cs -> 1
    pop  {r0, r1, r2, r3, r4, r5, r6, r7, r8, pc}


max7219_init:
    push {r0, r1, r2, lr}


    ldr  r0, =DECODE_MODE
    ldr  r1, =0xFF
    bl   max7219_send


    ldr  r0, =DISPLAY_TEST
    ldr  r1, =0x0
    bl   max7219_send


    ldr  r0, =SCAN_LIMIT
    ldr  r1, =0x7
    bl   max7219_send


    ldr  r0, =INTENSITY
    ldr  r1, =0xA
    bl   max7219_send


    ldr  r0, =SHUTDOWN
    ldr  r1, =0x1
    bl   max7219_send


    pop  {r0, r1, r2, pc}


check_button_init:
    ldr  r0, =4000000 @ delay 1s
    movs r0, r0
    b    check_button_delay


check_button_delay:
    beq  check_button_init
    ldr  r1, =0b1111111111111111
    ands r1, r0
    beq  check_button @ branch every 32.768 ms
    subs r0, 0x8
    b    check_button_delay


check_button:
    @ r10 gpio button input
    @ r11 latest button value
    @ r12 confirmed button value
    ldrh r1, [r10]
    lsr  r1, 13
    mov  r4, 1
    and  r1, r4
    cmp  r1, r11
    mov  r11, r1
    beq  check_button_confirmed
    subs r0, 8
    b    check_button_delay


check_button_confirmed:
    subs r1, r11, r12
    cmp  r1, 1
    mov  r12, r11
    beq  check_button_end
    subs r0, 8
    b    check_button_delay


check_button_end:
    bx   lr
```

## 實驗結果與問題回答

### Show1~9&A~F

- 首先先將它存成陣列
- 然後將它一個一個照著 delay  讀取
- 然將設定成對的輸出方式然後一直變換輸出
-

### Show student id

- 將學生 ID 放在陣列一次一個讀
- 然後將其輸出

## 費伯納數列

**依照前幾次作業將其組合，然後當它輸出超過，就顯示-1。**

**然後做好 debounce。**

- ## 心得討論與應用聯想

真的超級難，然後又在期中考周出作業，寫超級久，BUG 也超級多。