LAB: GPIO Digital InOut 7-segment

LAB: GPIO Digital InOut 7-segment

Date: 2024-09-20

Author/Partner: Jong-Hyeon Kim

Github: Repository link

Demo Video: Problem 0 / Problem 1 / Problem 2

Introduction

In this lab, you are required to create a simple program to control a 7-segment display to show a decimal number (0~9) that increases by pressing a push-button.

Requirement

Hardware

- MCU
  - NUCLEO-F411RE
- Actuator/Sensor/Others:
  - 7-segment display(5101ASR)
  - Array resistor (330 ohm)
  - decoder chip(74LS47)
  - breadboard

Software

- Keil uVision, CMSIS, EC_HAL library

Exercise

Port/Pin	Description	Register setting
Port A Pin 5	Clear Pin5 mode	GPIOA->MODER &= ~(3<<(5*2))
Port A Pin 5	Set Pin5 mode = Output	GPIOA->MODER |= (1<<(5*2))
Port A Pin 6	Clear Pin6 mode	GPIOA->MODER &=~(3<<(6*2))
Port A Pin 6	Set Pin6 mode = Output	GPIOA->MODER |=(1<<(6*2))
Port A Pin Y	Clear PinY mode	GPIOA->MODER &=~(3<<(Y*2))
Port A Pin Y	Set PinY mode = Output	GPIOA->MODER |= (1<<(Y*2))
Port A Pin 5~9	Clear Pin5~9 mode	GPIOA->MODER &=~(0x3FF<<(5*2))
	Set Pin5~9 mode = Output	GPIOA->MODER |= (0x155<<(5*2))
Port X Pin Y	Clear Pin Y mode	GPIOX->MODER &= ~(3<<(Y*2))

Port/Pin	Description	Register setting
	Set Pin Y mode = Output	GPIOX->MODER |= (1<<(Y*2))
Port A Pin5	Set Pin5 otype=push-pull	GPIOA->OTYPER &= ~(1<<5)
Port A PinY	Set PinY otype=push-pull	GPIOA->OTYPER &= ~(1<<Y)
Port A Pin5	Set Pin5 ospeed=Fast	GPIOA->OSPEEDR |= (3<<(5*2))
Port A PinY	Set PinY ospeed=Fast	GPIOA->OSPEEDR |= (3<<(5*2))
Port A Pin 5	Set Pin5 PUPD=no pull up/down	GPIOA->PUPDR &= ~(3<<(5*2))
Port A Pin Y	Set PinY PUPD=no pull up/down	GPIOA->PUPDR &= ~(3<<(Y*2))

Problem 0: Connection of 7-Segment Display and Decoder

video link

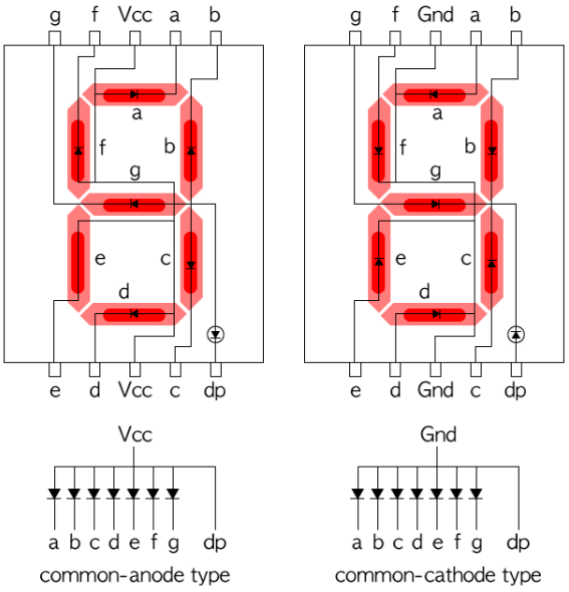
(In the lecture we already checked if 7-Segment display and 7-segment Decoder are worked, So, I will upload only Discussion part for this problem. Thank you.)

Discussion

1. **Draw the truth table for the BCD 7-segment decoder with the 4-bit input.**

D	C	B	A	a	b	c	d	e	f	g
0	0	0	0	0	0	0	0	0	0	1
0	0	0	1	1	0	0	1	1	1	1
0	0	1	0	0	0	1	0	0	1	0
0	0	1	1	0	0	0	0	1	1	0
0	1	0	0	1	0	0	1	1	0	0
0	1	0	1	0	1	0	0	1	0	0
0	1	1	0	1	1	0	0	0	0	0
0	1	1	1	0	0	0	1	1	1	1
1	0	0	0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	1	1	0	0

2. **What are the common cathode and common anode of 7-segment display?**



The 7-segment is made by connecting several leads. The points where the legs of the leads meet each other are made into a single terminal, and the common anode connects the positive electrode to all LEDs one by one. So, the opposite direction should give GND to the input to turn on the light as the current flows from (+) to (-). Likewise, the common cathode is the principle that the negative electrode is shared by all eight leads, so you have to give 5V to the input on the other side to turn on the light while flowing from (+) to (-).

3. **Does the LED of a 7-segment display (common anode) pin turn ON when 'HIGH' is given to the LED pin from the MCU?**

No, it can not in theoretically. Because of VCC is shared every LED elements, then oppsite site has to be GND to the pin turn on.

Problem 1: Display a Number with Button Press

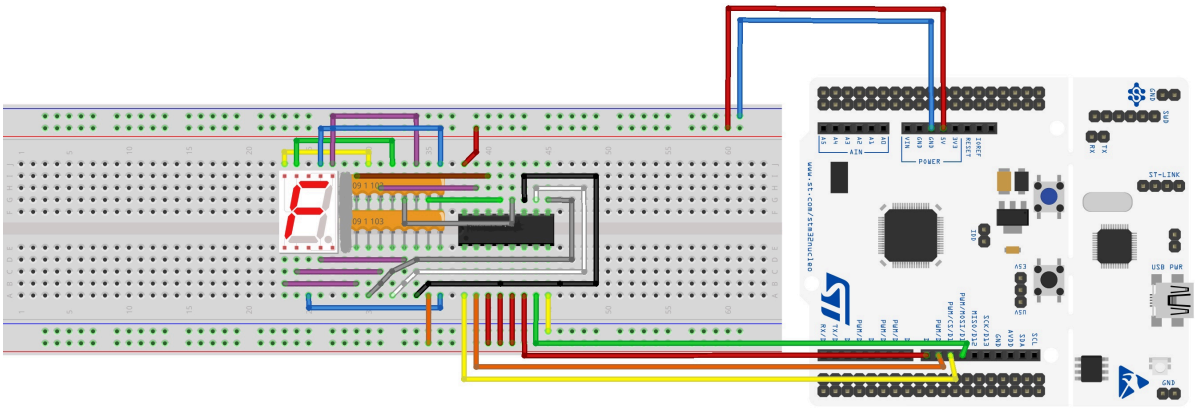
Procedure

- num: 0 to 9 only (unsigned)
- Create a code that increase the displayed number from 0 to 9 with each button press.
- After the number '9', it should start from '0' again.

Configuration

Digital In for Button (B1)	**Digital Out for 7-Segment**
Digital In	Digital Out
PC13	PA7, PB6, PC7, PA9
PULL-UP	Push-Pull, No Pull-up-Pull-down, Medium Speed

Circuit Diagram



Code

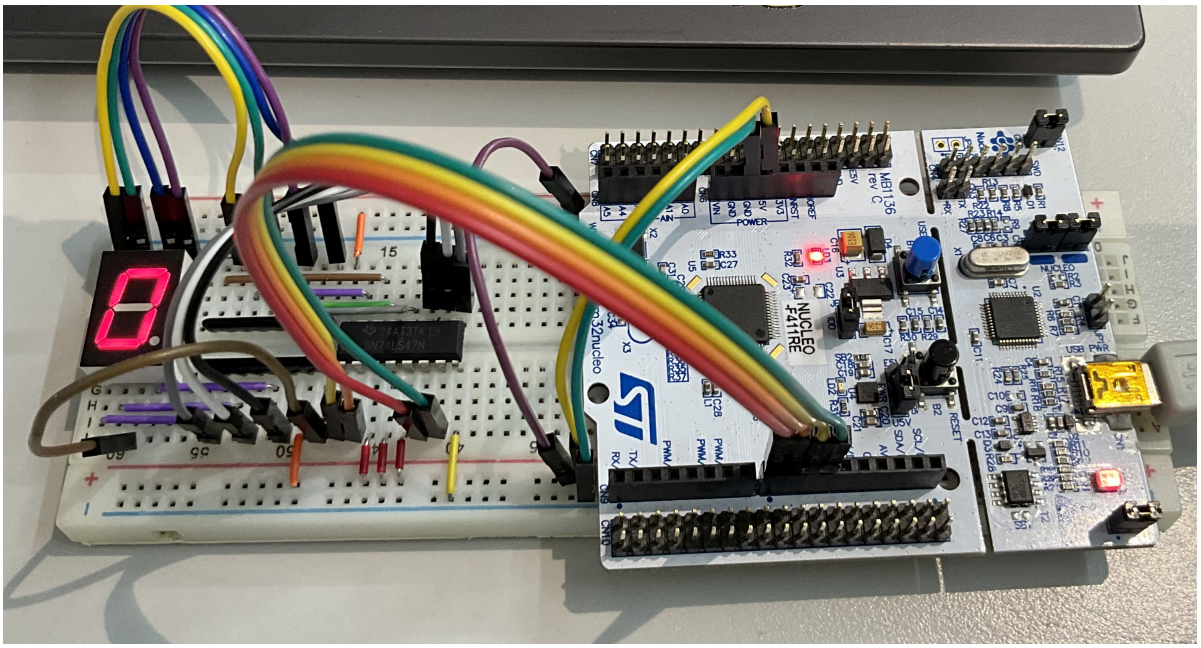
[Problem 1 Code link in Github]

```
#include "stm32f4xx.h"
#include "ecGPIO2.h"
#include "ecRCC2.h"

void setup(void);

int main(void) {
    // Initialiization ---------------------------------------------------------
    setup();
    unsigned int cnt = 0;

    // Inifinite Loop ----------------------------------------------------------
    while(1){
        sevensegment_display(cnt % 10);                     // Give the argument
"cnt%10" to the fucntion

        if(GPIO_read(BUTTON_PIN) == 0) cnt++;
        if (cnt > 9) cnt = 0;
        for(int i = 0; i < 500000;i++){}                    // delay_ms(500);
    }
}

// Initialiization
void setup(void){
    RCC_HSI_init();
    GPIO_init(BUTTON_PIN, INPUT);                               // calls
RCC_GPIOC_enable()
    sevensegment_display_init(PA_7, PB_6, PC_7, PA_9);          // Decoder input
A,B,C,D
}

// Initialiization for 7-segment_display function
```

```
void sevensegment_display_init(PinName_t pinNameA, PinName_t pinNameB, PinName_t
pinNameC, PinName_t pinNameD){                                  // Setup for the
A,B,C,D for 7-Segment Decoder
    // PA_7
    GPIO_init(pinNameA, OUTPUT);            // calls RCC_GPIOA_enable() / OUTPUT
    GPIO_ospeed(pinNameA, MSPEED);          // Medium-Speed
    GPIO_pupd(pinNameA, EC_NP);             // No Pull up, Pull down
    GPIO_otype(pinNameA, PSPL);             // Push-Pull
    // PB_6
    GPIO_init(pinNameB, OUTPUT);            // calls RCC_GPIOB_enable() / OUTPUT
    GPIO_ospeed(pinNameB, MSPEED);          // Medium-Speed
    GPIO_pupd(pinNameB, EC_NP);             // No Pull up, Pull down
    GPIO_otype(pinNameB, PSPL);             // Push-Pull
    // PC_7
    GPIO_init(pinNameC, OUTPUT);            // calls RCC_GPIOC_enable() / OUTPUT
    GPIO_ospeed(pinNameC, MSPEED);          // Medium-Speed
    GPIO_pupd(pinNameC, EC_NP);             // No Pull up, Pull down
    GPIO_otype(pinNameC, PSPL);             // Push-Pull
    // PA_9
    GPIO_init(pinNameD, OUTPUT);            // calls RCC_GPIOA_enable() / OUTPUT
    GPIO_ospeed(pinNameD, MSPEED);          // Medium-Speed
    GPIO_pupd(pinNameD, EC_NP);             // No Pull up, Pull down
    GPIO_otype(pinNameD, PSPL);             // Push-Pull
}

// Displaying the "0"-"9" in 7-segment display by Decoder A,B,C,D
void sevensegment_display(uint8_t num){     // num: Number for diplaying
    int BCD_A= 0;                           // PA_7 - A (LSB: Least Siginificant
Bit)
    int BCD_B= 0;                           // PB_6 - B
    int BCD_C= 0;                           // PC_7 - C
    int BCD_D= 0;                           // PA_9 - D (MSB: Least Siginificant
Bit)

    for(int i=1;i<=num;i++){
                BCD_A = !BCD_A;             // Toggle every iteration.
        if(i%2==0) BCD_B = !BCD_B;          // Toggle at the even number
        if(i%4==0) BCD_C = !BCD_C;          // Toggle every 4th time.
        if(i%8==0) BCD_D = !BCD_D;          // Toggle every 8th time.
    }

    GPIO_write(PA_7, BCD_A);
    GPIO_write(PB_6, BCD_B);
    GPIO_write(PC_7, BCD_C);
    GPIO_write(PA_9, BCD_D);
}
```

Results

- Circuit Image



- Video

[Click to watch the result of Problem 1]

Discussion

D	C	B	A	a	b	c	d	e	f	g
0	0	0	0	0	0	0	0	0	0	1
0	0	0	1	1	0	0	1	1	1	1
0	0	1	0	0	0	1	0	0	1	0
0	0	1	1	0	0	0	0	1	1	0
0	1	0	0	1	0	0	1	1	0	0
0	1	0	1	0	1	0	0	1	0	0
0	1	1	0	1	1	0	0	0	0	0
0	1	1	1	0	0	0	1	1	1	1
1	0	0	0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	1	1	0	0

For the Coding for problem 1, I think it is very important that inputs A, B, C, D is toggled regularly. So that, I can use toggled in each iteration for up-counting. For the bit A, it is Least Significant Bit. So, it has to be toggled every iteration. For the bit B, it has to be toggled one time in every second iteration. So, for the N_{th} bit, start from 0, we can recognize that it has to be one toggling in every $2^N{}_{th}$ iteration.

Problem 2: Program BCD-7-segment decoder

- Instead of using the decoder chip, we are going to make the 7-segment decoder with the MCU programming.

Procedure

- num: 0 to 9 only (unsigned)
- Create a code that increases the displayed number from 0 to 9 with each button press.
- After the number '9', it should start from '0' again.

Configuration

Digital In for Button (B1)	Digital Out for 7-Segment
Digital In	Digital Out
PC13	PA5, PA6, PA7, PB6, PC7, PA9, PA8 ('a'~'g', respectively)
PULL-UP	Push-Pull, No Pull-up-Pull-down, Medium Speed

Code

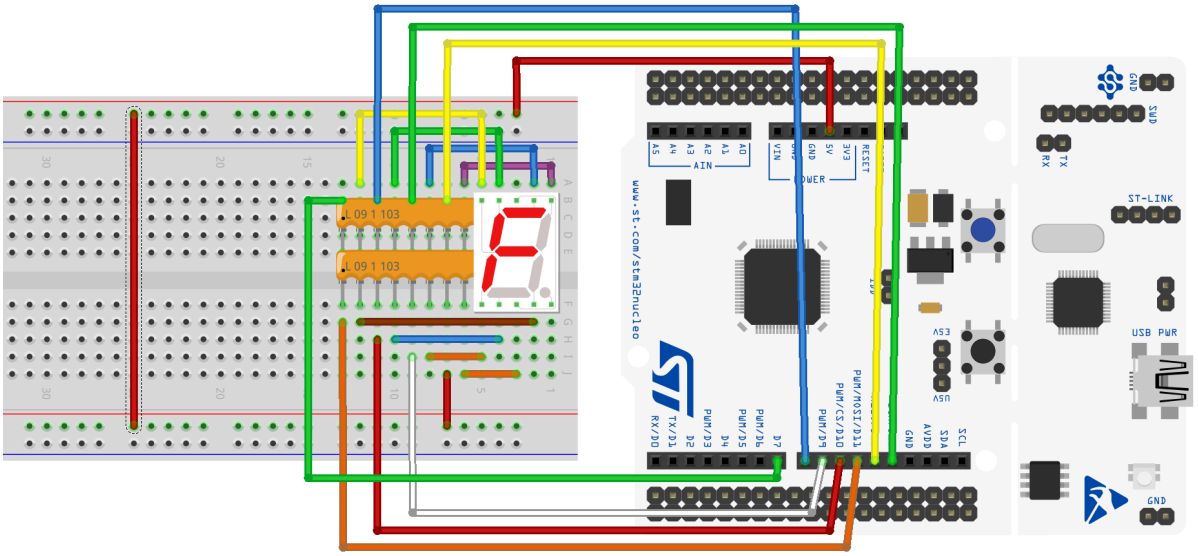
Problem 2 Code link in Github

```
#include "stm32f4xx.h"
#include "ecGPIO2.h"
#include "ecRCC2.h"

void setup(void);

int main(void) {
    // Initialiization --------------------------------------------------------
    setup();
    unsigned int cnt = 0;

    // Inifinite Loop ---------------------------------------------------------
    while(1){
        sevensegment_decoder(cnt %10);                      // Give the argument
"cnt%10" to the fucntion
        if(GPIO_read(BUTTON_PIN) == 0) cnt++;
        if (cnt > 9) cnt = 0;
        for(int i = 0; i < 500000;i++){}                        // delay_ms(500);
    }
}

// Initialization
void setup(void){
    RCC_HSI_init();
```

```
    GPIO_init(BUTTON_PIN, INPUT);
// calls RCC_GPIOC_enable()
    sevensegment_decoder_init();
// Call for initialization
}


// Initialization for 7-segment_decoder function
void sevensegment_decoder_init(void){
    GPIO_init(PA_5, OUTPUT);        // calls RCC_GPIOA_enable() / OUTPUT
    GPIO_ospeed(PA_5, MSPEED);      // Medium-Speed
    GPIO_pupd(PA_5, EC_NP);             // No Pull up, Pull down
    GPIO_otype(PA_5, PSPL);             // Push-Pull

    GPIO_init(PA_6, OUTPUT);        // calls RCC_GPIOA_enable() / OUTPUT
    GPIO_ospeed(PA_6, MSPEED);      // Medium-Speed
    GPIO_pupd(PA_6, EC_NP);             // No Pull up, Pull down
    GPIO_otype(PA_6, PSPL);             // Push-Pull

    GPIO_init(PA_7, OUTPUT);        // calls RCC_GPIOA_enable() / OUTPUT
    GPIO_ospeed(PA_7, MSPEED);      // Medium-Speed
    GPIO_pupd(PA_7, EC_NP);             // No Pull up, Pull down
    GPIO_otype(PA_7, PSPL);             // Push-Pull

    GPIO_init(PB_6, OUTPUT);        // calls RCC_GPIOB_enable() / OUTPUT
    GPIO_ospeed(PB_6, MSPEED);      // Medium-Speed
    GPIO_pupd(PB_6, EC_NP);             // No Pull up, Pull down
    GPIO_otype(PB_6, PSPL);             // Push-Pull

    GPIO_init(PC_7, OUTPUT);        // calls RCC_GPIOC_enable() / OUTPUT
    GPIO_ospeed(PC_7, MSPEED);      // Medium-Speed
    GPIO_pupd(PC_7, EC_NP);             // No Pull up, Pull down
    GPIO_otype(PC_7, PSPL);             // Push-Pull

    GPIO_init(PA_9, OUTPUT);        // calls RCC_GPIOA_enable() / OUTPUT
    GPIO_ospeed(PA_9, MSPEED);      // Medium-Speed
    GPIO_pupd(PA_9, EC_NP);             // No Pull up, Pull down
    GPIO_otype(PA_9, PSPL);             // Push-Pull

    GPIO_init(PA_8, OUTPUT);        // calls RCC_GPIOA_enable() / OUTPUT
    GPIO_ospeed(PA_8, MSPEED);      // Medium-Speed
    GPIO_pupd(PA_8, EC_NP);             // No Pull up, Pull down
    GPIO_otype(PA_8, PSPL);             // Push-Pull
    /* We are not gonna use dp for point
    GPIO_init(PB_10, OUTPUT);           // calls RCC_GPIOB_enable() / OUTPUT
    GPIO_ospeed(PB_10, MSPEED);     // Medium-Speed
    GPIO_pupd(PB_10, EC_NP);                // No Pull up, Pull down
    GPIO_otype(PB_10, PSPL);                // Push-Pull
    */
}

// Displaying "0"-"9" in 7-segment display by Code that play role as a decoder
void sevensegment_decoder(uint8_t num){
    // initialization
    // a~g for 7-seg Display
    int dispA = 1;
```

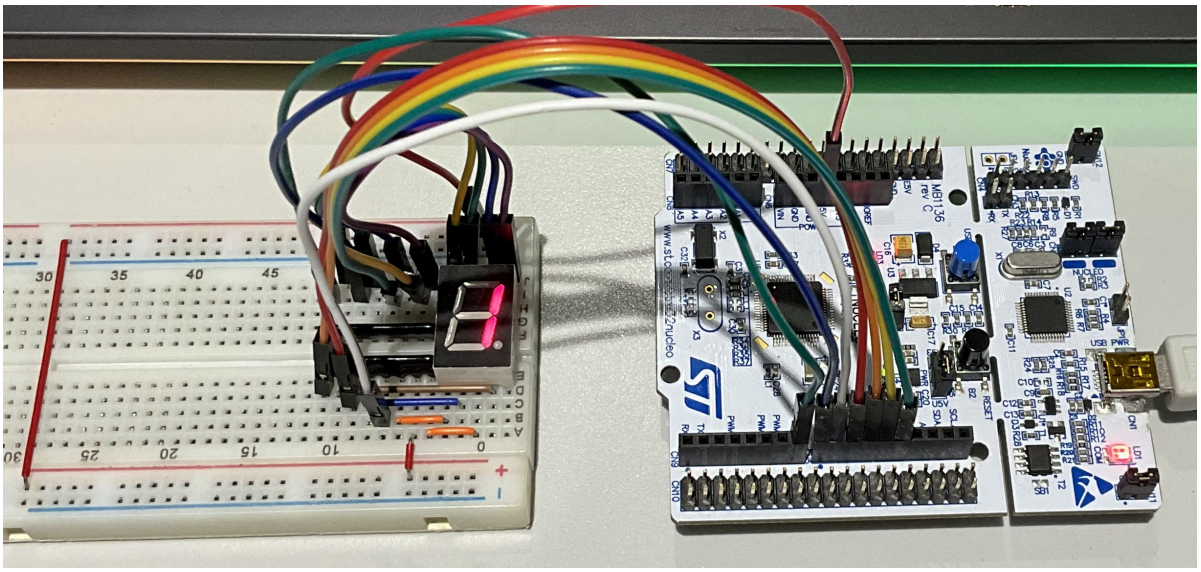
```
    int dispB = 1;
    int dispC = 1;
    int dispD = 1;
    int dispE = 1;
    int dispF = 1;
    int dispG = 1;
    int Goto[] = {63, 6, 91, 79, 102, 109, 124, 7, 127, 103};
// Navigation for Displaying Number of 7-segment display.

    for(int i=1;i<=Goto[num];i++){
        // Up counting in all possible 7-Segment displaying
                                dispA = !dispA;                 // Least
Significant Bit (LSB)
        if(i%2==0)  dispB = !dispB;                 // Bit 1
        if(i%4==0)  dispC = !dispC;                 // Bit 2
        if(i%8==0)  dispD = !dispD;                 // Bit 3
        if(i%16==0) dispE = !dispE;                 // Bit 4
        if(i%32==0) dispF = !dispF;                 // Bit 5
        if(i%64==0) dispG = !dispG;                 // Most Significant Bit (MSB)
    }

    GPIO_write(PA_5,dispA);
    GPIO_write(PA_6,dispB);
    GPIO_write(PA_7,dispC);
    GPIO_write(PB_6,dispD);
    GPIO_write(PC_7,dispE);
    GPIO_write(PA_9,dispF);
    GPIO_write(PA_8,dispG);
}
```

Circuit Diagram



Results

- Circuit Image



- Video

[Click to watch the result of Problem 2]

Discussion

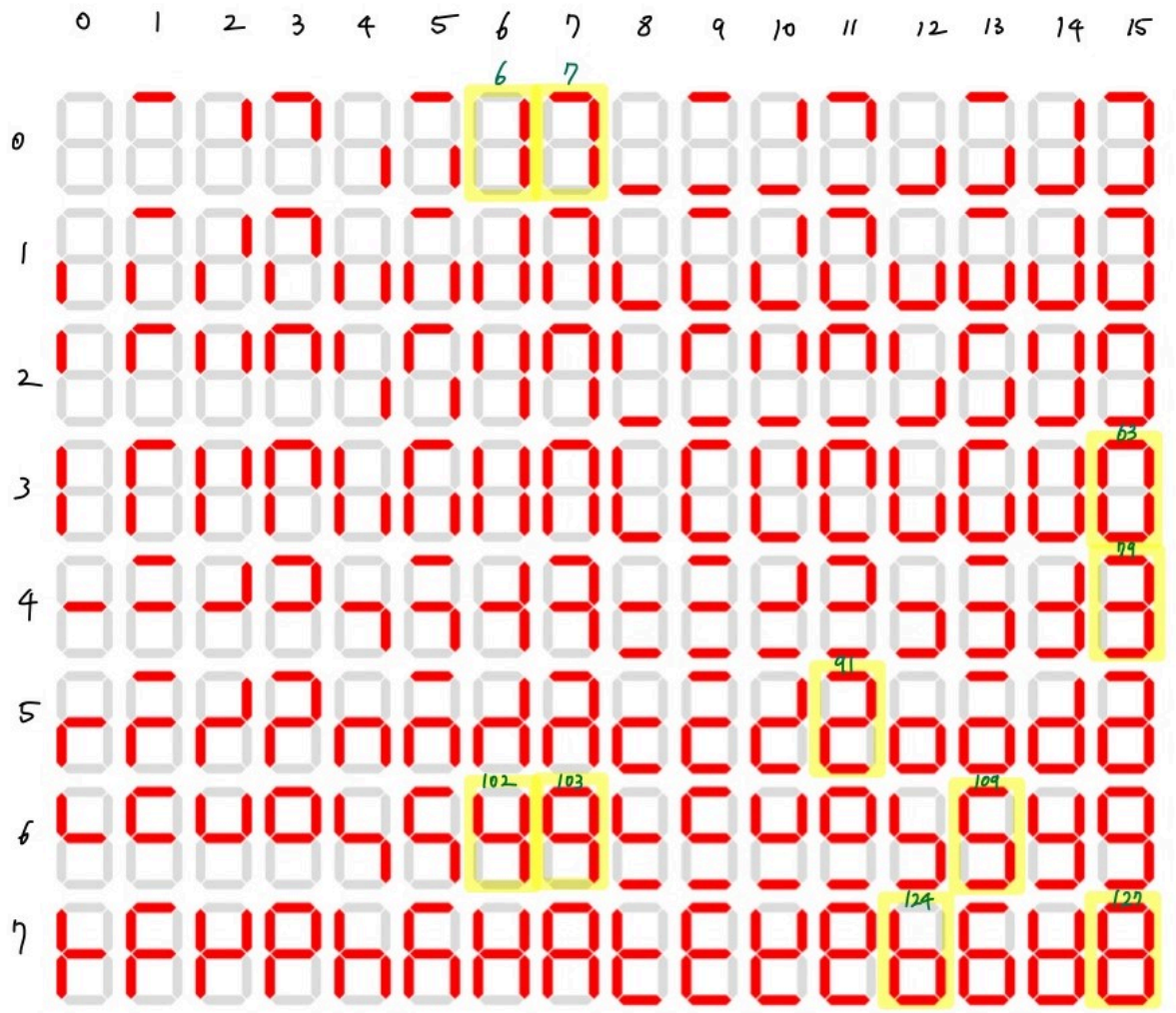
- Advantage of Array in the LAB

```
switch(num){
    case 0: Goto=63;    break;
    case 1: Goto=6;     break;
    case 2: Goto=91;    break;
    case 3: Goto=79;    break;
    case 4: Goto=102;   break;
    case 5: Goto=109;   break;
    case 6: Goto=124;   break;
    case 7: Goto=7;     break;
    case 8: Goto=127;   break;
    case 9: Goto=103;   break;
    default: Goto=63;   break;
```

```
int Goto[] = {63, 6, 91, 79, 102, 109, 124, 7, 127, 103};
```

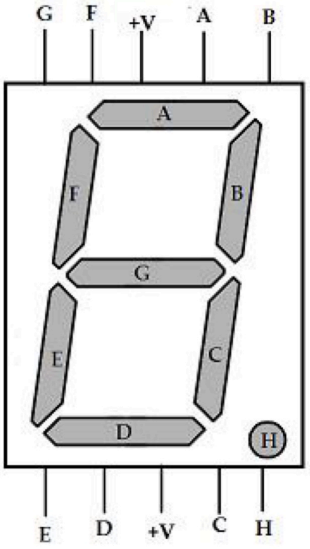
For the code we can use the switch function to select the numbers' navigation but, it is counting. It means there is a pattern in order. So, It dosen't need to use switch but array. we can just take the each value by loop.

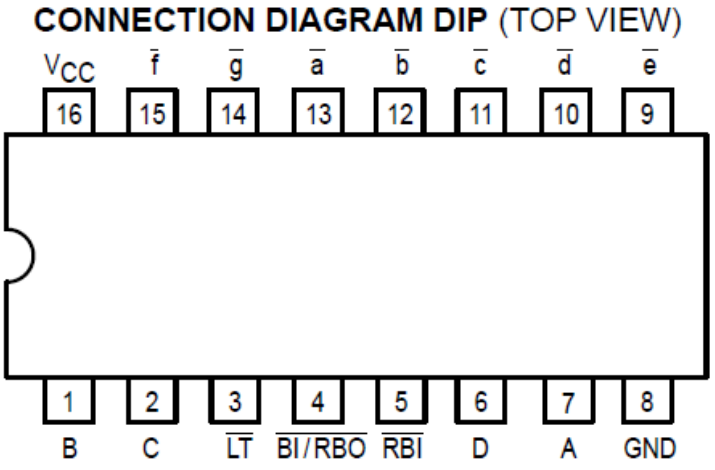
- How to display number by simply method?



Since there are 7 combinations of all displays that 7-seg can make from a to g, 2^7=128, that is, all 128 combinations can be created. In it, seven binary bits are shown on LED as they increase step by step in order from a to g. We treat the location of the number as if it were an address value and up-count it by repeating the code as much as the number up to that location. Then you can display the desired number. If you want to calculate the location of the desired number, if the bits from a to g are treated as one decimal number, that value will be the location value of the desired number. This way, you can find the location of the desired alphabet and print it right away.

Appendix





Reference

Complete list of all references used (github, blog, paper, etc)

```
https://en.wikipedia.org/wiki/Seven-segment_display
https://www.alldatasheet.com/datasheet-pdf/download/5724/MOTOROLA/SN74LS47N.html
https://ykkim.gitbook.io/ec
```