



King Mongkut's University of Technology Thonburi

Department of Electronics and Telecommunication Engineering Faculty of Engineering
EIE/ENE 335 Digital Circuit and Microprocessor Lab for the 3rd year student

Experiment: introduction to Nu_LB-002 (Cortex™-M0)

Objectives

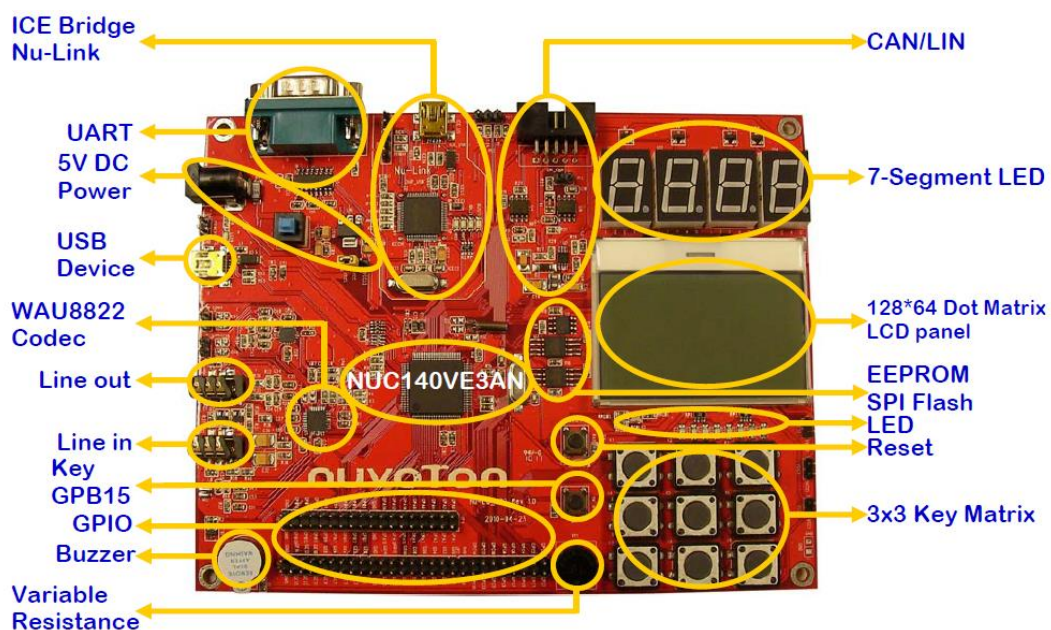
- To be familiar with Nu_LB-002 (Nuvoton learning board)
- Introduction to Keil
- How to use the NuMicro™ NUC100 series driver to do the fast application software development
- How to use GPIO/LCD

Background Theory

Nu_LB-002

The NUC100 series is 32-bit microcontrollers with embedded ARM® Cortex™-M0 core for industrial control at a cost equivalent to a traditional 8-bit microcontroller.

The NUC1XX series with Cortex™-M0 core runs up to 50MHz, up to 32K/64K/128K-byte embedded flash, and 4K/8K/16K-byte embedded SRAM, it also integrates Timers, Watchdog Timer, RTC, PDMA, UART, SPI/SSP, I2C, PWM Timer, GPIO, LIN, CAN, USB 2.0 FS Device, 12-bit ADC, Analog Comparator, Low Voltage Detector and Brown-out detector.



Equipment required

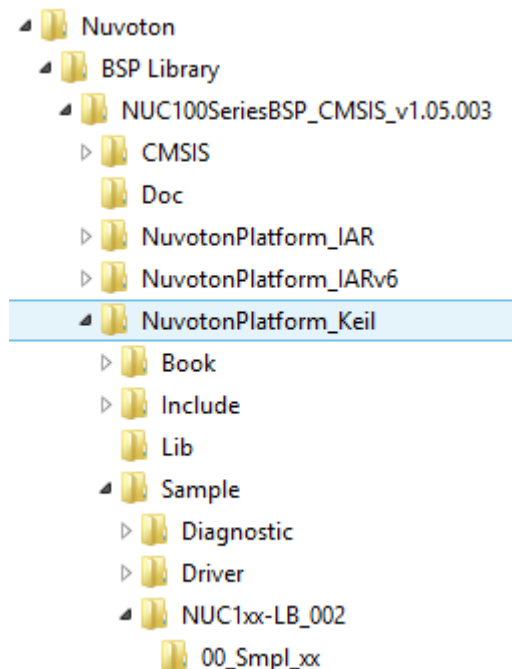
- Nu_LB-002 (Nuvoton learning board)

Reference:

1. [Nu_LB-002 Rev 2.1 User's Manual](#)
2. [NuMicro™ NUC130_140 Technical Reference Manual EN V2.02](#)
3. [NuMicro™ NUC100 Series Driver Reference Guide V1.05.002](#)

Procedure 1: the first example project

1. After installation of the software according to the procedure on the web-page (<http://webstaff.kmutt.ac.th/~dejwoot.kha/CalenderYear/Year2556/ene335/index.html> : [HowTO install Nu tools](#)), there are subdirectories under drive C:

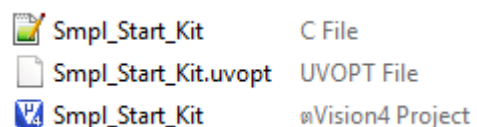


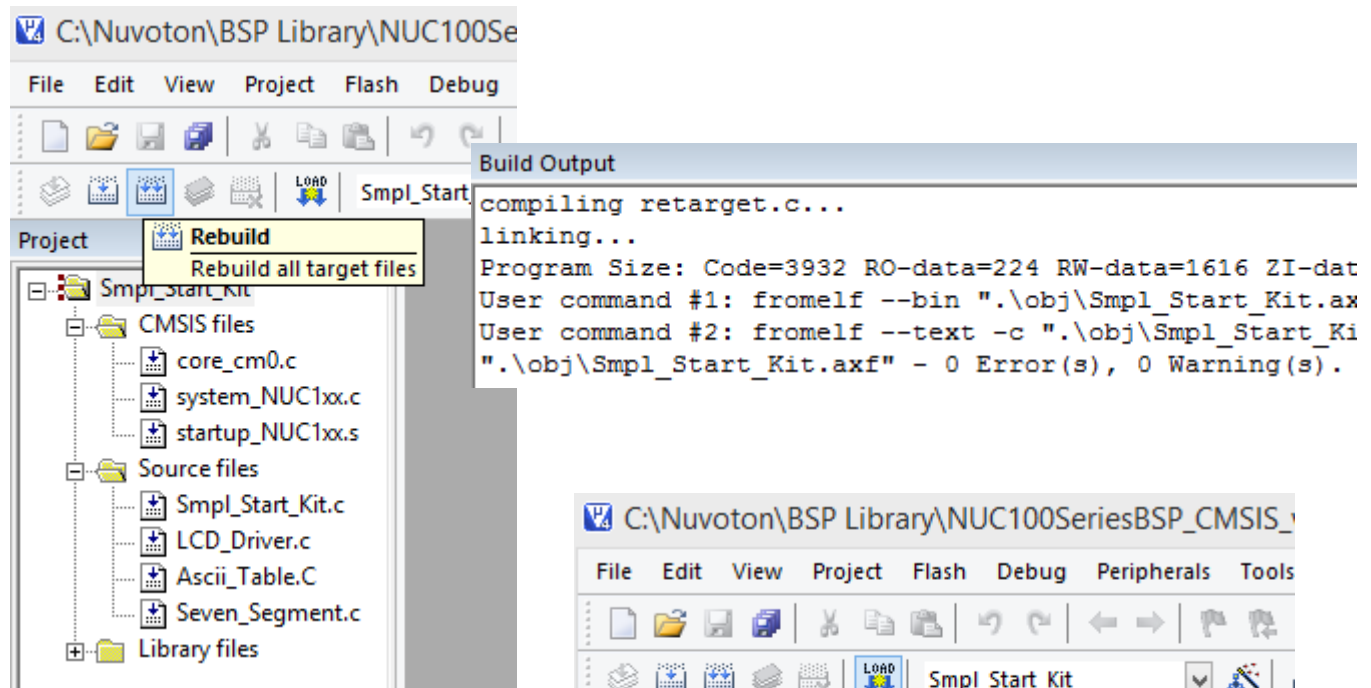
2. Download the zip file ([Smpl_Start_Kit.zip](#)) and save in the directory ("C:\Nuvoton\BSP Library\NUC100SeriesBSP_CMSIS_v1.05.003\NuvotonPlatform_Keil\Sample\NUC1xx-LB_002").
3. Unzip the file into the same directory. And rename the new directory to your preference, so you can work on your own.
(Smpl_Start_Kit_ENE335_secXX)

4. Open the project file by going to the new subdirectory that contain 3 files. Double click Smpl_Start_Kit.uvproj (Vision4 Project).

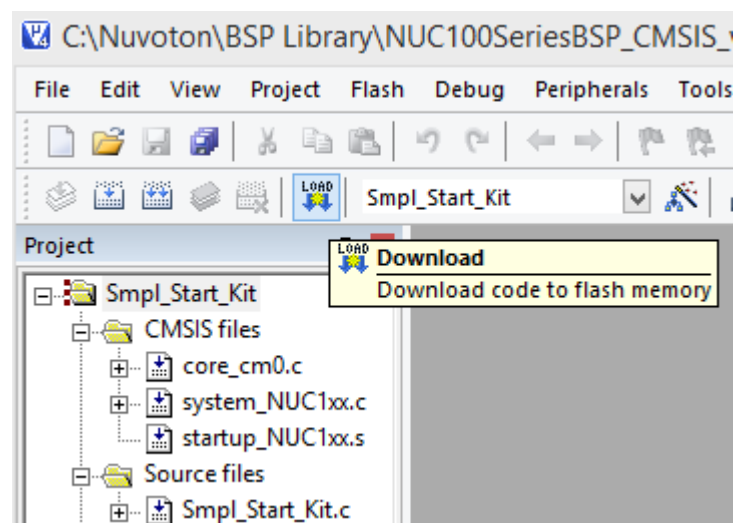
5. The Keil window will show up.

6. Compile the project to create the executable file by click at '**Rebuild**' icon.

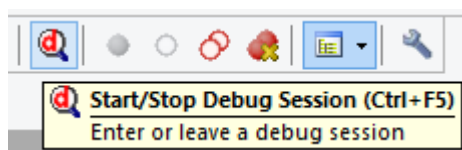




7. Make sure that there is no error after compilation is done. Connect the **ICE Bridge Nu-Link** to the **PC** using **USB cable**. Download the program to the Nu_LB-002 by clicking at the '**LOAD**' icon.



8. To run the program, reset the Nu_LB-002. (Use the reset button on the board)
9. Or download and debug the program by clicking at the



'Start/Stop Debug' icon.

10. Look at the 'Smpl_Start_Kit.c' and answer the following questions.

```

8  #include <stdio.h>
9  #include "NUC1xx.h"
10 #include "Driver\DrvSYS.h"
11 #include "Driver\DrvGPIO.h"
12 #include "LCD_Driver.h"
13 #include "Seven_Segment.h"
14
15 void delay_loop(void) {
16     uint32_t i, j;
17     for (i=0;i<3;i++) {
18         for (j=0;j<60000;j++);
19     }
20 }
21
22 /*-----
23    Interrupt subroutine
24 -----*/
25 static unsigned char count = 0;
26 static unsigned char loop = 12;
27
28 void TMR0_IRQHandler(void) { // Timer0 interrupt subroutine
29     unsigned char i = 0;
30     TIMER0->TISR.TIF = 1;
31     count++;
32     if (count == 5) {
33         DrvGPIO_ClrBit(E_GPC,loop);
34         loop++;
35         count=0;
36         if (loop == 17) {
37             for(i=12;i<16;i++) {
38                 DrvGPIO_SetBit(E_GPC,i);
39             }
40             loop = 12;
41         }
42     }
43 }
44
45 void Timer_initial(void) {
46     /* Step 1. Enable and Select Timer clock source */
47     SYSCLK->CLKSEL1.TMR0_S = 0; // Select 12Mhz for Timer0 clock source
48     SYSCLK->APBCLK.TMR0_EN = 1; // Enable Timer0 clock source
49
50     /* Step 2. Select Operation mode */
51     TIMER0->TCSR.MODE = 1; // Select periodic mode for operation mode
52
53     /* Step 3. Select Time out period
54     = (Period of timer clock input) * (8-bit Prescale + 1) * (24-bit TCMP) */
55     TIMER0->TCSR.PRESCALE = 0; // Set Prescale [0~255]
56
57     /* Step 4. Enable interrupt */
58     TIMER0->TCSR.IE = 1;
59     TIMER0->TISR.TIF = 1; // Write 1 to clear for safty
60     NVIC_EnableIRQ(TMR0_IRQn); // Enable Timer0 Interrupt
61
62     /* Step 5. Enable Timer module */
63     TIMER0->TCSR.CRST = 1; // Reset up counter
64     TIMER0->TCSR.CEN = 1; // Enable Timer0
65
66     TIMER0->TCSR.TDR_EN = 1; // Enable TDR function
67 }
68

```

```

69 int main(void) {
70     int i = 0, j = 0;
71     /* Unlock the protected registers */
72     UNLOCKREG();
73     /* Enable the 12MHz oscillator oscillation */
74     DrvSYS_SetOscCtrl(E_SYS_XTL12M, 1);
75     /* Waiting for 12M Xtal stable */
76     SysTimerDelay(5000);
77     /* HCLK clock source. 0: external 12MHz; 4:internal 22MHz RC oscillator */
78     DrvSYS_SelectHCLKSource(0);
79     /*lock the protected registers */
80     LOCKREG();
81
82     DrvSYS_SetClockDivider(E_SYS_HCLK_DIV, 0); /* HCLK clock frequency
83     = HCLK clock source / (HCLK_N + 1) */
84
85     for (i=12;i<16;i++) {
86         DrvGPIO_Open(E_GPC, i, E_IO_OUTPUT);
87     }
88
89     Initial_pannel(); //call initial pannel function
90     clr_all_pannal();
91
92     print_lcd(0, "Welcome! Nuvoton");
93     print_lcd(1, "This is LB test ");
94     print_lcd(2, "Nu_LB-002      ");
95     print_lcd(3, "1234567890123456");
96     Timer_initial();
97
98     while(1) {
99         for(i=0;i<9;i++) {
100             for(j=0;j<4;j++) {
101                 close_seven_segment();
102                 show_seven_segment(j,i);
103                 delay_loop();
104             }
105         }
106     }
107 }
108

```

Questions (the first example project)

1. What does the function on line 33 (DrvGPIO_ClrBit(E_GPC,loop);) do? When loop = 15. (Try set Breakpoint on line 33, then 'Run' and 'Step over the current line' in Debug mode)
2. What does the function on line 38 (DrvGPIO_SetBit(E_GPC,i);) do? When i = 14.

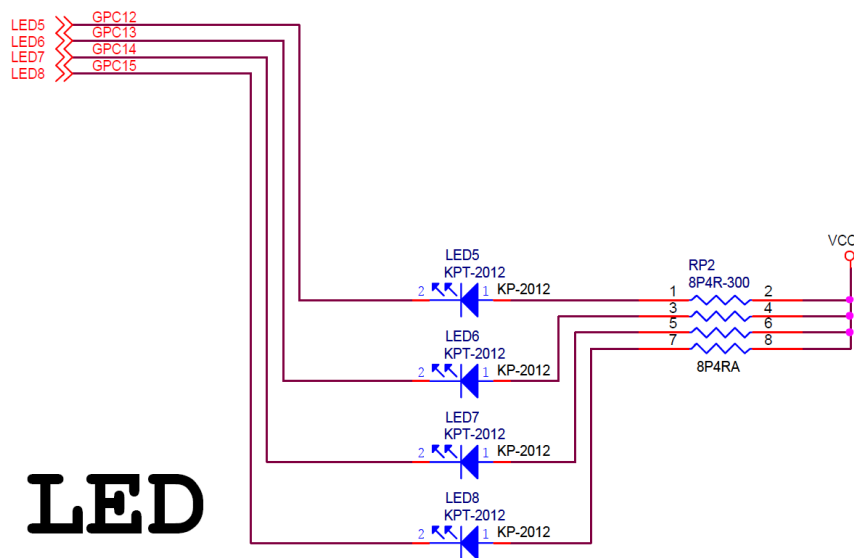
3. What does the function on line 102 (`show_seven_segment(j,i);`) do? When $j = 1$ and $i = 3$.
4. What is the time out period of Timer0? (Show how to get the result)
5. How long does the LED (GPC12) turn 'ON'? (Show how to get the result)

General Purpose I/O (GPIO)

NuMicro™ NUC130/NUC140 has up to 80 General Purpose I/O pins can be shared with other function pins; it depends on the chip configuration. These 80 pins are arranged in 5 ports named with GPIOA, GPIOB, GPIOC, GPIOD and GPIOE. Each port equips maximum 16 pins. Each one of the 80 pins is independent and has the corresponding register bits to control the pin mode function and data.

The I/O type of each of I/O pins can be configured by software individually as input, output, and open-drain or quasi-bidirectional mode. After reset, the I/O type of all pins stay in quasi-bidirectional mode and port data register GPIOx_DOUT[15:0] resets to 0x0000_FFFF. Each I/O pin equips a very weakly individual pull-up resistor which is about 110 K Ω ~300 K Ω for VDD is from 5.0 V to 2.5 V.

LED: GPC8-GPC15 to control LED function.



Procedure 2: LED: GPC8-GPC15 to control LED function.

1. Replace the content of the 'Smpl_Start_Kit.c' with the '[LED](#)' lab file.
2. Compile the project, and run the program.
3. Study the program and answer the following questions.


```

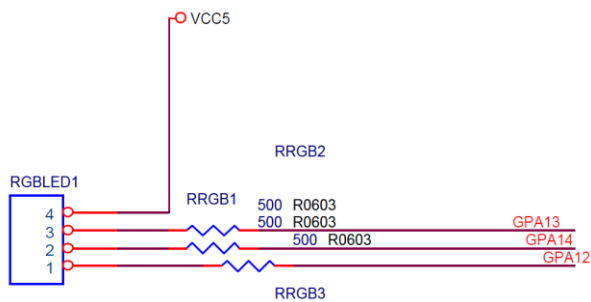
1 // Nu_LB-002: L00_01_LED_GPC_12_15.c
2 // use function, macro and directly at Register to turn 'ON' & 'OFF' LED
3
4 //
5 // Smp1_GPIO_LED : GPC12 ~ 15 to control on-board LEDs
6 //               output low to enable red LEDs
7 //
8 /*-----*/
9 /*
10 /* Copyright (c) Nuvoton Technology Corp. All rights reserved.
11 /* edited: june 2014: dejwoot.kha@mail.kmutt.ac.th
12 /*-----*/
13
14 #include <stdio.h>
15 #include "NUC1xx.h"
16 #include "Driver\DrvGPIO.h"
17 #include "Driver\DrvSYS.h"
18
19 #define DELAY300ms 300000 // The maximal delay time is 335000 us.
20
21 // Initial GPIO pins (GPC 12,13,14,15) to Output mode
22 void Init_LED() {
23     // initialize GPIO pins
24     DrvGPIO_Open(E_GPC, 12, E_IO_OUTPUT); // GPC12 pin set to output mode
25     DrvGPIO_Open(E_GPC, 13, E_IO_OUTPUT); // GPC13 pin set to output mode
26     DrvGPIO_Open(E_GPC, 14, E_IO_OUTPUT); // GPC14 pin set to output mode
27     DrvGPIO_Open(E_GPC, 15, E_IO_OUTPUT); // GPC15 pin set to output mode
28     // set GPIO pins to output Low
29     DrvGPIO_SetBit(E_GPC, 12); // GPC12 pin output Hi to turn off LED
30     DrvGPIO_SetBit(E_GPC, 13); // GPC13 pin output Hi to turn off LED
31     DrvGPIO_SetBit(E_GPC, 14); // GPC14 pin output Hi to turn off LED
32     DrvGPIO_SetBit(E_GPC, 15); // GPC15 pin output Hi to turn off LED
33 }
34
35 int main (void) {
36
37     Init_LED(); // Initialize LEDs (four on-board LEDs below LCD panel)
38
39     while (1) { // forever loop to keep flashing four LEDs one at a time
40         DrvGPIO_ClrBit(E_GPC, 12); // Function -> output Low to turn on LED
41         DrvSYS_Delay(DELAY300ms); // delay (The maximal delay time is 335000 us
42         DrvGPIO_SetBit(E_GPC, 12); // output Hi to turn off LED
43         DrvSYS_Delay(DELAY300ms); // delay
44
45         GPC_13 = 0; // Macro -> Turn 'On' LED
46         DrvSYS_Delay(DELAY300ms); // delay
47         GPC_13 = 1; // Macro -> Turn 'OFF' LED
48         DrvSYS_Delay(DELAY300ms); // delay
49
50         _DRVGPIO_DOUT (E_GPC, 14) = 0; // Macro -> Turn 'On' LED
51         DrvSYS_Delay(DELAY300ms); // delay
52         _DRVGPIO_DOUT (E_GPC, 14) = 1; // Macro -> Turn 'OFF' LED
53         DrvSYS_Delay(DELAY300ms); // delay
54
55         //DrvGPIO_ClrBit(E_GPC, 15); // output Low to turn on LED
56         GPIOC->DOUT &= 0x7FFF; // turn on only LED GPC 15
57         DrvSYS_Delay(DELAY300ms); // delay
58         //DrvGPIO_SetBit(E_GPC, 15); // output Hi to turn off LED
59         GPIOC->DOUT |= 0x8000; // turn off only LED GPC 15
60         DrvSYS_Delay(DELAY300ms); // delay
61     }
62 }
63

```


Questions (LED)

1. What is the difference between Function on line 40 and Macro on line 45?
2. From previous question, which one is execute faster and why?
3. What is the difference between Macro on line 45 and Macro on line 50?
4. From previous question, which one is execute faster and why?

RGBLED: GPA12-GPA13 to control RGB LED function



Procedure 3: RGBLED: GPA12-GPA13 to control RGB LED function

1. Replace the content of the 'Smpl_Start_Kit.c' with the 'RGB-LED' lab file.
2. Compile the project, and run the program.
3. Study the program and answer the following questions.

```

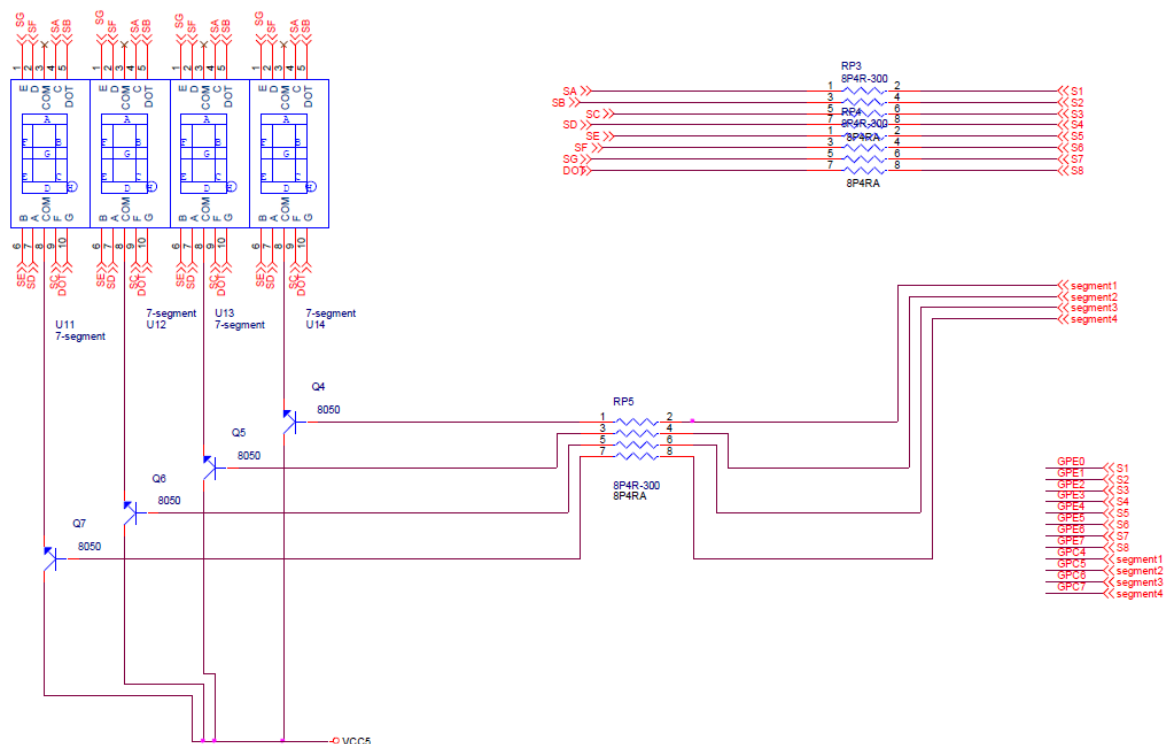
11 #include <stdio.h>
12 #include "NUC1xx.h"
13 #include "Driver\DrvSYS.h"
14 #include "Driver\DrvGPIO.h"
15
16 #define DELAY300ms 300000 // The maximal delay time is 335000 us.
17
18 // Initial GPIO pins (GPA 12,13,14) to Output mode
19 void Init_RGB_LED() {
20     // initialize GPIO pins
21     DrvGPIO_Open(E_GPA, 12, E_IO_OUTPUT); // GPA12 pin set to output mode
22     DrvGPIO_Open(E_GPA, 13, E_IO_OUTPUT); // GPA13 pin set to output mode
23     DrvGPIO_Open(E_GPA, 14, E_IO_OUTPUT); // GPA14 pin set to output mode
24     // set GPIO pins output Hi to disable LEDs
25     DrvGPIO_SetBit(E_GPA, 12); // GPA12 pin output Hi to turn off Blue LED
26     DrvGPIO_SetBit(E_GPA, 13); // GPA13 pin output Hi to turn off Green LED
27     DrvGPIO_SetBit(E_GPA, 14); // GPA14 pin output Hi to turn off Red LED
28 }
29
30 int main(void) {
31     Init_RGB_LED();
32
33     // GPA12 = Blue, 0 : on, 1 : off
34     // GPA13 = Green, 0 : on, 1 : off
35     // GPA14 = Red, 0 : on, 1 : off
36     while(1) {
37         // set RGBled to Blue
38         DrvGPIO_ClrBit(E_GPA,12); // GPA12 = Blue, 0 : on, 1 : off
39         DrvGPIO_SetBit(E_GPA,13);
40         DrvGPIO_SetBit(E_GPA,14);
41         DrvSYS_Delay(DELAY300ms); // delay
42
43         // set RGBled to Green
44         DrvGPIO_SetBit(E_GPA,12);
45         DrvGPIO_ClrBit(E_GPA,13); // GPA13 = Green, 0 : on, 1 : off
46         DrvGPIO_SetBit(E_GPA,14);
47         DrvSYS_Delay(DELAY300ms); // delay
48
49         // set RGBled to Red
50         DrvGPIO_SetBit(E_GPA,12);
51         DrvGPIO_SetBit(E_GPA,13);
52         DrvGPIO_ClrBit(E_GPA,14); // GPA14 = Red, 0 : on, 1 : off
53         DrvSYS_Delay(DELAY300ms); // delay
54
55         // set RGBled to off
56         DrvGPIO_SetBit(E_GPA,12); // GPA12 = Blue, 0 : on, 1 : off
57         DrvGPIO_SetBit(E_GPA,13); // GPA13 = Green, 0 : on, 1 : off
58         DrvGPIO_SetBit(E_GPA,14); // GPA14 = Red, 0 : on, 1 : off
59         DrvSYS_Delay(DELAY300ms); // delay
60     }
61 }

```

Questions (RGBLED)

1. What is an alternate function for the GPA12
2. What Macros functions can we replace the function on line 38? (2 answers)

7-SEGMENT: GPE0-GPE7, GPC4-GPC7 control



Procedure 4: 7-SEGMENT: GPE0-GPE7, GPC4-GPC7 control

1. Replace the content of the 'Smpl_Start_Kit.c' with the '[7SEG](#)' lab file.
2. Compile the project, and run the program.
3. Study the program and answer the following question.

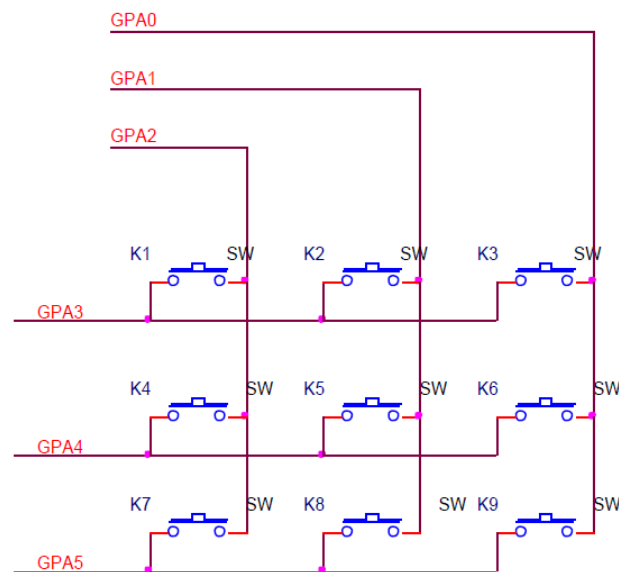
```

12 #include <stdio.h>
13 #include "NUC1xx.h"
14 #include "DrvSYS.h"
15 #include "Seven_Segment.h"
16
17 #define DELAY300ms 300000 // The maximal delay time is 335000 us.
18 #define scanDelay 4000
19
20 // display an integer on four 7-segment LEDs
21 void seg_display(int16_t value) {
22     int8_t digit;
23     digit = value / 1000;
24     close_seven_segment();
25     show_seven_segment(3,digit);
26     DrvSYS_Delay(scanDelay);
27
28     value = value - digit * 1000;
29     digit = value / 100;
30     close_seven_segment();
31     show_seven_segment(2,digit);
32     DrvSYS_Delay(scanDelay);
33
34     value = value - digit * 100;
35     digit = value / 10;
36     close_seven_segment();
37     show_seven_segment(1,digit);
38     DrvSYS_Delay(scanDelay);
39
40     value = value - digit * 10;
41     digit = value;
42     close_seven_segment();
43     show_seven_segment(0,digit);
44     DrvSYS_Delay(scanDelay);
45 }
46
47 int32_t main (void) {
48     int32_t i = 0;
49
50     while(1) {
51         seg_display(i/10); // display i on 7-segment display
52         i++; // increment i
53         if (i == 100000) i = 0;
54     }
55 }

```

Questions (7-SEGMENT)

1. What is the function to display the left most digit with '7'.

KEYBOARD: GPA0-GPA5 to control KEYBOARD function**Procedure 5: KEYBOARD:** GPA0-GPA5 to control KEYBOARD function

1. Replace the content of the 'Smpl_Start_Kit.c' with the '[KeyPad-7SEG](#)' lab file.
2. Compile the project, and run the program. (Add ScanKey.c to the project, from "C:\Nuvoton\BSP Library\NUC100SeriesBSP\CMSIS_v1.05.003\NuvotonPlatform_Keil\Src\NUC1xx-LB_002\ScanKey.c")
3. Study the program and answer the following question.

```

18 #include <stdio.h>
19 #include "NUC1xx.h"
20 #include "DrvSYS.h"
21 #include "Seven_Segment.h"
22 #include "scankey.h"
23
24 int32_t main (void) {
25     int8_t number;
26
27     OpenKeyPad();
28
29     while(1) {
30         number = Scankey();           // scan keypad to get a number (1~9)
31         show_seven_segment(0,number); // display number on 7-segment LEDs
32         DrvSYS_Delay(5000);           // delay time for keeping 7-segment display
33         close_seven_segment();        // turn off 7-segment LEDs
34     }
35 }

```

```

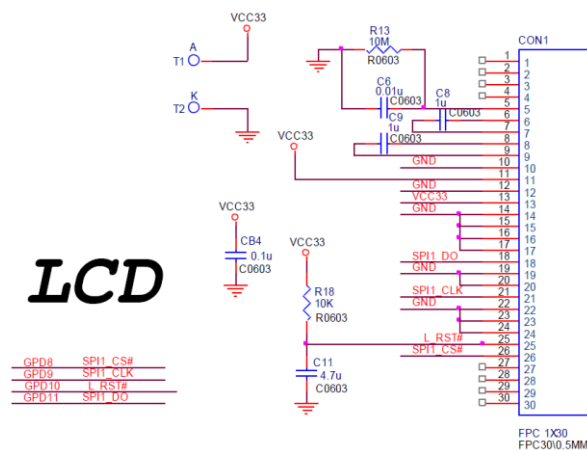
6  #include <stdio.h>
7  #include "Driver\DrvGPIO.h"
8  #include "ScanKey.h"
9
10 void delay(void)
11 {
12     int j;
13     for(j=0;j<1000;j++);
14 }
15
16 void OpenKeyPad(void)
17 {
18     uint8_t i;
19     /* Initial key pad */
20     for(i=0;i<6;i++)
21         DrvGPIO_Open(E_GPA, i, E_IO_QUASI);
22 }
23
24 void CloseKeyPad(void)
25 {
26     uint8_t i;
27
28     for(i=0;i<6;i++)
29         DrvGPIO_Close(E_GPA, i);
30 }
31
32 uint8_t Scankey(void)
33 {
34     uint8_t act[4]={0x3b, 0x3d, 0x3e};
35     uint8_t i,temp,pin;
36
37     for(i=0;i<3;i++)
38     {
39         temp=act[i];
40         for(pin=0;pin<6;pin++)
41         {
42             if((temp&0x01)==0x01)
43                 DrvGPIO_SetBit(E_GPA,pin);
44             else
45                 DrvGPIO_ClrBit(E_GPA,pin);
46             temp>>=1;
47         }
48         delay();
49         if(DrvGPIO_GetBit(E_GPA,3)==0)
50             return(i+1);
51         if(DrvGPIO_GetBit(E_GPA,4)==0)
52             return(i+4);
53         if(DrvGPIO_GetBit(E_GPA,5)==0)
54             return(i+7);
55     }
56     return 0;
57 }
58

```

Questions (KEYBOARD)

1. What is going to happen if the code on line 34 are changed from 0x3b, 0x3d, 0x3e to 0x3F, 0x3F, and 0x3d, respectively?

LCD: GPD8-GPD11 connect to LCD function.



Procedure 6: LCD: GPD8-GPD11 connect to LCD function.

1. Replace the content of the 'SmpI_Start_Kit.c' with the 'LCD' lab file.
2. Compile the project, and run the program.
3. Study the program and do the assignment in the class.

```

10 #include <stdio.h>
11 #include "NUC1xx.h"
12 #include "LCD_Driver.h"
13 #include "Driver\DrvGPIO.h"
14 #include "Driver\DrvUART.h"
15 #include "Driver\DrvSYS.h"
16
17 #define DELAY300ms 300000 // The maximal delay time is 335000 us.
18
19 int main (void) {
20     char lcd3_buffer[18];
21     uint32_t lcdDemoCounter = 0;
22
23     Initial_pannel(); //call initial pannel function
24     clr_all_pannel();
25
26     print_lcd(0, "LCD demo");
27     print_lcd(1, "12345678901234567890");
28     sprintf(lcd3_buffer, "count up = ");
29
30     while (1) {
31         sprintf(lcd3_buffer+11, "%d", lcdDemoCounter);
32         print_lcd(2, lcd3_buffer);
33         DrvSYS_Delay(DELAY300ms); // delay
34         lcdDemoCounter++;
35     }
36 }

```


Assignment(s)

Summarize what you suppose to learn in this class.