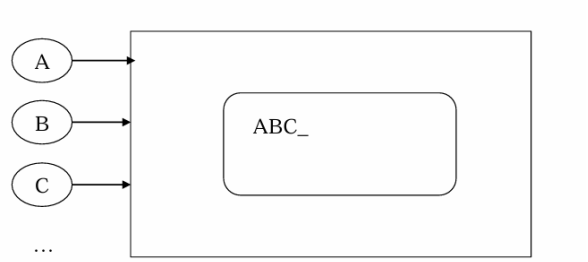
# Lab05 Final Report

B1029009 簡鈺晴

## ● Problem description

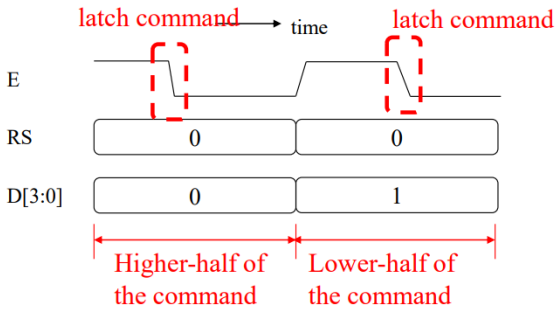
Display characters according to the button pressed



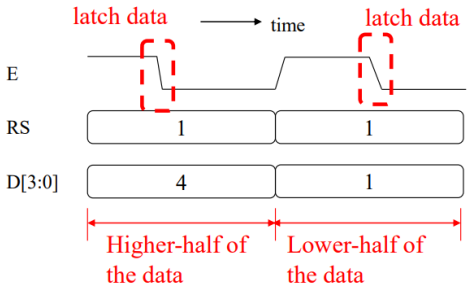

Basic: Display the character pressed at the cursor position.
Bonus 1: Implement the `new-line' key ◦ Change to the next line if a new-line button is pressed at Line 1 ◦ Scroll the screen if a new-line button is pressed at Line 2.
Bonus 2: Implement the arrow keys (up, down, left, right) ◦ Move the cursor by the arrow key ◦ Insert character at the cursor position.

## ● Code and explanations

```
void LCD_SendCommand (char cmd){
    LCD_Status_ClearRS ();

    ///send the higher half
    LCD_Status_SetWord ((cmd>>4) & 0x0f);
    LCD_Status_SetEnable ();
    P3 = LCD_status;
    LCD_Delay ();
    LCD_Status_ClearEnable ();
    P3 = LCD_status;
    LCD_Delay ();

    ///send the lower half
    LCD_Status_SetWord (cmd&0x0f);
    LCD_Status_SetEnable ();
    P3 = LCD_status;
    LCD_Delay ();
    LCD_Status_ClearEnable ();
    P3 = LCD_status;
    LCD_Delay ();
}
```



```
void LCD_SendData (char dat){
    LCD_Status_SetRS ();

    ///send the higher half
    LCD_Status_SetWord ((dat>>4) & 0x0f);
    LCD_Status_SetEnable ();
    P3 = LCD_status;
    LCD_Delay ();
    LCD_Status_ClearEnable ();
    P3 = LCD_status;
    LCD_Delay ();

    ///send the lower half
    LCD_Status_SetWord (dat&0x0f);
    LCD_Status_SetEnable ();
    P3 = LCD_status;
    LCD_Delay ();
    LCD_Status_ClearEnable ();
    P3 = LCD_status;
    LCD_Delay ();
}
```




This lab also need to send commend and send data to LCD, so we set 0 to commend/ set 1 to data and let D[7:0] separated two part – higher half and lower half. First is set higher half, we need to let data to shift right 4, prepare to status word. When E is enable, it will send out status word. The lower half is as same as below, but don't need to shift right.

```
void
LCD_ClearScreen ()
{
    LCD_SendCommand (0x01);
}

void Shutup_WatchDog (){
    WDTCN = 0xde;
    WDTCN = 0xad;
}//end of function Shutup_WatchDog
```

```
unsigned int delay_lcd=20000;

void LCD_Delay (){
    int i;
    for (i=0;i<delay_lcd;i++);
}
```

```
void LCD_PortConfig (){
    //initialize SFR setup page
    SFRPAGE = CONFIG_PAGE;                  // Switch to configuration page
    P2MDIN = 0xff;
    P2MDOUT = 0x00;
    P1MDOUT = 0xff;

    //setup the cross-bar and configure the I/O ports
    XBR2 = 0xc0;
    P3MDOUT = 0x3f;

    //set to normal mode
    SFRPAGE = LEGACY_PAGE;
}//end of function LCD_PortConfig ()
```

```
void
LCD_SendCommand (char cmd);

void LCD_Init (){
    LCD_SendCommand(0x02);          // Initialize as 4-bit mode
    LCD_SendCommand (0x28);       //Display function: 2 rows for 4-bit data, small
    LCD_SendCommand (0x0e);       //display and curson ON, curson blink off
    LCD_SendCommand (0x01);       //clear display, cursor to home
    //LCD_SendCommand (0x10);         //cursor shift left
    //LCD_SendCommand (0x06);         //cursor increment, shift off
}
```

And also have some basic setting we need to config like the P1 port.Next, the delay will be 20000 to check if LCD is clean. Then, there has initialize to LCD when we restart the code.

Because the basic will use at bonus1 and bonus2, so I explain the code of bonus1 and bouns2.

## Bonus1:

```
#include "C8051F040.h"
#include "LCD.h"

char LCD_status;
char array[16];
int i = 0;
int j = 0;
```

```
void main (){
    //int i;
    P1 = 0x00;
    Shutup_WatchDog ();
    LCD_PortConfig ();
    LCD_Init ();
    LCD_ClearScreen ();
    P1 = 0xff;
    for (j = 0; j < 16; j++){
        array[j] = ' ';
    }
```

First, we need to include some .h file and set array is one dimension. Next, we done some basic setting and initialize the array.

```
while (1){
    P2 = 0x00;
    if( i < 16){
        if(P2 == 128) {
            LCD_SendData ('A');
            array[i] = 'A';
            i++;
        } else if(P2 == 64){
            LCD_SendData ('B');
            array[i] = 'B';
            i++;
        } else if(P2 == 32){
            LCD_SendData ('C');
            array[i] = 'C';
            i++;
        } else if(P2 == 16){
            LCD_ClearScreen ();
            //LCD_PrintString (array);
            for (j = 0; j < i ; j++){
                LCD_SendData (array[j]);
            }
            LCD_SendCommand(0x00C0);
            i = 0;
        }
    } else {
        if(P2 == 16){
            LCD_ClearScreen ();
            //LCD_PrintString (array);
            for (j = 0; j < i ; j++){
                LCD_SendData (array[j]);
            }
            LCD_SendCommand(0x00C0);
            i = 0;
        }
    }
```

Then, we set P2 = 0. When we press the button P2.7 to P2.4, there has some function that we can do. First, P2 == 128 (P2.7) can print A on LCD. And P2 == 64 (P2.6) can print B. P2 == 32 (P2.5) can print C. Above character we use array to record it, so that we can use at change line. Next, P2 == 16 (P2.4) can change the line, so we need to clear the LCD screen and then print that the character that we record before, at last use commend 0xC0 to change the line to second line.

Here have some tips that the LCD will overflow when the user press the button exceed the screen display, so we have to detect whether input is bigger than 16 and if yes than cannot input any character again.

## Bonus 2:

```
#include "C8051F040.h"
#include "LCD.h"

char LCD_status;
char array[2][16];
int cur_x, cur_y;
int a = 0;
char k;
```

```
void main (){
    P1 = 0x00;
    Shutup_WatchDog ();
    LCD_PortConfig ();
    LCD_Init ();
    LCD_ClearScreen ();
    P1 = 0xff;
    cur_x = 0;
    cur_y = 0;
    for (a = 0; a < 16; a++){
        array[0][a] = ' ';
        array[1][a] = ' ';
    }
```

First, we need to include some .h file and set array is two dimension. Next, we done some basic setting and initialize the array.

```
while (1){
    P2 = 0x00;
    // Insert A
    if(P2 == 128) {
        insert_data('A');
        if (cur_x < 15) {
            cur_x++;
        }
        jump_cursor(cur_x, cur_y);
    // Insert B
    } else if(P2 == 64){
        insert_data('B');
        if (cur_x < 15) {
            cur_x++;
        }
        jump_cursor(cur_x, cur_y);
    // Insert C
    } else if(P2 == 32){
        insert_data('C');
        if (cur_x < 15) {
            cur_x++;
        }
        jump_cursor(cur_x, cur_y);
    // cursor left
    } else if (P2 == 8) {
        if (cur_x > 0) {
            LCD_SendCommand(0x0010);
            cur_x--;
        }
    // cursor right
    } else if (P2 == 4) {
        if (cur_x < 15) {
            LCD_SendCommand(0x0014);
            cur_x++;
        }
    // cursor up
    } else if (P2 == 2) {
        if (cur_y == 1) {
            cur_y = 0;
            jump_cursor(cur_x, cur_y);
        }
    // cursor down
    } else if (P2 == 1) {
        if (cur_y == 0) {
            cur_y = 1;
            jump_cursor(cur_x, cur_y);
        }
    }
```

Then, we set P2 = 0. When we press the button P2.7 to P2.4, there has some function that we can do. First, P2 == 128 (P2.7) can insert A on LCD. And P2 == 64 (P2.6) can insert B. P2 == 32 (P2.5) can insert C. Above character we will move cursor, so that we can know where cursor it is. Next, P2 == 8(P2.3) and P2 == 4(P2.2) are cursor shift left and shift right, we use commend 0x10 and 0x14 implement those function. Afterward, we implement cursor up and down, which detect current cursor y location to change the line to avoid the line will go to not first or second line. I will interpret the jump_cursor and insert_data below.

```
void jump_cursor(int x, int y) {
    if (y) {LCD_SendCommand(0x00C0);
    else {LCD_SendCommand(0x0002);}
    for (a = 0; a < x; a++) {
        LCD_SendCommand(0x0014);
    }
}
```

The function jump_cursor can locate the position of cursor and display on LCD.

```
void insert_data(char c) {
    LCD_SendData(c);
    for (a = cur_x; a < 15; a++) {
        k = array[cur_y][a+1];
        array[cur_y][a+1] = array[cur_y][cur_x];
        LCD_SendData(array[cur_y][a+1]);
        array[cur_y][cur_x] = k;
    }
    array[cur_y][cur_x] = c;
}
```

The function insert_data can insert the button that we push to display. Then we need to let the character behind the letter we press to move back.

So we use k to store the first character behind the letter, and store what the current cursor point to array[a+1] then print out after the insert character. At last, let the original position a+1 store to current can prepare to next character print at LCD cursor position+2. And when current characters are already move in array, the new insert letter will also store in array [current cursor].

- Difficulties you've encountered and your solutions

This time we have encountered a lot of problems.

1. Board Problem

   The C8051 board maybe is too old or otherwise, we have already meet a lot of time from lab3 until now. We always debug on our code, but at last is the board problem. Hope next time we don't need to change the board again.

2. Change line overflow

   When we try to demo, the assistance find that our code didn't restrict the range of the array, so when press over 16 time from 0, the character will disappear at LCD and after 24 times it will display in the second row, so we need to restrict the input range to solve this problem.

3. When insert data have something wrong.

   We meet some problem bonus 2. When we want to insert data at the cursor position, but LCD always cannot display correct characters behind insert data. After we think about the running logical and ask assistant help and had solved this problem.

- Discussions or what you have learned

This lab learn about how LCD use and also know about a lot of commend can use in LCD. I also learn more about C array with for loop using. Hope I can done more quickly when I encounter the similar question. On the other hand, the board wish no more failure again.