Ministry of Education of Moldova

Technical University of Moldova

Faculty "Computers, Informatics and Microelectronics"

REPORT

Topic: ” INTERFACING OF 4x4 KEYPAD ”

Performed by st. gr. FAF-141 : Botnari Nicolae

Verified by : Andrei Bragarenco

Chişinău 2016

INTERFACING OF 4x4 KEYPAD

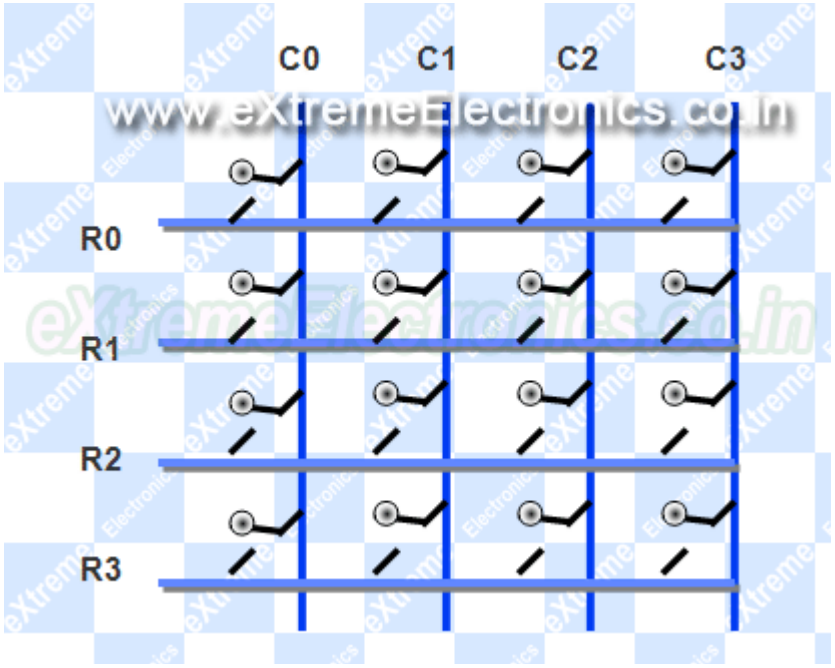
Theory

Many application requires large number of keys connected to a computing system. Example includes a PC keyboard, Cell Phone keypad and Calculators. If we connect a single key to MCU, we just connect it directly to i/o line. But we cannot connect, say 10 or 100 keys directly MCUs i/o. Because :-

- It will eat up precious i/o line.
- MCU to Keypad interface will contain lots of wires.

Buy Matrix Keypad

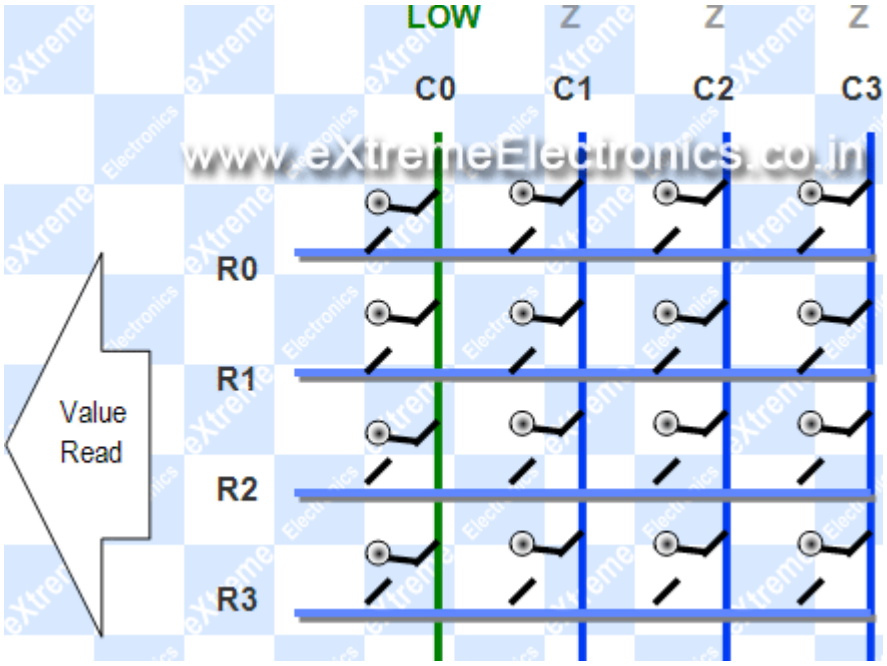
We want to avoid all these troubles so we use some clever technique. The technique is called multiplexed matrix keypad. In this technique keys are connected in a matrix (row/column) style as shown below.



Matrix Keypad Basic Connection

The rows R0 to R3 are connected to Input lines of Microcontroller. The i/o pins where they are connected are made Input. This is done by setting the proper DDR Register in AVR and TRIS Register in PIC. The column C0 to C3 are also connected to MCUs i/o line. These are kept at High Impedance State (AKA input), in high z state (z= impedance) state these pins are neither HIGH or LOW they are in TRISTATE. And in their PORT value we set them all as low, so as soon as we change their DDR bit to 1 they become output with value LOW.

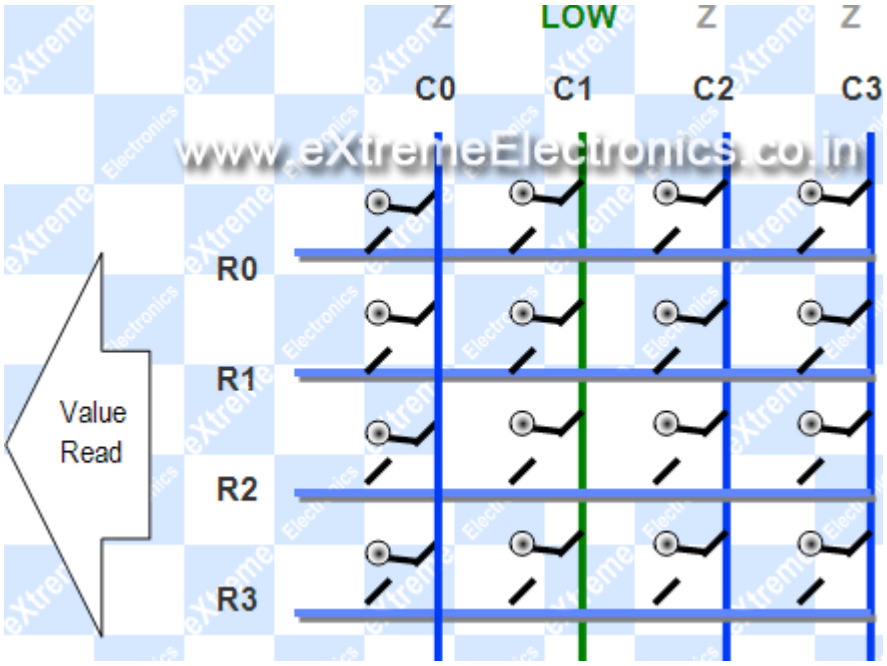
One by One we make each Column LOW (from high Z state) and read state of R0 to R3.



Column 0 Selected

As you can see in the image above C0 is made LOW while all other Columns are in HIGH Z State. We can read the Value of R0 to R3 to get their pressed status. If they are high the button is NOT pressed. As we have enabled internal pullups on them, these pullups keep their value high when they are floating (that means NOT connected to anything). But when a key is pressed it is connected to LOW line from the column thus making it LOW.

After that we make the C0 High Z again and make C1 LOW. And read R0 to R3 again. This gives us status of the second column of keys. Similarly we scan all columns.



Column 1 Selected

How to Do it All with AVRs

Each i/o port in AVR has three related registers PORTx, DDRx and PINx. For example port A has

- **PORTA** Port Driver – when any bit is set to 1 it appears as HIGH i.e. 5v . But this is the case only if that bit is OUTPUT. If it is input, setting any bit to 1 enables the internal pullup on that bit.
- **DDRA D**ATA **D**IRECTION **R**EGISTER – Make any pin on than port as IN or OUT. When bit is 1 it represents Output. When bit is 0 it represents Input. Input state is also called tristate or high Z state.
- **PINA** – Read it to get the level (HIGH or LOW) at the actual i/o pin. It is read when the pin is made input.

So now you know

- How to make any i/o line Input(high Z) or Output.
- How to enable internal pullup register on input lines.
- How to read value that is present on input lines.

Solution:

Interfacing of microcontroller to 4x4 Keypad is very simple.

The Simple logic behind this is just scan all the key in the matrix.

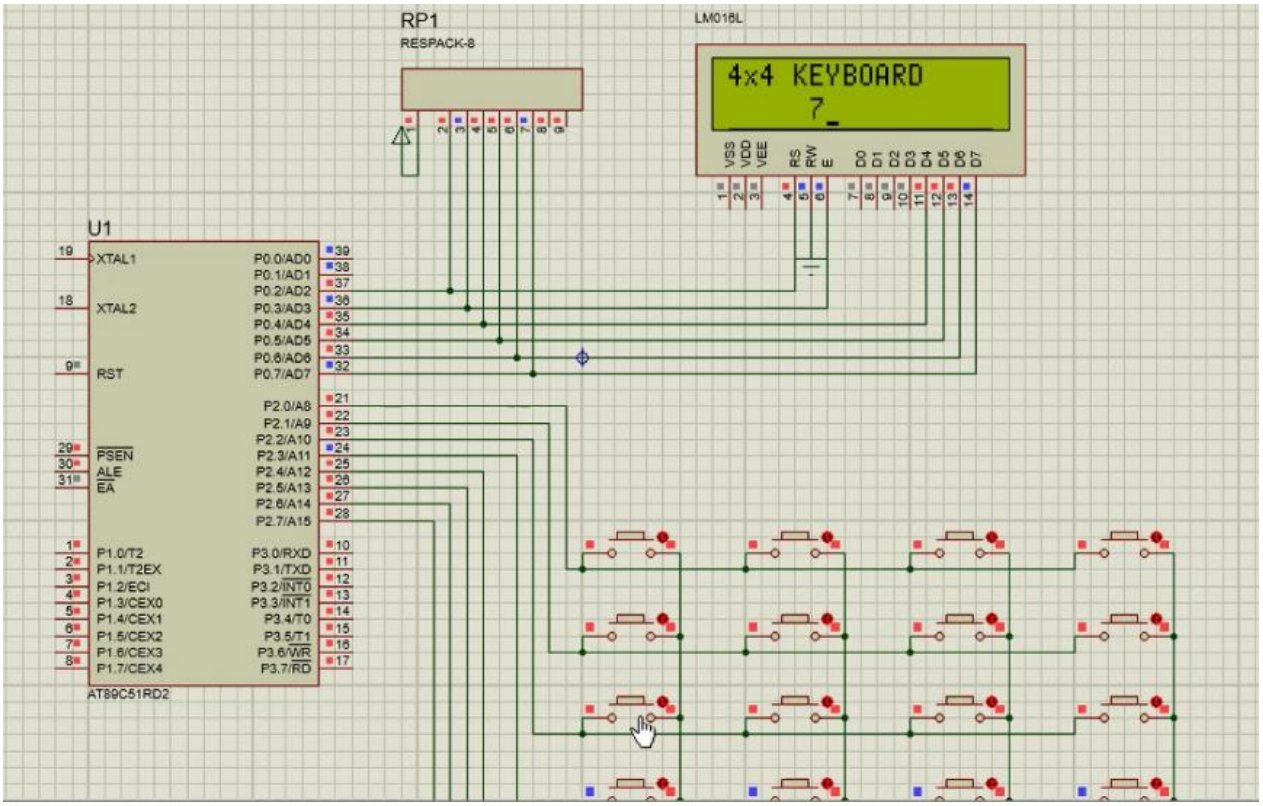
Let assume 4 line are output and 4 lines are input.

Columns are input.

Row are output.

C = Columns
R = Rows

	C1	C2	C3	C4
R1	0	1	2	3
R2	4	5	6	7
R3	8	9	A	B
R4	C	D	E	F



Conclusion:

This small project was very intersing and I learned how the 4X4 keypad works.

Appendex:

LCD.h

```
#define lcd P0
 sbit rs = P0^2;
 sbit en = P0^3;

 void delay(int ms)
 {
    int i,j;
    for(i=0;i<=ms;i++)
    for(j=0;j<=120;j++);
 }

 void tog()
 {
    en = 1; delay(1); en = 0;
 }

 void lcd_cmd_hf(char v1)
 {
    rs = 0;
    lcd &= 0x0F;
    lcd |= (v1 & 0xF0);
    tog();
 }

 void lcd_cmd(char v2)
 {
    rs = 0;
    lcd &= 0x0F;
    lcd |= (v2 & 0xF0);
    tog();

    lcd &= 0x0F;
    lcd |= ((v2 & 0x0F)<<4);
    tog();
 }

 void lcd_dwr(char v3)
 {
    rs = 1;
    lcd &= 0x0F;
    lcd |= (v3 & 0xF0);
    tog();

    lcd &= 0x0F;
    lcd |= ((v3 & 0x0F)<<4);
    tog();
 }

 void lcd_init()
 {
    lcd_cmd_hf(0x30);
    lcd_cmd_hf(0x30);
    lcd_cmd_hf(0x20);
    lcd_cmd(0x28);
```

```
    lcd_cmd(0x0E);
    lcd_cmd(0x01);
    lcd_cmd(0x06);
    lcd_cmd(0x80);
 }

 void lcd_msg(char *c)
 {
    while(*c != 0)
        lcd_dwr(*c++);
 }
```

Main.c

```
#include<reg51f.h>    //---Reg file 8051
#include"lcd.h"
sbit r1 = P2^0; //----R1 is the row 1
sbit r2 = P2^1; //----Row 2
sbit r3 = P2^2; //----Row 3
sbit r4 = P2^3; //----Row 4

sbit c1 = P2^4; //----Column 1
sbit c2 = P2^5; //----Column 2
sbit c3 = P2^6; //----Column 3
sbit c4 = P2^7; //----Column 4

//----Prot0-type----//
void scan();
//----Main Program----//
void main()
{
        P2 = 0xF0;      //---PORT2(0-3) are output and PORT2(4-7) are output
        P0 = 0x00;  //---PORT0 output
        lcd_init();
        lcd_msg("4x4 KEYBOARD");
        while(1)
        {
            scan();
        }
}

void scan()
{
    r1 = 0; r2 = r3 = r4 = 1;   //---Row 1 = 0; and all other row is high
    //--Check for for columns one and wait for 32 ms for debounce peroid
    //--if still key is pressed dispaly corresponding value --//
    if(c1 == 0){delay(32);while(c1 == 0){lcd_cmd(0xC5);lcd_dwr('0');}}
    //--Check for for columns two and wait for 32 ms for debounce peroid
    //--if still key is pressed dispaly corresponding value --//
    if(c2 == 0){delay(32);while(c2 == 0){lcd_cmd(0xC5);lcd_dwr('1');}}
    //--Check for for columns three and wait for 32 ms for debounce peroid
    //--if still key is pressed dispaly corresponding value --//
    if(c3 == 0){delay(32);while(c3 == 0){lcd_cmd(0xC5);lcd_dwr('2');}}
    //--Check for for columns four and wait for 32 ms for debounce peroid
    //--if still key is pressed dispaly corresponding value --//
    if(c4 == 0){delay(32);while(c4 == 0){lcd_cmd(0xC5);lcd_dwr('3');}}
```

```
    r2 = 0; r1 = r3 = r4 = 1; //---Row 2 = 0; and all other row is high
    //--Check for for columns one and wait for 32 ms for debounce peroid
    //--if still key is pressed dispaly corresponding value --//
    if(c1 == 0){delay(32);while(c1 == 0){lcd_cmd(0xC5);lcd_dwr('4');}}
    //--Check for for columns two and wait for 32 ms for debounce peroid
    //--if still key is pressed dispaly corresponding value --//
    if(c2 == 0){delay(32);while(c2 == 0){lcd_cmd(0xC5);lcd_dwr('5');}}
    //--Check for for columns three and wait for 32 ms for debounce peroid
    //--if still key is pressed dispaly corresponding value --//
    if(c3 == 0){delay(32);while(c3 == 0){lcd_cmd(0xC5);lcd_dwr('6');}}
    //--Check for for columns four and wait for 32 ms for debounce peroid
    //--if still key is pressed dispaly corresponding value --//
    if(c4 == 0){delay(32);while(c4 == 0){lcd_cmd(0xC5);lcd_dwr('7');}}

    r3 = 0; r1 = r2 = r4 = 1; //----Row 3 = 0; rest all row are 1
    //--Check for for columns one and wait for 32 ms for debounce peroid
    //--if still key is pressed dispaly corresponding value --//
    if(c1 == 0){delay(32);while(c1 == 0){lcd_cmd(0xC5);lcd_dwr('8');}}
    //--Check for for columns two and wait for 32 ms for debounce peroid
    //--if still key is pressed dispaly corresponding value --//
    if(c2 == 0){delay(32);while(c2 == 0){lcd_cmd(0xC5);lcd_dwr('9');}}
    //--Check for for columns three and wait for 32 ms for debounce peroid
    //--if still key is pressed dispaly corresponding value --//
    if(c3 == 0){delay(32);while(c3 == 0){lcd_cmd(0xC5);lcd_dwr('A');}}
    //--Check for for columns four and wait for 32 ms for debounce peroid
    //--if still key is pressed dispaly corresponding value --//
    if(c4 == 0){delay(32);while(c4 == 0){lcd_cmd(0xC5);lcd_dwr('B');}}

    r4 = 0; r1 = r2 = r3 = 1;       //----Row 4 = 0; all rest are 1
    //--Check for for columns one and wait for 32 ms for debounce peroid
    //--if still key is pressed dispaly corresponding value --//
    if(c1 == 0){delay(32);while(c1 == 0){lcd_cmd(0xC5);lcd_dwr('C');}}
    //--Check for for columns two and wait for 32 ms for debounce peroid
    //--if still key is pressed dispaly corresponding value --//
    if(c2 == 0){delay(32);while(c2 == 0){lcd_cmd(0xC5);lcd_dwr('D');}}
    //--Check for for columns three and wait for 32 ms for debounce peroid
    //--if still key is pressed dispaly corresponding value --//
    if(c3 == 0){delay(32);while(c3 == 0){lcd_cmd(0xC5);lcd_dwr('E');}}
    //--Check for for columns four and wait for 32 ms for debounce peroid
    //--if still key is pressed dispaly corresponding value --//
    if(c4 == 0){delay(32);while(c4 == 0){lcd_cmd(0xC5);lcd_dwr('F');}}
}
```