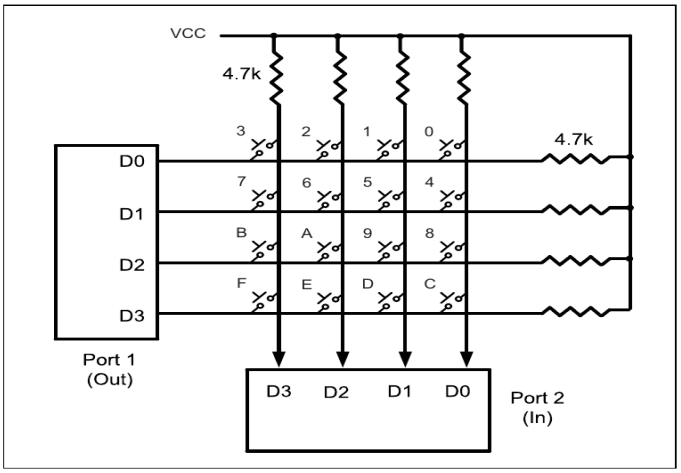
KEYBOARD INTERFACING

- Keyboards are organized in a matrix of rows and columns
- The CPU accesses both rows and columns through ports
- With two 8-bit ports, an 8 × 8 matrix of keys can be connected to a microcontroller
- When a key is pressed, a row and a column make a contact; otherwise, there is no connection between rows and columns.

Scanning and identifying the key

Figure below shows a 4 × 4 matrix connected to two ports.



- We can also use a single port in which 4 pins can bes used as input and 4 pins as output
- The rows are connected to an output port and the columns are connected to an input port
- If no key has been pressed, the input port will be 1s for all columns since they are all connected to high (VCC)
- Initially all the rows are grounded (ie. all bits are made 0s). When a key is pressed, one of the columns will have 0 since the key pressed provides the path to ground. So when a column become 0, the system realize that a key has been pressed and the next step is to identify the key.
-

The microcontroller takes the following steps to detect and identify the key:

- Grounds all rows by providing 0 to the output and then it reads the columns.
- If the data read from the columns is D3–D0 = 1111, no key has been pressed.
- If one of the column bits has a zero (ie. a key has been pressed), then microcontroller grounds the first row only, and reads the columns.
- If the data read is all 1s, no key in that row is pressed and the process (grounding) is moved to the next row and checks for any 0 in the input
- This process continues until the row is identified

For example, in the row, if D3 – D0 is 1110 and in the column, D3 – D0 is 1011, Key 2 is pressed.

Key debouncing: After the key press detection, the microcontroller waits 20 ms for the bounce and then scans the columns again. It prevents the same key press from being interpreted as a multiple key press.

Qn: Develop an AVR C program to interface a 4x4 keyboard. This program accepts numeric keys from the keypad and send it to a seven segment LED display (Common Cathode).

```
#define F_CPU 1000000UL
#include <avr/io.h>
#include <util/delay.h>

unsigned char getKey();
int main(void)
{
    unsigned char x;
        DDRC=0x0F;                      //PC.0 to PC.3 as output and PC.4 to PC.
as input
        DDRB = 0xFF;            // Set PortB as output

    while (1)
    {
                PORTC=0xF0;             //No keys pressed
                _delay_ms(20);
                if (PINC != 0xF0)       //If any key pressed
                {
                        x=getKey();             //Gets the seven segment LED cod
of the pressed key
                        PORTB=x;                // Send it to PortB
                }
        }
        return 0;
}

unsigned char getKey()
{
        PORTC=0b11111110;                       //Grounding first row
        if ((PINC & (1<<4))==0)         // Reading first column
        {
                _delay_ms(20);
                return 0x07;
        }
        else if ((PINC & (1<<5))==0)    // Reading second column
        {
                _delay_ms(20);
                return 0x7f;
        }
        else if ((PINC & (1<<6))==0)    // Reading third column
        {
                _delay_ms(20);
                return 0x6f;
        }


        PORTC=0b11111101;               //Grounding second row
        if ((PINC & (1<<4))==0)         // Reading first column
        {
                _delay_ms(20);
```

```
            return 0x66;
        }
        else if ((PINC & (1<<5))==0)    // Reading second column
        {
            _delay_ms(20);
            return 0x6d;
        }
        else if ((PINC & (1<<6))==0)    // Reading third column
        {
            _delay_ms(20);
            return 0x7d;
        }

        PORTC=0b11111011;               //Grounding third row
        if ((PINC & (1<<4))==0)         // Reading first column
        {
            _delay_ms(20);
            return 0x06;
        }
        else if ((PINC & (1<<5))==0)    // Reading second column
        {
            _delay_ms(20);
            return 0x5b;
        }
        else if ((PINC & (1<<6))==0)    // Reading third column
        {
            _delay_ms(20);
            return 0x4f;
        }

        PORTC=0b11110111;               //Grounding fourth row
        if ((PINC & (1<<5))==0)         // Reading first column
        {
            _delay_ms(20);
            return 0x3f;
        }
        else return 0;


}
```