LCD and Keypad

(chapter 12)

MBSD, 6th Semester

DCSE, UET Peshawar

Bilal Habib

INTERFACING LCD TO 8051

LCD Operation

- LCD is finding widespread use replacing LEDs
  - The declining prices of LCD
  - The ability to display numbers, characters, and graphics
  - Incorporation of a refreshing controller into the LCD, thereby relieving the CPU of the task of refreshing the LCD
  - Ease of programming for characters and graphics

INTERFACING LCD TO 8051

LCD Pin Descriptions

Pin Descriptions for LCD

Pin	Symbol	I/O	Descriptions
1	VSS	--	Ground
2	VCC	--	+5V power supply
3	VEE	--	Power supply to control contrast
4	RS	I	RS=0 to select command register, RS=1 to select data register
5	R/W	I	R/W=0 for write, R/W=1 for read
6	E	I/O	Enable
7	DB0	I/O	The 8-bit data bus
8	DB1	I/O	The 8-bit data bus
9	DB2	I/O	The 8-bit data bus
10	DB3	I/O	The 8-bit data bus
11	DB4	I/O	The 8-bit data bus
12	DB5	I/O	The 8-bit data bus
13	DB6	I/O	The 8-bit data bus
14	DB7	I/O	The 8-bit data bus

used by the LCD to latch information presented to its data bus

- Send displayed information or instruction command codes to the LCD
- Read the contents of the LCD's internal registers

INTERFACING LCD TO 8051

LCD Command Codes

LCD Command Codes

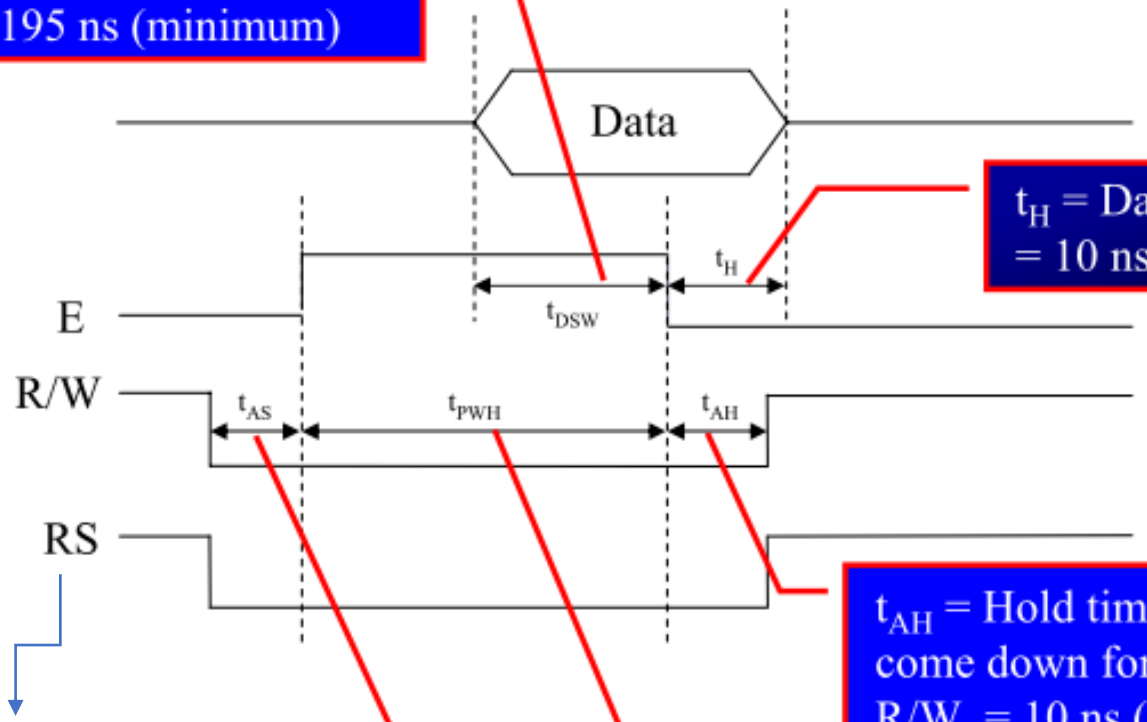
Code (Hex)	Command to LCD Instruction Register
1	Clear display screen
2	Return home
4	Decrement cursor (shift cursor to left)
6	Increment cursor (shift cursor to right)
5	Shift display right
7	Shift display left
8	Display off, cursor off
A	Display off, cursor on
C	Display on, cursor off
E	Display on, cursor blinking
F	Display on, cursor blinking
10	Shift cursor position to left
14	Shift cursor position to right
18	Shift the entire display to the left
1C	Shift the entire display to the right
80	Force cursor to beginning to 1st line
C0	Force cursor to beginning to 2nd line
38	2 lines and 5x7 matrix

INTERFACING LCD TO 8051

LCD Data Sheet (cont')

LCD Timing

t_{DSW} = Data set up time = 195 ns (minimum)



t_H = Data hold time = 10 ns (minimum)

t_{AH} = Hold time after E has come down for both RS and R/W = 10 ns (minimum)

t_{PWH} = Enable pulse width = 450 ns (minimum)

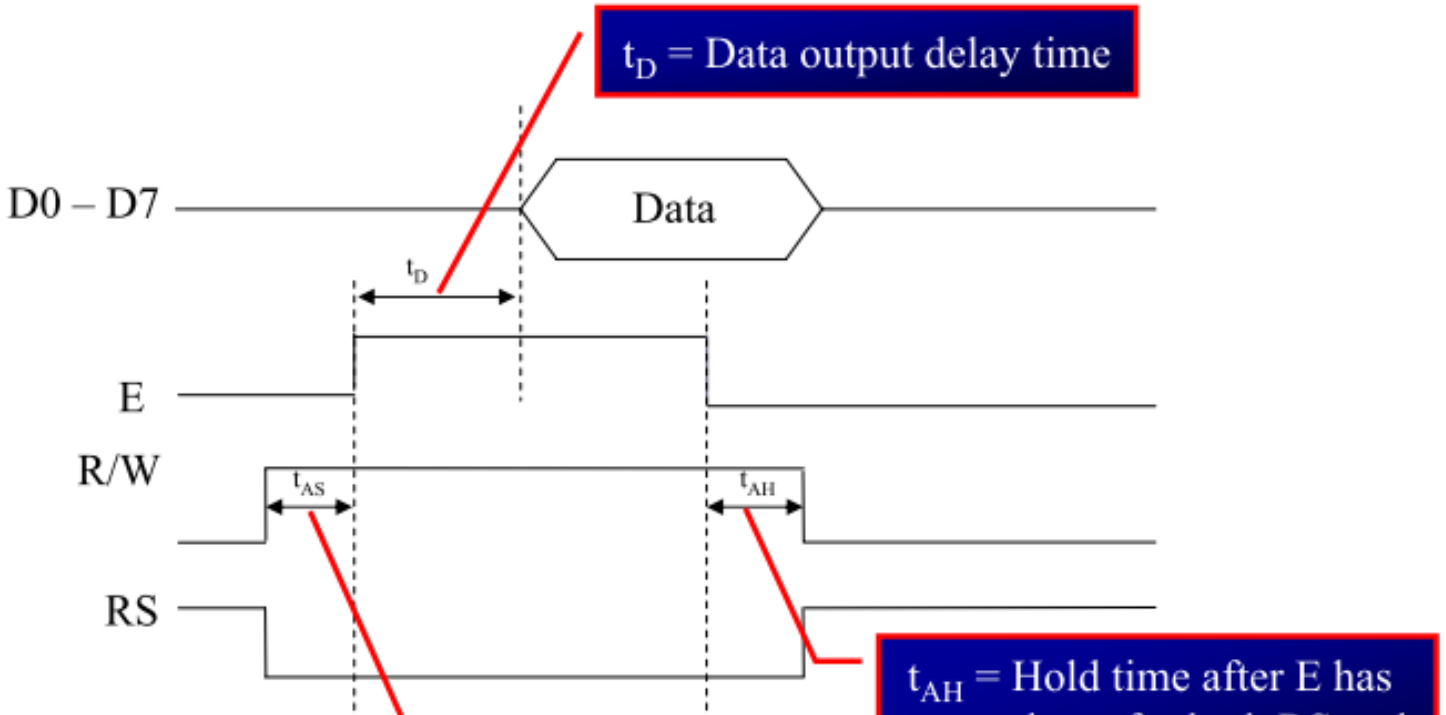
t_{AS} = Set up time prior to E (going high) for both RS and R/W = 140 ns (minimum)

RS = 0, Command

LCD INTERFACING

Sending Codes and Data to LCDs w/ Busy Flag (cont')

LCD Timing for Read

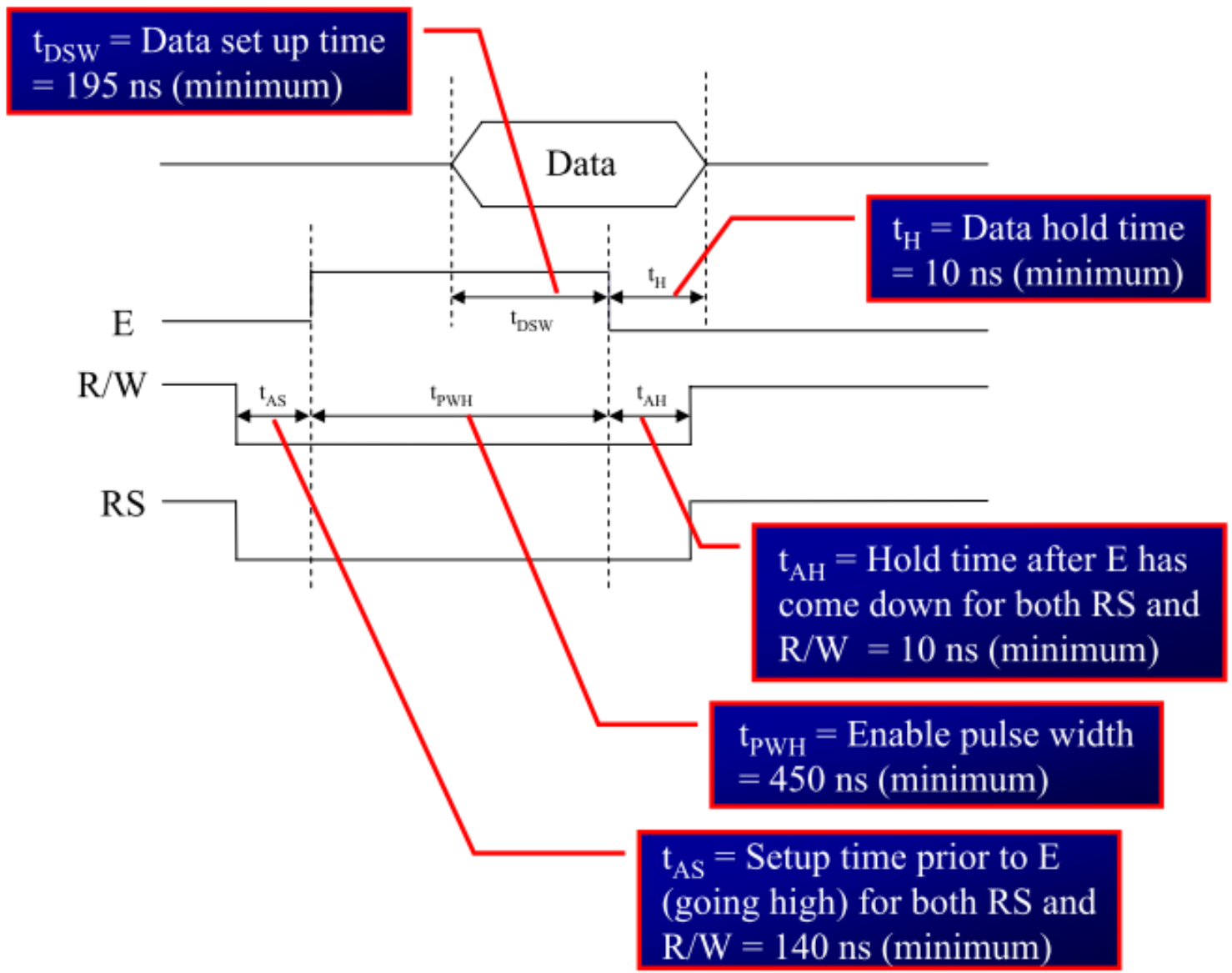


Note : Read requires an L-to-H pulse for the E pin

LCD INTERFACING

Sending Codes and Data to LCDs w/ Busy Flag (cont')

LCD Timing for Write



Write Command and Write Data

```
void writecmd(int z)
{
   RS = 0;              // This is command
   P2 = z;              //Data transfer
   E  = 1;              // => E = 1
   delay(150);
   E  = 0;              // => E = 0
   delay(150);
}
```

```
void writedata(char t)
{
   RS = 1;              // This is data
   P2 = t;              //Data transfer
   E  = 1;              // => E = 1
   delay(150);
   E  = 0;              // => E = 0
   delay(150);
}
```

Keypad Interfacing

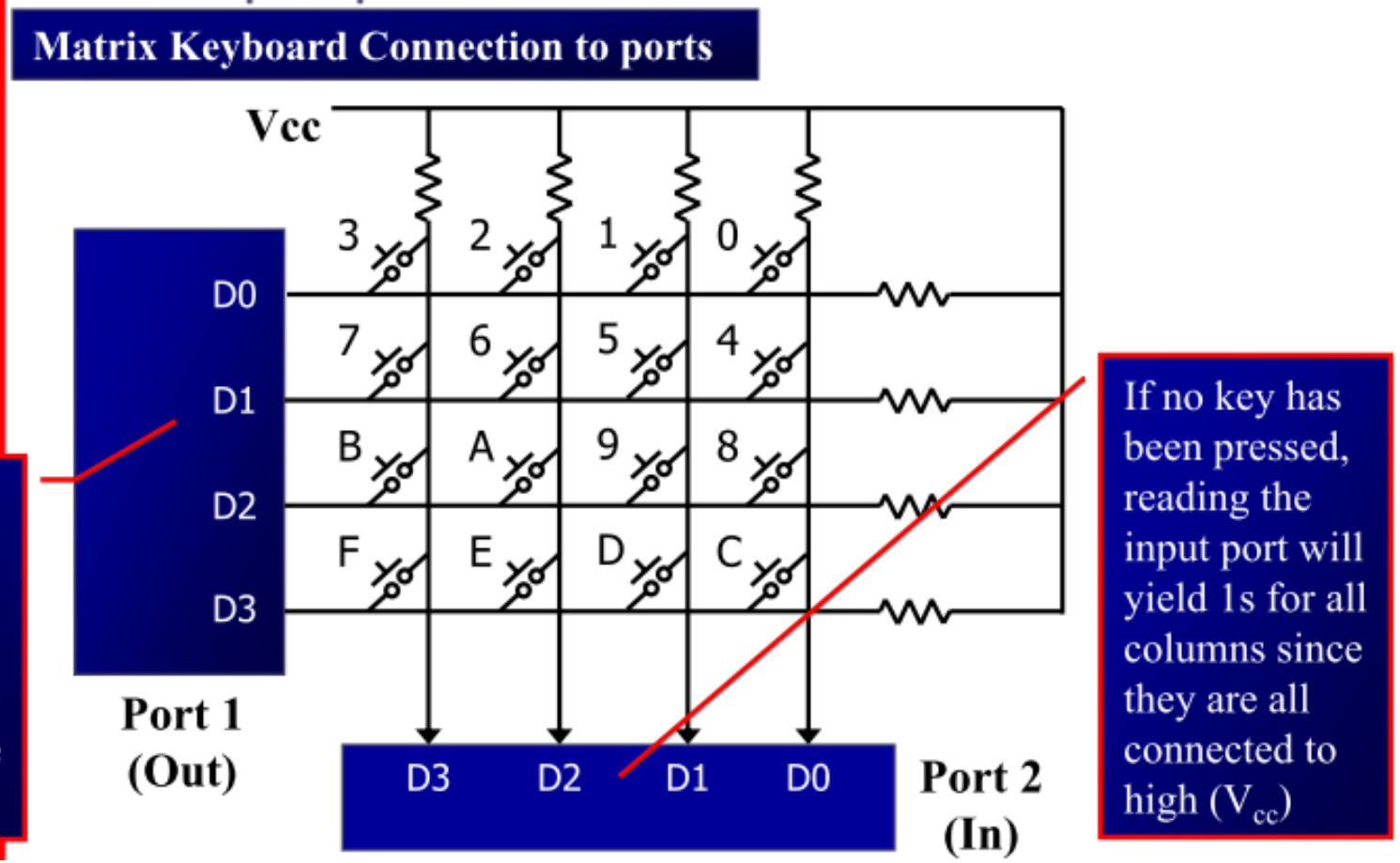
KEYBOARD INTERFACING

- Keyboards are organized in a matrix of rows and columns
  - The CPU accesses both rows and columns through ports
    - Therefore, with two 8-bit ports, an 8 x 8 matrix of keys can be connected to a microprocessor
  - When a key is pressed, a row and a column make a contact
    - Otherwise, there is no connection between rows and columns
- In IBM PC keyboards, a single microcontroller takes care of hardware and software interfacing

KEYBOARD INTERFACING

Scanning and Identifying the Key

- A 4x4 matrix connected to two ports
  - The rows are connected to an output port and the columns are connected to an input port



If all the rows are grounded and a key is pressed, one of the columns will have 0 since the key pressed provides the path to ground

If no key has been pressed, reading the input port will yield 1s for all columns since they are all connected to high (V_{cc})

KEYBOARD INTERFACING

Grounding Rows and Reading Columns

- It is the function of the microcontroller to scan the keyboard continuously to detect and identify the key pressed
- To detect a pressed key, the microcontroller grounds all rows by providing 0 to the output latch, then it reads the columns
  - If the data read from columns is D3 – D0 = 1111, no key has been pressed and the process continues till key press is detected
  - If one of the column bits has a zero, this means that a key press has occurred
    - For example, if D3 – D0 = 1101, this means that a key in the D1 column has been pressed
    - After detecting a key press, microcontroller will go through the process of identifying the key

KEYBOARD INTERFACING

Grounding Rows and Reading Columns (cont')

- Starting with the top row, the microcontroller grounds it by providing a low to row D0 only
  - It reads the columns, if the data read is all 1s, no key in that row is activated and the process is moved to the next row
- It grounds the next row, reads the columns, and checks for any zero
  - This process continues until the row is identified
- After identification of the row in which the key has been pressed
  - Find out which column the pressed key belongs to

KEYBOARD INTERFACING

Grounding Rows and Reading Columns (cont')

Example 12-3

From Figure 12-6, identify the row and column of the pressed key for each of the following.

(a) D3 – D0 = 1110 for the row, D3 – D0 = 1011 for the column

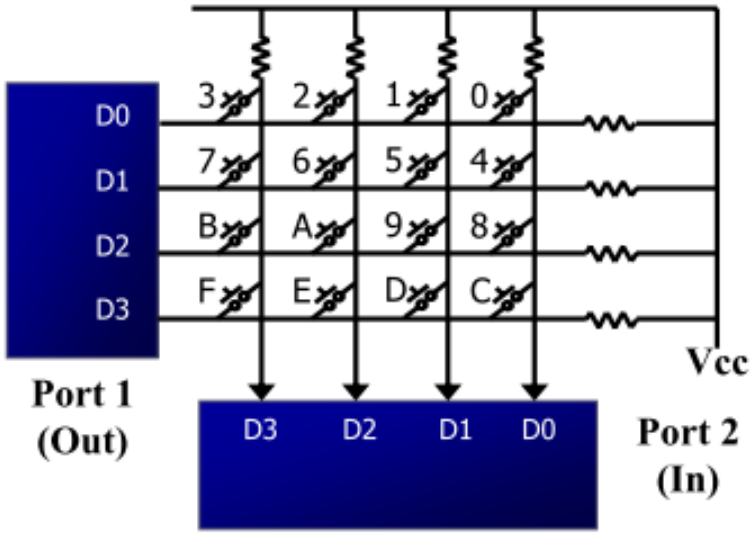
(b) D3 – D0 = 1101 for the row, D3 – D0 = 0111 for the column

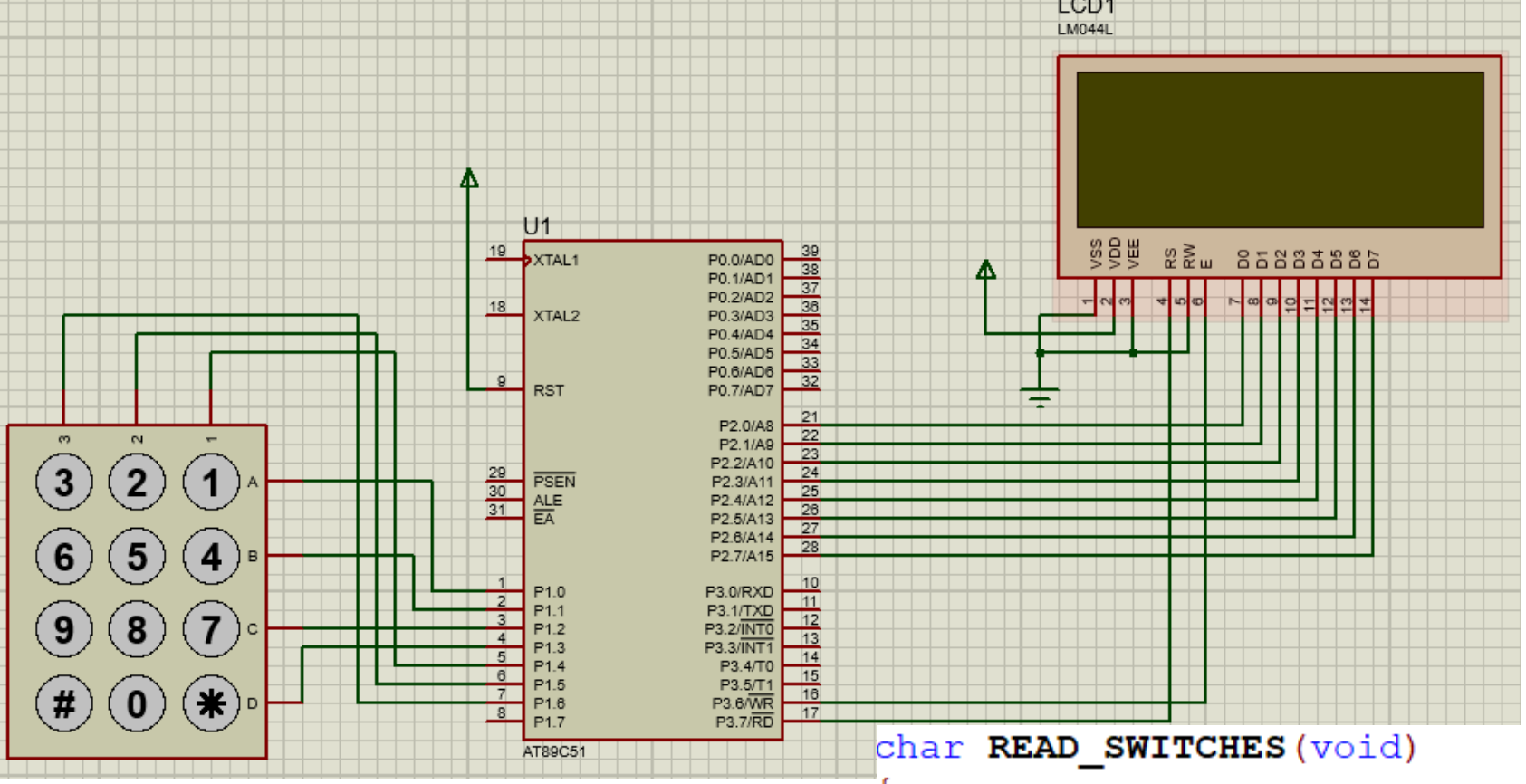
Solution :

From Figure 13-5 the row and column can be used to identify the key.

(a) The row belongs to D0 and the column belongs to D2; therefore, key number 2 was pressed.

(b) The row belongs to D1 and the column belongs to D3; therefore, key number 7 was pressed.






```
char READ_SWITCHES(void)
{
    RowA = 0; RowB = 1; RowC = 1; RowD = 1;      //Test Row A

    if (C1 == 0)
        { delay(10000);
            while (C1==0);
        return '1'; }

    if (C2 == 0)
        { delay(10000);
        while (C2==0);
        return '2'; }
```

```
char READ_SWITCHES(void)
{
    RowA = 0; RowB = 1; RowC = 1; RowD = 1;      //Test Row A

    if (C1 == 0) { delay(10000); while (C1==0); return '1'; }
    if (C2 == 0) { delay(10000); while (C2==0); return '2'; }
    if (C3 == 0) { delay(10000); while (C3==0); return '3'; }

    RowA = 1; RowB = 0; RowC = 1; RowD = 1;      //Test Row B

    if (C1 == 0) { delay(10000); while (C1==0); return '4'; }
    if (C2 == 0) { delay(10000); while (C2==0); return '5'; }
    if (C3 == 0) { delay(10000); while (C3==0); return '6'; }
```