EC336: Embedded Systems

Interfacing Display Devices

Arvind S Kumar (15EC106), Pavan M (15EC137),
Samarth B (15EC143), Sripathi M (15EC149)

22 August 2018

Submitted to:

Prof. Ramesh Kini M, Prof. Arulalan Rajan

Department of Electronics and Communication Engineering

NITK, Surathkal

1 Goal of the Lab

The aim of this lab was to interface the chosen microcontroller with input devices such as the Hex keypad and the Keyboard. The microcontroller chosen by our group is TI's MSP430G2.

2 Components Used

For this lab the components used are :

- MSP430 microcontroller
- LEDs
- 220 Ω resistors
- Hex Keypad
- Keyboard

3 Board Details

Some of the features of the MSP430 are - (Refer Figure 1 and Figure 2 for more information)

- 16-bit RISC architecture
- Von-Neumann architecture
- Upto 16 MHz clock frequency
- Low power consumption and five power saving modes
- 16 general purpose registers

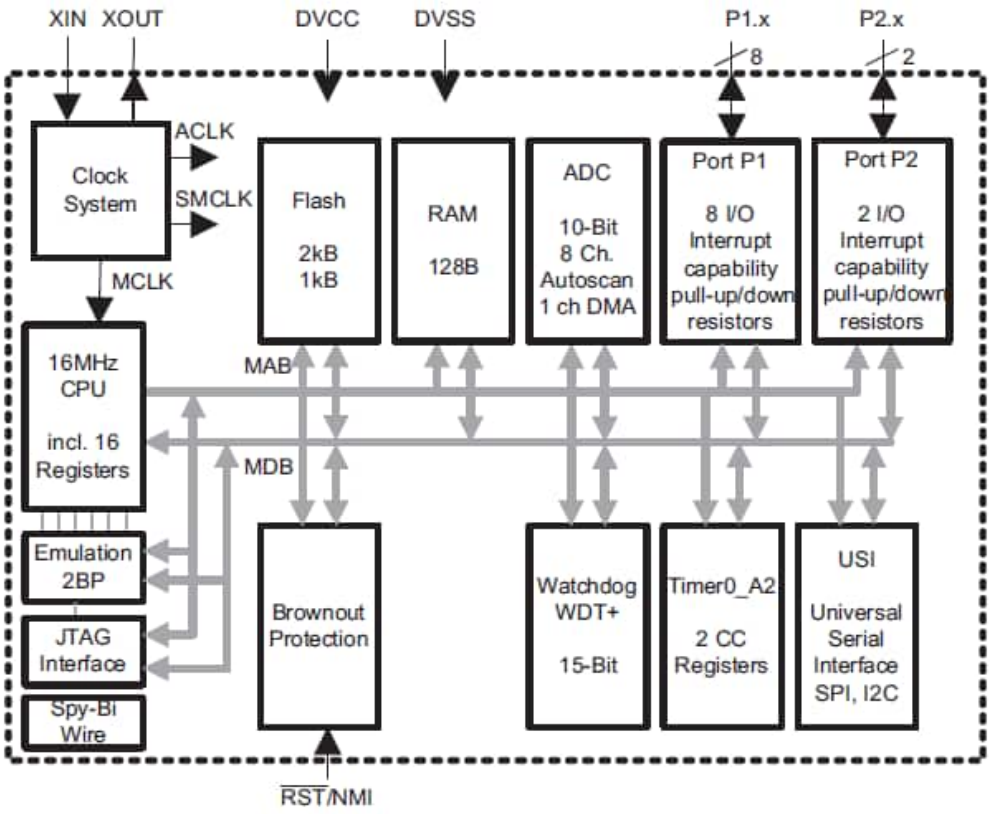
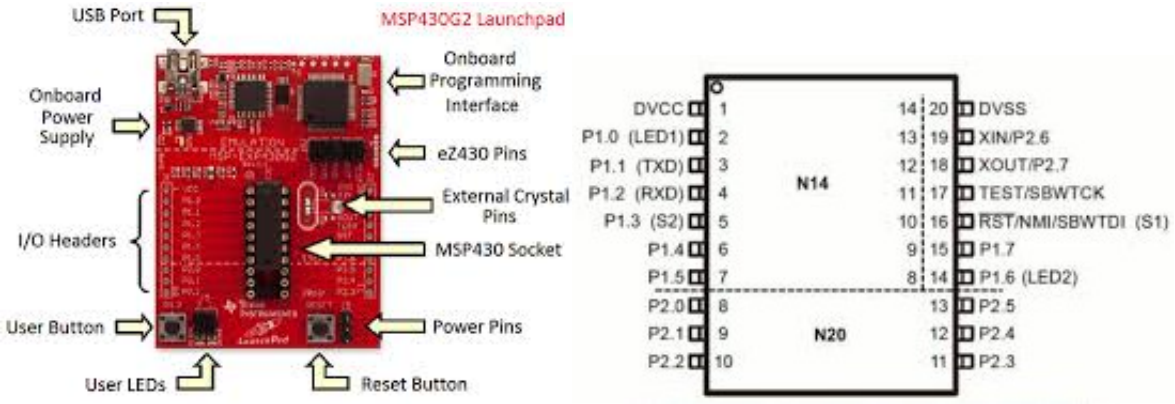


Figure 1: MSP430 Architecture

- 10-bit ADC
- Supports SPI, I2C, UART and USB interfaces
- Has 16 GPIO pins divided as two ports (8-bit each)

4 Interfacing Details

Two input devices - the Hex Keypad and the Keyboard were interfaced. Both the devices were manufactured by Electro Systems, Bangalore and datasheets for the devices are available at the Embedded Lab. In order to observe



(a) Board (b) Pin Diagram

Figure 2: Board details

the outputs of these devices, we also connected LEDs to the MSP430 and programmed a software decoder to translate key presses to a 4-bit binary value that were displayed using LEDs.

4.1 General layout

The internal schematic for each of the devices discussed here will be elaborated in the respective sections. The general structure, however, is the same. Each of the keys act like push-buttons in the circuit. There are two controlling lines that help the user detect which key was pressed. The lines and the keys form a matrix. Each line controlling the rows are inputs and line controlling the columns are outputs from the peripheral.

Initially, all the rows are enabled and the MSP waits for a key press. As soon as a key is pressed, the column is detected and to detect the row, the rows are enabled one by one and the column is monitored. The enabled row for which the column is also enabled, gives us the exact location of the key pressed.

VCC = 5V and GND are connected to the MSP 5V output (TP1) and

GND respectively.

4.2 Hex Keypad

The hex keypad is a 4x4 keypad with numbers from 0-9 and the hex digits A-F. Internally it is structured as 4x4 matrix. The rows are connected to Port 2 of the MSP and the columns are connected to Port 1 of the MSP. 4 LEDs are connected to Port 2 to display the decoded output. Port 2 is set to output and Port 1 to input.

4.3 Keyboard

The keyboard is a 4x6 keyboard with numbers from 0-9, basic math operators - $=, ., +, -, *, /$ and % and some basic calculator operators M+, M-, MR, MC, AC, CE and CHK. Internally it is structured as 3x8 matrix. The rows are connected to Port 2 of the MSP and the columns are connected to Port 1 of the MSP. 4 LEDs are connected to Port 2 to display the decoded output. Port 2 is set to output and Port 1 to input.

For ease of display and due to less number of pins available, the row controlling the calculator functions are disabled.

The internal structure of the keyboard is as shown in Fig 3.

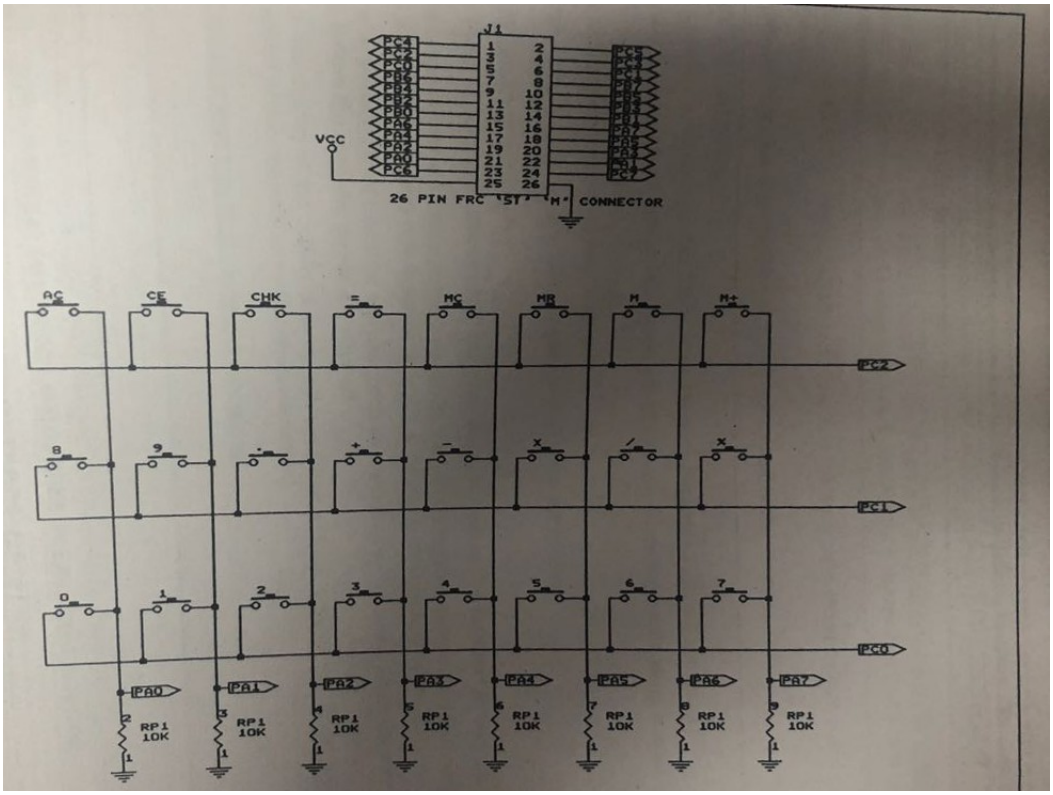


Figure 3: Keyboard

5 Program

Since the procedure for both the keypad and the keyboard are similar, the code here is presented for the more complex keyboard.

Interfacing Keyboard using MSP430	**Pseudo code**
#include <msp430.h> int main(void){ WDTCTL = WDTPW | WDTHOLD; P2DIR = 0x3F;	 Stop watchdog timer P2 0-3 used for rows (PCs - input, writing to it)
Continued on next page	

Table 1 – continued from previous page

P1DIR = 0x00;	P1 0-7 used for columns (PAs - output, reading from it)
unsigned int output, i, column, row, tempP2OUT;	
unsigned int digits[2][8] = {{0, 1, 2, 3, 4, 5, 6, 7}, {8, 9, 10, 11, 12, 13, 14, 15}};	Array to store which row and column Corresponds to which Key on Keyboard
for(;;) {	Begin loop
P2OUT = 0x03;	Setting All Rows to high
column = P1IN;	Reading Column Input
if(column ! = 0x00) {	Until Any of the Column is Triggered
switch(column) {	Finding out which column is triggered
case 0x01: column = 0; break;	
case 0x02: column = 1; break;	
case 0x04: column = 2; break;	
case 0x08: column = 3; break;	
case 0x10: column = 4; break	
case 0x20: column = 5; break;	
case 0x40: column = 6; break;	
case 0x80: column = 7; break;	

Continued on next page

Table 1 – continued from previous page

default: column = 8;	Checking Condition for more than one column pressed
}	
if(column ! = 8) {	if more than one column is not pressed
P2OUT = 0x00;	Setting All Rows
for(i = 0; i <= 1; i++) {	Setting rows ON one by one to find out which row is pressed
P2OUT = 0x02 >> i;	
if(P1IN ! = 0x00)	
row = i;	
}	
}	
tempP2OUT = (digits[row][column] << 2) & 0x3C;	
P2OUT = 0x20 ^ tempP2OUT;	Setting Out Accordingly to turn on Externally Connected LEDs
}	
}	
return 0;	
}	

6 Knowledge gained

1. Learnt how keyboard and keypad is designed and controlled.
2. Understood the importance of datasheet and that it must be followed to the dot. For example, while interfacing the keyboard, the datasheet

showed that the schematic was for a 3x8 key matrix whereas what we had on the board was a 4x6 matrix. We wasted a lot of time since we didn't believe the datasheet.

3. Learnt the importance and need for debouncing. Hardware debouncing can be implemented using latches embedded in the keyboard or software debouncing.