EC336: Embedded Systems

Interfacing Display Devices

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

15 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 display devices such as LEDs, 7-segment displays and LCD. The micro controller chosen by our group is TI's MSP430G2.

2 Components Used

For this lab the components used are :

- MSP430 microcontroller
- LEDs
- 2.2 kΩ resistors
- 2 7-segment displays
- LCD

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

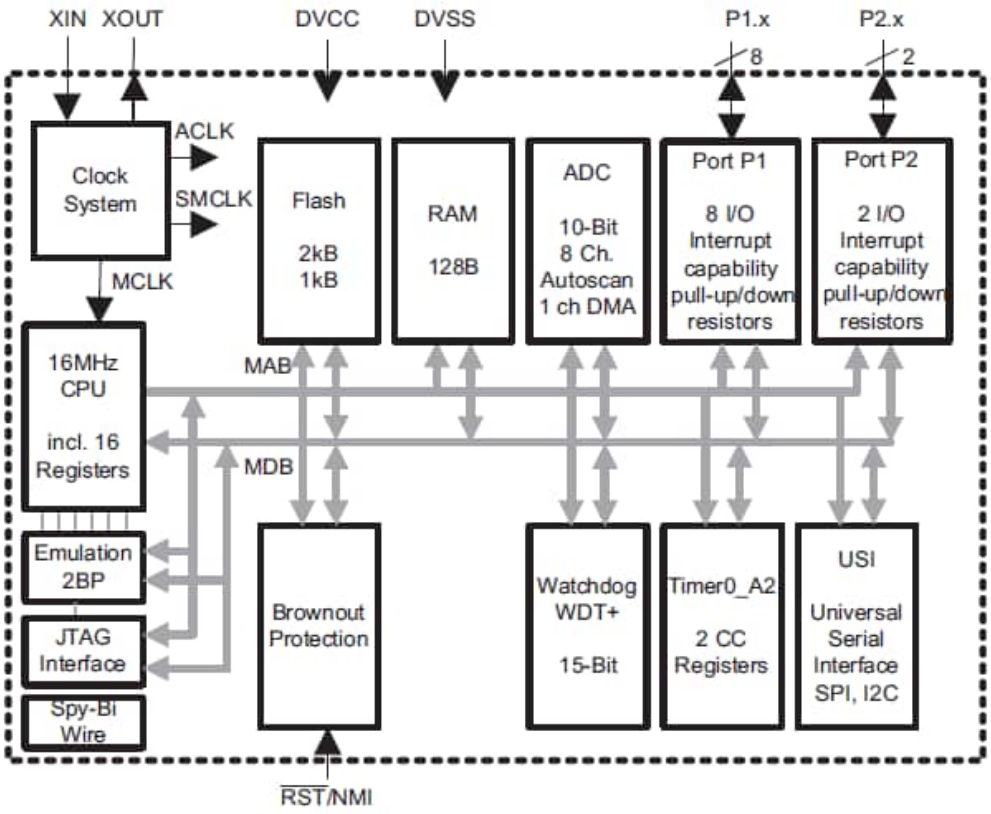
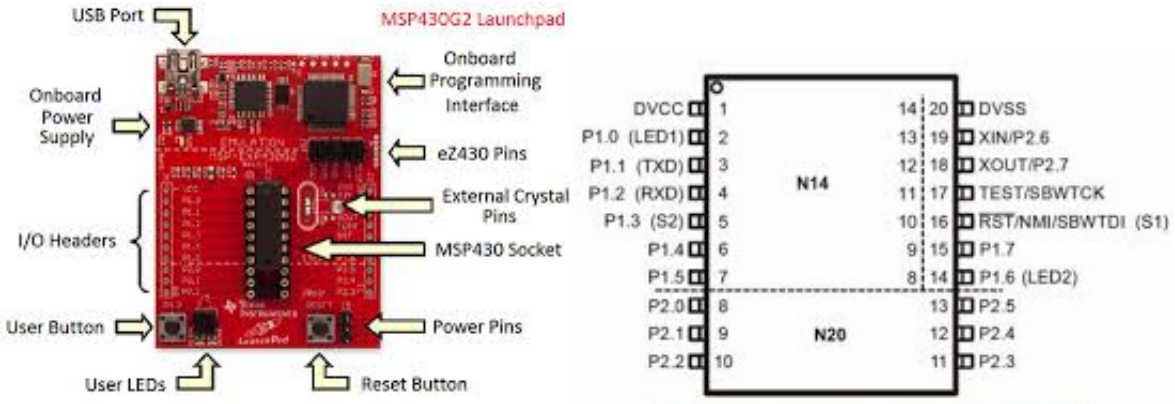


Figure 1: MSP430 Architecture

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

4 Interfacing Details

Three display devices - LEDs, 7-segment displays and an LCD display were interfaced. The details for each of them are as follows -



(a) Board (b) Pin Diagram

Figure 2: Board details

4.1 LEDs

To do this, first the on-board LEDs (LED1 and LED2) were made to blink using the given example program. LED1 is connected to Port 1 BIT0 and LED2 is connected to Port 2 BIT6.

To blink an external LED, we connected an LED through a 2.2k resistor to Port 1 BIT4.

The delays were controlled using do...while loops

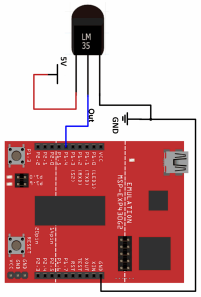


Figure 3: LED interface

4.2 7-Segment Display

To interface the 7-segment display, an array of the 7-segment 8 bit patterns were created and stored. A loop is run through these values and a delay is provided using do...while loops to be able to observe the transition. All 8 pins of Port 1 were used to interface the 7-segment display (including the decimal point)

Next, a two digit counter was made to count from 0 to 99. Both the 7-segment displays were connected to the same data lines (i.e Port 1) and to switch between the two displays the COM ports of the displays were controlled using a signal. This signal was generated using multiplexing between two bits on Port 2. By controlling the amount of delay, we were able to generate the illusion of a two digit counter.

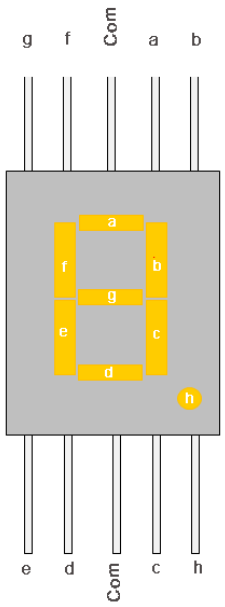


Figure 4: 7-Segment Display

4.3 LCD

The LCD was interfaced in the 4-bit 2-line configuration. The LCD was connected to the MSP430 as shown. 4 pins from Port 1 (BIT0 to BIT3) were connected to the LCD data pins D7-D4. The EN and RS pins of the LCD were connected to BIT4 and BIT5 of Port 1 respectively. Since we only want to write to the LCD, the RW pin of the LCD was grounded. To control the contrast, the contrast pin of the LCD was connected to a potential divider created using 4 2.2 kΩ resistors in a 3:1 configuration. The VCC and VSS pins of the LCD were connected to the 5V supply of the MSP430 (T1) and

ground respectively. These help control the intensity of the back light.

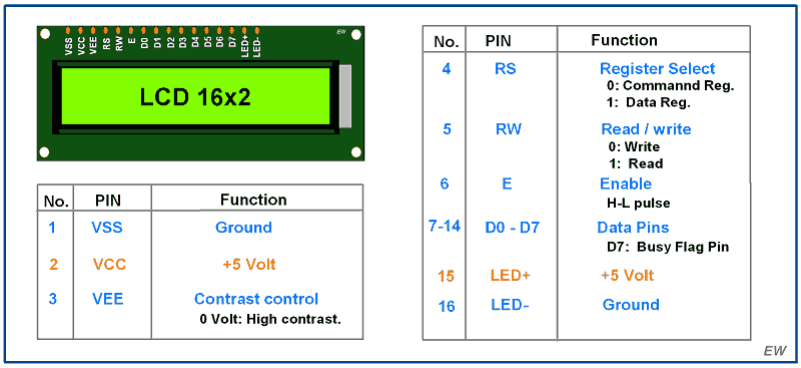


Figure 5: LCD PIN diagram

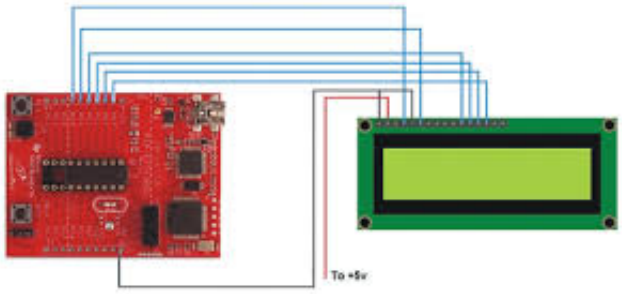


Figure 6: LCD interface

In order to print a message that is in infinite scroll on the LCD screen, a string with the message was used. Each chunk of 16 characters was sent to the LCD after a specific delay which was controlled using the delaycycles() command. At each iteration the first character of the 16 character chunk was removed and added to the end of the string.

5 Program

Scanning of 7-Segment Display using MSP430	Pseudo code
#include <msp430.h>	
#include <stdio.h>	
int main(void){	
WDTCTL = WDTPW | WDTHOLD;	Stop watchdog timer
P1DIR |= 0xff;	Set all to output direction
P2DIR |= 0x03;	set pin 2 to output direction for turning one of 7 segment on
for(;;) {	Begin loop
long i;	
int segment[10] = {0xfc, 0x60, 0xd2, 0xf2, 0x66, 0xb6, 0xbe, 0xe0, 0xfe, 0xf6};	
int var, loop;	
for(var = 0; var<100; var++)	Count 0 to 99
{	
int seg1 = segment[var/10];	selecting 10's place
int seg2 = segment[var%10];	selecting 1's place
i = 10000;	
do {	
P2OUT = 0x02;	
P1OUT = 0xFF ŝeg1;	turn on tens place and ones place off
int j = 100;	delay

Continued on next page

Table 1 – continued from previous page

do j−−;	
while(j ! = 0);	
P1OUT = 0xFF;	turning of all leds of 7 segment
j = 100;	delay
do j−−;	
while(j ! = 0);	
P2OUT = 0x01;	
P1OUT = 0xFF ŝeg2;	turn on ones place and turn off tens place
j = 100;	
do j−−;	delay
while(j ! = 0);	
P1OUT = 0xFF;	turning of all leds of 7 segment
j = 100;	
do j−−;	delay
while(j ! = 0);	
i−−;	
}while(i ! = 0);	Repeat again for some time
}	
}	
}	

Scrolling LCD Display using MSP430	Pseudo code
#include <msp430g2553.h>	
#include <string.h>	
#define lcdPort P1OUT	Defining ports
#define lcdPortDir P1DIR	
#define lcdEN BIT4	
#define lcdRS BIT5	
void lcdReset()	function to reset
{	
lcdPortDir = 0xff;	setting pins to output mode
lcdPort = 0xff;	
__delay_cycles(20000);	delay
}	
void lcdCmd (char cmd)	
{	
lcdPort = ((cmd >> 4) & 0x0F)|lcdEN;	Send upper nibble
lcdPort = ((cmd >> 4) & 0x0F);	
lcdPort = (cmd & 0x0F)|lcdEN;	Send lower nibble
lcdPort = (cmd & 0x0F);	
__delay_cycles(4000); }	
void lcdInit ()	
{	
lcdReset();	Call LCD reset
lcdCmd(0x28);	4-bit mode 2 line 5x7 font.
lcdCmd(0x0C);	Display no cursor - no blink.
lcdCmd(0x80);	Address DDRAM with 0 offset 80h.

Continued on next page

Table 2 – continued from previous page

lcdCmd(0x01);	Clear screen
}	
void lcdData (unsigned char dat)	Send the Data to LCD
{	
lcdPort = (((dat >> 4) & 0x0F)|lcdEN|lcdRS);	Send upper nibble by right shifting the dat 4 four so we get the upper 4 bits
lcdPort = (((dat >> 4) & 0x0F)|lcdRS);	
lcdPort = ((dat & 0x0F)|lcdEN|lcdRS);	Send lower nibble
lcdPort = ((dat & 0x0F)|lcdRS);	
__delay_cycles(4000);	a small delay
}	
void displayLine(char *line)	Function to Display the Character
{	
while (*line)	
lcdData(*line++);	Call lcdData function
}	
int main()	
{	
WDTCTL = WDTPW + WDTHOLD;	Stop Watch Dog Timer
lcdInit();	Initialize LCD
char name[] = "EC336 EMBEDDED SYSTEMS ";	

Continued on next page

Table 2 – continued from previous page

int len = strlen(name);	Get the length of the string
int j = 0,i,k,c;	
char sub[16];	
while(1){	Start loop
c = 0;	
lcdCmd(0x80);	
while (c < 16)	getting a substring of 16 characters
{	
sub[c] = name[j+c];	
c++;	
}	
char var = name[j];	
k = 0;	
while(k<len-1)	shifting all the characters left by one and putting the first character to the end
{	
name[k] = name[k+1];	
k++;	
}	
name[len-1] = var;	
name[len] = '\0';	
sub[c] = '\0';	
displayLine(sub);	Calling display function to print on lcd

Continued on next page

Table 2 – continued from previous page

__delay_cycles(500000);
 	
}
while (1);
}
	After small delay repeat the loop

6 Knowledge Gained

1. Exposure to a new micro-controller platform. None of us had used MSP430 before neither in our hobby projects nor in course projects and hence it was a new learning experience.

2. Shift in using in-built commands to register manipulation during programming resulted in a new learning curve.

3. Understanding how an LCD is configured internally and trying to interface it with the controller and the logic employed to get a scrolling display.