**Lab 3 - SPI - "I/O"**

**Mega Prelab**

A hard copy of this Mega Prelab is required to be turned in. Answers should not be handwritten. The timing diagram may be NEATLY drawn by hand with the assistance of a straightedge on engineering paper.

**Nokia1202 LCD BoosterPack v4-5**

Look at the schematic for the Nokia1202 LCD BoosterPack. Complete the following table. The pin number should be the pin number that signal connects to on the MSP 430. You may need to examine page 3 of the MSP430 Data Sheet. The type is from the perspective of the MSP430 and is one of the following: input, output, or power . For example, the RST' line should be listed as having an output type because it is an output from the MSP430 (to an input on the Nokia 1202). For input and output types (do nothing further with power), list their bit settings in the three registers listed (see pages 328 and 329 of the MSP430g2553 User's Guide).

| **Name** | **Pin #** | **Type** | **PxDIR** | **PxREN** | **PxOUT** |
| --- | --- | --- | --- | --- | --- |
| GND | 20 | Power |  |  |  |
| RST | 16 | Output | 1 | 0 | 0 |
| P1.4 | 6 | Output | 1 | 0 | 0 |
| MOSI | 15 | Output | 1 | 0 | 0 |
| SCLK | 7 | Output | 1 | 0 | 0 |
| VCC | 1 | Power |  |  |  |
| S1 | 9 | Output | 1 | 0 | 0 |
| S2 | 10 | Output | 1 | 0 | 0 |
| S3 | 11 | Output | 1 | 0 | 0 |
| S4 | 12 | Output | 1 | 0 | 0 |

**Configure the MSP430**

The following is a portion of the code that initalizes the directions of the pins connecting the MSP430 to the Nokia 1202 display. Use the information from the previous question to identify the names of the registers missing in the following code (identified by the letters A - D). Put the register names in the table below.

mov.b #LCD1202\_CS\_PIN|LCD1202\_BACKLIGHT\_PIN|LCD1202\_SCLK\_PIN|LCD1202\_MOSI\_PIN, & A

mov.b #LCD1202\_CS\_PIN|LCD1202\_BACKLIGHT\_PIN|LCD1202\_SCLK\_PIN|LCD1202\_MOSI\_PIN, & B

mov.b #LCD1202\_RESET\_PIN, & C

mov.b #LCD1202\_RESET\_PIN, & D

| **Mystery Label** | **Register** |
| --- | --- |
| A | P1OUT |
| B | P1DIR |
| C | P2OUT |
| D | P2DIR |

The following initializes the SPI subsystem of the MSP430. For each of the bits listed in the table below, identify how the code-snippet configures that pin and what function is realized by that setting. For example, setting the UCMSB bit of the UCB0CTL0 register forces the SPI subsystem to output the bits starting from the LSB. Also, list the bit position that each occupies in its associated register.

bis.b #UCCKPH|UCMSB|UCMST|UCSYNC, &UCB0CTL0

bis.b #UCSSEL\_2, &UCB0CTL1

bic.b #UCSWRST, &UCB0CTL1

| **ID** | **Bit** | **Function as set in the code** |
| --- | --- | --- |
| UCCKPH | 7 | Captures data on the first UCLK edge and changes it on following edges |
| UCMSB | 5 | Forces the SPI subsystem to output the bits starting from the LSB |
| UCMST | 3 | Selects the device as the master and the SPI as the slave |
| UCSYNCH | 0 | Sets the clock to synchronous |
| UCSSEL\_2 | 7 | Sets the clock to SMCLK |
| UCSWRST | 0 | Disables the software reset |

**Communicate to the Nokia1202 display**

The following code communicates one byte to the Nokia 1202 display using its 9-bit protocol. Use this code to draw a timing diagram of the expected behavior of LCD1202\_CS\_PIN, LCD1202\_SCLK\_PIN, LCD1202\_MOSI\_PINs from the begining of this subroutine to the end. Make sure that you clearly show the relationship of the edges in the clk and data waveforms.

;-------------------------------------------------------------------------------

; Name: writeNokiaByte

; Inputs: R12 selects between (1) Data or (0) Command string

; R13 the data or command byte

; Outputs: none

; Purpose: Write a command or data byte to the display using 9-bit format

;-------------------------------------------------------------------------------

writeNokiaByte:

push R12

push R13

bic.b #LCD1202\_CS\_PIN, &P1OUT ; LCD1202\_SELECT

bic.b #LCD1202\_SCLK\_PIN | LCD1202\_MOSI\_PIN, &P1SEL ; Enable I/O function by clearing

bic.b #LCD1202\_SCLK\_PIN | LCD1202\_MOSI\_PIN, &P1SEL2 ; LCD1202\_DISABLE\_HARDWARE\_SPI

bit.b #01h, R12

jeq cmd

bis.b #LCD1202\_MOSI\_PIN, &P1OUT ; LCD1202\_MOSI\_LO

jmp clock

cmd:

bic.b #LCD1202\_MOSI\_PIN, &P1OUT ; LCD1202\_MOSI\_HIGH

clock:

bis.b #LCD1202\_SCLK\_PIN, &P1OUT ; LCD1202\_CLOCK positive edge

nop

bic.b #LCD1202\_SCLK\_PIN, &P1OUT ; negative edge

bis.b #LCD1202\_SCLK\_PIN | LCD1202\_MOSI\_PIN, &P1SEL ; LCD1202\_ENABLE\_HARDWARE\_SPI;

bis.b #LCD1202\_SCLK\_PIN | LCD1202\_MOSI\_PIN, &P1SEL2 ;

mov.b R13, UCB0TXBUF

pollSPI:

bit.b #UCBUSY, &UCB0STAT

jz pollSPI ; while (UCB0STAT & UCBUSY);

bis.b #LCD1202\_CS\_PIN, &P1OUT ; LCD1202\_DESELECT

pop R13

pop R12

ret

**Configure the Nokia1202 display**

The following code configures the Nokia 1202 display to display pixels. The code consists of two main areas. The first section holds the reset line low and then high for a specific length of time. You will measure the duration of the reset pulse later in the lab.   
  
The second section sends a sequence of commands to the Nokia 1202 display. Your task is to use the the information on page 42 (and beyond) of the STE2007 technical document to decode the symbolic constants moved into register R13 on the lines marked with "DECODE HERE".

;-------------------------------------------------------------------------------

; Name: initNokia 68(rows)x92(columns)

; Inputs: none

; Outputs: none

; Purpose: Reset and initialize the Nokia Display

;-------------------------------------------------------------------------------

initNokia:

push R12

push R13

bis.b #LCD1202\_CS\_PIN, &P1OUT

;-------------------------------------------------------------------------------

; Measure the time that the RESET\_PIN is held low by the delayNokiaResetLow loop

bic.b #LCD1202\_RESET\_PIN, &P2OUT

mov #0FFFFh, R12

delayNokiaResetLow:

dec R12

jne delayNokiaResetLow

bis.b #LCD1202\_RESET\_PIN, &P2OUT

;-------------------------------------------------------------------------------

mov #0FFFFh, R12

delayNokiaResetHigh:

dec R12

jne delayNokiaResetHigh

bic.b #LCD1202\_CS\_PIN, &P1OUT

; First write seems to come out a bit garbled - not sure cause

; but it can't hurt to write a reset command twice

mov #NOKIA\_CMD, R12

mov #STE2007\_RESET, R13 ; DECODE HERE

call #writeNokiaByte

mov #NOKIA\_CMD, R12

mov #STE2007\_RESET, R13

call #writeNokiaByte

mov #NOKIA\_CMD, R12

mov #STE2007\_DISPLAYALLPOINTSOFF, R13 ; DECODE HERE

call #writeNokiaByte

mov #NOKIA\_CMD, R12

mov #STE2007\_POWERCONTROL | STE2007\_POWERCTRL\_ALL\_ON, R13 ; DECODE HERE

call #writeNokiaByte

mov #NOKIA\_CMD, R12

mov #STE2007\_DISPLAYNORMAL, R13 ; DECODE HERE

call #writeNokiaByte

mov #NOKIA\_CMD, R12

mov #STE2007\_DISPLAYON, R13 ; DECODE HERE

call #writeNokiaByte

pop R13

pop R12

ret

Complete the table below. To answer this question you will have to use some common sense in decoding the meaning of the symbolic constants.

| **Symbolic Constant** | **Hex** | **Function** |
| --- | --- | --- |
| #STE2007\_RESET | E2 | Initializes the driver |
| #STE2007\_DISPLAYALLPOINTSOFF | A4 | Forces the display points OFF |
| #STE2007\_POWERCONTROL | 28 | Sets a power supply function on |
| #STE2007\_POWERCTRL\_ALL\_ON | 2F | Sets all power supply functions on |
| #STE2007\_DISPLAYNORMAL | A6 | When the DDRAM data is high, the LCD is on |
| #STE2007\_DISPLAYON | AF | Turns the display on |