1. (25 points) Software

A. (10 points) Consider the following code segment. Show the content of the stack at the moment before the statement in line 6 is executed. Mark the top of the stack. Assume that the allocation on the stack proceeds in the program order and that all variables listed are allocated on the stack.

```c
1   int x = 5;  // an integer x
2   int *p_x;   // a pointer to int
3   int y;     // an integer y (uninitialized)
4   long int z = 0x12345678; // long integer z
5
6   p_x = &x;        // p_x points to x
7   *p_x = 9;        // p_x points to x
8   y = *p_x + 2;
```

B. (5 points) Show the content of the stack at the moment after the statement in line 8 is executed.

C. (10 points) For each statement from line 6 to line 8 show its assembly language implementation. Note: use the SP and indexed addressing mode to access the operands on the stack.

<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
<th>Assembly</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>p_x = &amp;x;        // p_x points to x</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>*p_x = 9;        // p_x points to x</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>y = *p_x + 2;    // p_x points to x</td>
<td></td>
</tr>
</tbody>
</table>
Problem #2 (25 points) Microcontroller MSP430 is using 32KHz crystal connected to LFXT1 Oscillator, 8MHz crystal connected to XT2 Oscillator, and 3V power supply. See Appendix (pages 6-9) for necessary information.

Datasheet specifications:
- \( \frac{f_{Rsel+1}}{f_{Rsel}} = 1.65 \),
- \( \frac{f_{DCO+1}}{f_{DCO}} = 1.12 \),
- DCOR: use internal \( R_{osc} \)

Set the following modes of operation (If the bit can be either 0 or 1, put X):

**A. (5 points)** processor clock (MCLK) to 8MHz, ACLK to 8KHz, SMCLK to 750 KHz.

<table>
<thead>
<tr>
<th>BCSCTL1:</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>XT2Off</td>
<td>XT</td>
<td>DIVA.1</td>
<td>DIVA.0</td>
<td>XT5V</td>
<td>Rsel2</td>
<td>Rsel1</td>
<td>Rsel0</td>
<td></td>
</tr>
<tr>
<td>BCSCTL2:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SELM.1</td>
<td>SELM.0</td>
<td>DIVM.1</td>
<td>DIVM.0</td>
<td>SELS</td>
<td>DIVS.1</td>
<td>DIVS.0</td>
<td>DCOR</td>
<td></td>
</tr>
<tr>
<td>DCOCTL:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DCO.2</td>
<td>DCO.1</td>
<td>DCO.0</td>
<td>MOD.4</td>
<td>MOD.3</td>
<td>MOD.2</td>
<td>MOD.1</td>
<td>MOD.0</td>
<td></td>
</tr>
</tbody>
</table>

**B. (5 points)** processor clock to 840KHz, SMCLK to 420KHz, and ACLK to 32KHz.

<table>
<thead>
<tr>
<th>BCSCTL1:</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>XT2Off</td>
<td>XT</td>
<td>DIVA.1</td>
<td>DIVA.0</td>
<td>XT5V</td>
<td>Rsel2</td>
<td>Rsel1</td>
<td>Rsel0</td>
<td></td>
</tr>
<tr>
<td>BCSCTL2:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SELM.1</td>
<td>SELM.0</td>
<td>DIVM.1</td>
<td>DIVM.0</td>
<td>SELS</td>
<td>DIVS.1</td>
<td>DIVS.0</td>
<td>DCOR</td>
<td></td>
</tr>
<tr>
<td>DCOCTL:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DCO.2</td>
<td>DCO.1</td>
<td>DCO.0</td>
<td>MOD.4</td>
<td>MOD.3</td>
<td>MOD.2</td>
<td>MOD.1</td>
<td>MOD.0</td>
<td></td>
</tr>
</tbody>
</table>
C. (5 points) processor clock to 2MHz, ACLK to 32 KHz, SMCLK to 1MHz.

BCSCTL1: _____
- XT2Off
- XTS
- DIVA.1
- DIVA.0
- XT5V
- Rsel2
- Rsel1
- Rsel0

BCSCTL2: _____
- SELM.1
- SELM.0
- DIVM.1
- DIVM.0
- SELS
- DIVS.1
- DIVS.0
- DCOR

DCOCTL: _____
- DCO.2
- DCO.1
- DCO.0
- MOD.4
- MOD.3
- MOD.2
- MOD.1
- MOD.0

D. (5 points) processor clock and SMCLK to 787 KHz and ACLK to 32KHz.

BCSCTL1: _____
- XT2Off
- XTS
- DIVA.1
- DIVA.0
- XT5V
- Rsel2
- Rsel1
- Rsel0

BCSCTL2: _____
- SELM.1
- SELM.0
- DIVM.1
- DIVM.0
- SELS
- DIVS.1
- DIVS.0
- DCOR

DCOCTL: _____
- DCO.2
- DCO.1
- DCO.0
- MOD.4
- MOD.3
- MOD.2
- MOD.1
- MOD.0

E. (5 points) What should be the value of MOD if MCLK needs to be set to 1.5MHz for the system that doesn’t use external oscillators (uses only DCO)? Give values for Rsel, DCO, and MOD. Show how you came up with the result.

[Hint: use the formula $T = ((32 - MOD) \times T_{DCO} + MOD \times T_{DCO+1}) / 32$]
Problem #3 (25 points) Interrupts
A. (10 points) Let us assume a P2.2 port pin is configured as interrupt request input. Describe all steps involved in handling an interrupt request, from the moment a request is received (rising edge detected on P2.2) until the return from the corresponding interrupt handling routine. Be precise, and describe what is done in hardware and what is done in software.

B. (5 points) What is interrupt nesting? Describe how the MSP430 deals with interrupt nesting.

C. (5 points) What is interrupt selective masking? Describe how the MSP430 deals with selective masking.

D. (5 points) How do we enter and exit low-power modes in the MSP430? Be specific.
4. (25 points, TimerA, Watchdog Timer)  
**A. (10 points)** Consider the following program. The main program is an infinite loop. The CPU enters a low power mode 3 (CPU, MCLK, SMCLK, and the DCO oscillator are all turned off, ACLK stays on). The WDT interrupt service routine should wake the CPU up every 1000 ms (1s). The CPU then pulses a LED connected at the P1.0 (turns it on, keeps it on for some time, and then turns it off) and then enters the LPM3 mode again. Describe what needs to be done at the beginning of the main routine and what needs to be done in the WDT interrupt service routine. Note: answer in plain English, no code is required. Assume that a watch crystal of 32768 Hz is connected to LFXT1. The watchdog timer taps are $2^9$, $2^{13}$, $2^{15}$. Could you estimate how long the LED will be on?

```c
void main(void) {
    // initialization (put your description below)
    //
    //
    while(1) {
        int i;
        _BIS_SR(LPM3_bits + GIE);           // Enter LPM3
        P1OUT |= 0x01;                      // Set P1.0 LED on
        for (i = 10000; i>0; i--);           // Delay
        P1OUT &= ~0x01;                     // Clear P1.0 LED off
    }
}
```

Could you estimate how long the LED will be on?

**B. (15 points)** You would like to generate two pulse-width modulated (PWM) signals P1 (75% of duty cycle) and P2 (50% of duty cycle), with frequency of 1 KHz. Assume that an external high-frequency resonator of 8 MHz is connected on LFXT1. Can you do this using TimerA? If yes, could you describe a TimerA configuration that will carry out signal generation? Note: use English and waveforms to describe your solution.

```
P1

P2

0ms 0.25ms 0.5ms 0.75ms 1ms 1.25ms 1.5ms 1.75ms
```
Appendix

<table>
<thead>
<tr>
<th>PARAMETER</th>
<th>TEST CONDITIONS</th>
<th>MIN</th>
<th>NOM</th>
<th>MAX</th>
<th>UNIT</th>
</tr>
</thead>
<tbody>
<tr>
<td>R_{sel} = 0, DCO = 3, MOD = 0, DCOR = 0, T_A = 25°C</td>
<td>V_{CC} = 2.2 V</td>
<td>0.08</td>
<td>0.12</td>
<td>0.15</td>
<td>MHz</td>
</tr>
<tr>
<td>R_{sel} = 1, DCO = 3, MOD = 0, DCOR = 0, T_A = 25°C</td>
<td>V_{CC} = 2.2 V</td>
<td>0.14</td>
<td>0.19</td>
<td>0.23</td>
<td>MHz</td>
</tr>
<tr>
<td>R_{sel} = 2, DCO = 3, MOD = 0, DCOR = 0, T_A = 25°C</td>
<td>V_{CC} = 2.2 V</td>
<td>0.22</td>
<td>0.30</td>
<td>0.36</td>
<td>MHz</td>
</tr>
<tr>
<td>R_{sel} = 3, DCO = 3, MOD = 0, DCOR = 0, T_A = 25°C</td>
<td>V_{CC} = 2.2 V</td>
<td>0.37</td>
<td>0.49</td>
<td>0.59</td>
<td>MHz</td>
</tr>
<tr>
<td>R_{sel} = 4, DCO = 3, MOD = 0, DCOR = 0, T_A = 25°C</td>
<td>V_{CC} = 2.2 V</td>
<td>0.61</td>
<td>0.77</td>
<td>0.93</td>
<td>MHz</td>
</tr>
<tr>
<td>R_{sel} = 5, DCO = 3, MOD = 0, DCOR = 0, T_A = 25°C</td>
<td>V_{CC} = 3 V</td>
<td>0.14</td>
<td>0.18</td>
<td>0.22</td>
<td>MHz</td>
</tr>
<tr>
<td>R_{ref}</td>
<td>V_{CC} = 2.2 V</td>
<td>0.13</td>
<td>0.16</td>
<td>0.16</td>
<td>MHz</td>
</tr>
<tr>
<td>V_{ref}</td>
<td>V_{CC} = 2.2 V</td>
<td>0.17</td>
<td>0.21</td>
<td>0.25</td>
<td>MHz</td>
</tr>
<tr>
<td>V_{ref}</td>
<td>V_{CC} = 2.2 V</td>
<td>0.31</td>
<td>0.36</td>
<td>0.40</td>
<td>MHz</td>
</tr>
<tr>
<td>V_{ref}</td>
<td>V_{CC} = 3 V</td>
<td>0.36</td>
<td>0.40</td>
<td>0.43</td>
<td>MHz</td>
</tr>
</tbody>
</table>

D1

The DCO generator is connected to pin P2.5/Rosc if DCOR control bit is set.
The port pin P2.5/Rosc is selected if DCOR control bit is reset (initial state).
7.5 Basic Clock Module Control Registers

The Basic Clock Module is configured using control registers DCCCTL, BCSCTL1, and BCSCTL2, and four bits from the CPU status register: SCG1, SCG0, OscOff, and CPUOFF. User software can modify these control registers from their default condition at any time. The Basic Clock Module control registers are located in the byte-wide peripheral map and should be accessed with byte (B) instructions.

<table>
<thead>
<tr>
<th>Register</th>
<th>Short Form</th>
<th>Register Type</th>
<th>Address</th>
<th>Initial State</th>
</tr>
</thead>
<tbody>
<tr>
<td>DCO control register</td>
<td>DCOCTL</td>
<td>Read/write</td>
<td>056h</td>
<td>060h</td>
</tr>
<tr>
<td>Basic clock system control 1</td>
<td>BCSCTL1</td>
<td>Read/write</td>
<td>057h</td>
<td>084h</td>
</tr>
<tr>
<td>Basic clock system control 1</td>
<td>BCSCTL2</td>
<td>Read/write</td>
<td>058h</td>
<td>reset</td>
</tr>
</tbody>
</table>

7.5.1 Digitally-Controlled Oscillator (DCO) Clock-Frequency Control

DCOCTL is loaded with a value of 060h with a valid PUC condition.

<table>
<thead>
<tr>
<th>DCOCTL</th>
<th>7</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>056h</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

MOD0..MOD4: The MOD constant defines how often the discrete frequency fDCO+1 is used within a period of 32 DCOCLK cycles. During the remaining clock cycles (32–MOD) the discrete frequency fDCO is used. When the DCO constant is set to seven, no modulation is possible since the highest feasible frequency has then been selected.

DCO0..DCO2: The DCO constant defines which one of the eight discrete frequencies is selected. The frequency is defined by the current injected into the dc generator.
7.5.2 Oscillator and Clock Control Register

BCSCTL1 is affected by a valid PUC or POR condition.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>X12Off</td>
</tr>
<tr>
<td>6</td>
<td>XTS</td>
</tr>
<tr>
<td>5</td>
<td>DIVA1</td>
</tr>
<tr>
<td>4</td>
<td>DIVA 0</td>
</tr>
<tr>
<td>3</td>
<td>XT5V</td>
</tr>
<tr>
<td>2</td>
<td>Rsel 2</td>
</tr>
<tr>
<td>1</td>
<td>Rsel 1</td>
</tr>
<tr>
<td>0</td>
<td>Rsel 0</td>
</tr>
</tbody>
</table>

**Bit0 to Bit2:** The internal resistor is selected in eight different steps.

**Rsel.0 to Rsel.2:** The value of the resistor defines the nominal frequency. The lowest nominal frequency is selected by setting Rsel=0.

**Bit3, XT5V:** XT5V should always be reset.

**Bit4 to Bit5:** The selected source for ACLK is divided by:

- DIVA = 0: 1
- DIVA = 1: 2
- DIVA = 2: 4
- DIVA = 3: 8

**Bit6, XTS:** The LFXT1 oscillator operates with a low-frequency clock crystal or with a high-frequency crystal:

- XTS = 0: The low-frequency oscillator is selected.
- XTS = 1: The high-frequency oscillator is selected.

The oscillator selection must meet the external crystal’s operating condition.

**Bit7, XT2Off:** The XT2 oscillator is switched on or off:

- XT2Off = 0: the oscillator is on
- XT2Off = 1: the oscillator is off if it is not used for MCLK or SMCLK

BCSCTL2 is affected by a valid PUC or POR condition.
<table>
<thead>
<tr>
<th>Bit 7, SELM</th>
<th>SELM.0</th>
<th>SELM.1</th>
<th>DIVM.1</th>
<th>DIVM.0</th>
<th>SELS</th>
<th>DIVS.1</th>
<th>DIVS.0</th>
<th>DCOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>0</td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-0</td>
<td>rw-0</td>
<td>rw-0</td>
<td>rw-0</td>
</tr>
</tbody>
</table>

Bit 0, DCOR: The DCOR bit selects the resistor for injecting current into the dc generator. Based on this current, the oscillator operates if activated.
- **DCOR = 0**: Internal resistor on, the oscillator can operate. The fail-safe mode is on.
- **DCOR = 1**: Internal resistor off, the current must be injected externally if the DCO output drives any clock using the DCOCLK.

Bit 1, Bit 2: The selected source for SMCLK is divided by:

- **DIVS.1**, **DIVS.0**
  - DIVS = 0: 1
  - DIVS = 1: 2
  - DIVS = 2: 4
  - DIVS = 3: 8

Bit 3, SELS: Selects the source for generating SMCLK:
- SELS = 0: Use the DCOCLK
- SELS = 1: Use the XT2CLK signal (in three-oscillator systems) or the LFXT1CLK signal (in two-oscillator systems)

Bit 4, Bit 5: The selected source for MCLK is divided by:

- **DIVM.0**, **DIVM.1**
  - DIVM = 0: 1
  - DIVM = 1: 2
  - DIVM = 2: 4
  - DIVM = 3: 8

Bit 6, Bit 7: Selects the source for generating MCLK:
- **SELM.0**, **SELM.1**
  - SELM = 0: Use the DCOCLK
  - SELM = 1: Use the DCOCLK
  - SELM = 2: Use the XT2CLK (x13x and x14x devices) or the LFXT1CLK (x11xx and x12xx devices)
  - SELM = 3: Use the LFXT1CLK