• Open book/open notes, 90-minute exam.
• No electronic devices are required or permitted.
• All work and solutions are to be written on the exam where appropriate.

Point System (for instructor and TA use only)

<table>
<thead>
<tr>
<th>Page</th>
<th>Points</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Page 2</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td>Page 3</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>Page 4</td>
<td>22</td>
<td></td>
</tr>
<tr>
<td>Page 5</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Page 6</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Page 7</td>
<td>33</td>
<td></td>
</tr>
<tr>
<td>I Brought My Etched Board!</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>TOTAL</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

TOTAL __________________ out of 102

Grade Review Information: (NOTE: deadline of request for grade review is the day the exam is returned.)


For the following problems you are given a microprocessor with a multiplexed address (A15:0) and data (D15:0) bus. The control bus consists of an AddressStrobe (+AS), DataStrobe (-DS) and a Read/Write (R/-W) signal. See the timing below.

uP Write/Read Cycle Timing Diagram

1. Show the required circuitry to de-multiplex the address and data bus. Assume a CPLD is available and label all signals. (6 pt.)

2. Why did the manufacturer of this device multiplex the address and data bus? Best answer = most points! (2 pt.)

3. The uP is designed such that after reset, the Program Counter is automatically loaded with the first address of the highest 8K block of memory. Thus this section of memory should be filled with an 8Kx16 EPROM or 8Kx16 EEPROM. What is the lowest address in the highest 8K block of memory in hex? (2 pt.)

4. A student building a computer unfortunately only has (2) 4Kx8 EPROMs. Assuming that there will be only one image of this block of non-volatile memory in the system, what is the logic equation required to decode this block of memory?

\[
/\text{EPROM\_CE} = \quad \text{(4 pt.)}
\]
5. The uP has several **internal registers** in the memory range 0-FFF and C000-CFFF Hex. Thus this range is not available for external devices. However from 1000 Hex to BFFF Hex, the memory map is open. Assuming that you have as many **32Kx16 SRAMs** are needed, fill the entire open area with SRAM. Show the **SRAMs, signal connections** and **required decode circuitry** below. Use as few SRAMs as possible to receive the most points. See the SRAM timings below for more information on the SRAM connections. (12 pt.)

---

**uP Specifications & Requirements – WRITE Cycle**

- R/-W & Address Valid to +AS True = 30 nsec
- Address hold time = 20 nsec
- +AS True to +AS False = 30 nsec
- +AS False to -DS True = 30 nsec
- -DS True to -DS False = 30 min/35 max
- Data Setup = 12 min/15 max
- Data Hold = 8 min/10 max
- R/-W Hold after -DS False = 15

**SRAM Specifications & Requirements – Write Cycle**

- R/-W & Address True to –CS True = 20/no maximum
- -CE Width = 10/no maximum
- *** -OE False = don’t care ***
- Data Setup time = 10/no maximum
- Hold time = 0/no maximum

**uP Specifications & Requirements – READ Cycle**

- R/-W & Address Valid to +AS True = 30 nsec
- Address hold time = 20 nsec
- +AS True to +AS False = 30 nsec
- +AS False to -DS True = 30 nsec
- -DS True to -DS False = 30 min/35 max
- Data Setup = 3 min/no maximum
- Data Hold = 5 min/15 maximum
- R/-W Hold after -DS False = 15

**SRAM Specifications & Requirements – Read Cycle**

- R/-W & Address True to –CS True = 20/no maximum
- -CE Width = 10/no maximum
- R/-W & Address True to -OE True = 20/no maximum
- R/-W, –CE, –OE False (whichever first) Hold time = 2/3
- Enable SRAM Memory Access time = 12/16
6. Using the uP bus and SRAM timings from the previous page, compute the range of possible propagation delays that are possible for the address decode CPLD. Show all work below for partial credit purposes. (6 pts)

\[
\text{nsec} \geq t_{\text{cpld}} \geq \text{nsec}
\]

7. For problem #6, what is the CPLD propagation delay range if R/W Hold after -DS False = 0 on the write cycle and 40 nsec on the read cycle? Show all work below for partial credit purposes. (6 pt.)

\[
\text{nsec} \geq t_{\text{cpld}} \geq \text{nsec}
\]

7.1 Why does the uP Read Cycle have 15 nsec for the maximum Data Hold specification? (2 pt.)

A student has been given an asynchronous serial LCD device that displays characters received via a receive (RX) pin on the LCD. They would like to interface this serial LCD to their DSP but have found that their Serial Communication Interfaces (UARTS) are not working. Instead they decide to use GPIO 0 to send out the data to the serial LCD RX pin. Given the following specifications, answer the questions that follow.

Serial Communication should be done via flag polling with Timer 2. The LCD communication protocol is 5000 Baud, 1 start bit, 1 stop bit, 8 data bits and no parity. *** DSP External Crystal is 25 MHz. ****

8. Before polling can begin, write the assembly code to set up all the registers to their required values. Initialize only the registers that are required in this application; you will lose points for added superfluous code. (10 pt.)
9. Using **Timer 2** and **polling**, create an assembly routine that sends out a character serially to the LCD via **bit banging GPIO 0**. Assume that the character will be passed to the routine via AL. (8 pt.)

SEND_Char_to_LCD:

```

```

LRET
You are given a new Keyboard Controller IC that is used on a laptop motherboard to interface to a low cost embedded laptop keyboard. The device allows a microprocessor to read in a character every time a key is pressed. The interface between the device and a uP is very similar to that used with an SRAM. The block diagram for the device is shown below:

### Register Definitions:

**Address**

- **0** (read only) => Key Stroke Buffer1: 1st Character (key stroke) received.
- **1** (read only) => Key Stroke Buffer 2: 2nd Character (key stroke) received.
- **2** (read only) => Key Stroke Buffer 3: 3rd Character (key stroke) received.
- **3** (read only) => Key Stroke Buffer 4: 4th Character (key stroke) received.
- **4** (read only) => Key Stroke Buffer 5: 5th Character (key stroke) received.
- **5** (read only) => Key Stroke Buffer 6: 6th Character (key stroke) received.
- **6** (read only) => Key Stroke Buffer 7: 7th Character (key stroke) received.
- **7** (read only) => Key Stroke Buffer 8: 8th Character (key stroke) received.

- **8** (read/write) => Keyboard Control Register: Bit 7 = Interrupt Enable, Enabled = 1, Disabled = 0.
  Bit 6 = Back light
  Bit 5:0 = Scan Rate

- **9** (read/write) => Key Stroke Flag Register: Bits 7:0 = Characters Flags for Key Stroke Buffers 8:1.
  i.e. 1st Char received, Bit 0 = 1, 2nd Char received, Bit 1 = 1,...
  Flags are automatically cleared when a character is read.

A (read/write) => Keyboard Interrupt/Overrun Reg: Bit 7 = Int. Pending Flag, If int. is enabled, any time a char is received the interrupt pending flag will be set. **You must write a ‘1’ to clear this flag.**

Bit 6 = key strokes have overrun the buffer flag.

10. Assuming that we can directly interface this to our DSP, show the connections below such that it is mapped to Zone 0. This should be a glue-less interface where there are only wires required; no CPLD is necessary.

**Write the DSP Signal Connections Below**

- **Key Board Controller A3:0** => _______________________________ (1 pt.)
- **Key Board Controller D7:0** => _______________________________ (1 pt.)
- **Key Board Controller -CE** => _______________________________ (3 pt.)
- **Key Board Controller R/-W** => _______________________________ (3 pt.)
11. How many images of this device will we have in Zone 0? (3 pt.)

12. What are the highest and lowest image address ranges for this device? (4 pt.)

14. Assume that the Key Board Controller's interrupt signal is now attached to the DSP's GPIO1 pin that needs to be configured to trigger PIE external interrupt #2 (XINT2) when pulsed low. Write the pseudo code steps necessary to configure and enable this interrupt and the Key Board Controller. Be as detailed and complete as possible for maximum points awarded. (14 pt.)

1. ________________________________________________________________
2. ________________________________________________________________
3. ________________________________________________________________
4. ________________________________________________________________
5. ________________________________________________________________
6. ________________________________________________________________
7. ________________________________________________________________
8. ________________________________________________________________
9. ________________________________________________________________

15. Write the pseudo code necessary to read in a character or characters every time an interrupt is fired. Also write the DSP address value of the Key Stroke Controller register you are reading. (7 pt.)

1. ________________________________________________________________
2. ________________________________________________________________
3. ________________________________________________________________
4. ________________________________________________________________
5. ________________________________________________________________
6. ________________________________________________________________
7. ________________________________________________________________

16. Briefly explain how your interrupt routine could be modified above to let your main program know that a character has been received and is ready to be processed. (3 pt.)

17. Based upon your extensive knowledge of scanning keyboards for a pressed key, what manufacturer's error has been made in the "G4 Key Board Controller" block diagram on the previous page? (2 pt.)