THE AUSTRALIAN NATIONAL UNIVERSITY

First Semester Supp. Exam 2009

ENGN3213
Digital Systems and Microprocessors

Writing Period: 3 hours
Study Period: 15 minutes duration

Permitted Materials: Pens

Total 100 Marks (100% of Subject)

You must attempt each of the 5 questions
Q1. General Knowledge (20 marks; 20% Total)

(a) Describe how the PICOBLAZE microcontroller executes a subroutine call. (3 marks)

(b) Draw the truth table of an SR latch shown below. Show all inputs and outputs. (6 marks)

![SR latch diagram]

(c) Explain using code snippets how a C-program can debounce a switch. (4 marks)

(d) Traces of the clock and D-input of a D-flip-flop are shown in the following figure.

(1) Explain the difference between a sequential and a combinational circuit. (2 marks)

(2) Redraw the traces in the figure in your script books and show the resulting output Q as a function of time (5 marks).

![Clock, D, and Q traces diagram]
Q2. Arithmetic (20 marks; 20% Total)

Evaluate the following.

(a) \(1024_{10} = X_2, \Rightarrow X = ?\) (2 marks),
(b) \(113_{10} = Y_5, \Rightarrow Y = ?\) (2 marks),
(c) \(10110_2 + 10101_2 = ?\) (2 marks),
(d) \(1101002 - 0001112 = ?\) (2 marks),
(e) If \(\text{unsigned char}\ Y = X*X, \text{ where } X = 17\) what does \(Y\) evaluate to? (4 marks),
(f) Evaluate \(11.0011_2\) divided by \(11.000_2\) (4 marks),
(g) Evaluate \(111.00101_2\) divided by \(1.010_2\) (4 marks),

Q3. State machines and RTL systems (20 marks; 20% Total)

A robot driven by a simple finite state machine travels in a two dimensional labyrinth. The robot has two wheels and can either move forward or rotate. The robot FSM has two inputs. The forward (F) input is “1” when there are no obstructions ahead. The right (R) input is “1” when there is no obstruction to the right.

Initially the robot is in a state where it moves forward whenever \(F=1\). Then of \(F = 0\), the robot stops and rotates until \(F = 1\), and then it moves again. In this phase however the robot knows that it has probably reached a dead end and so it now continues until it finds an \(R=1\). Once it reaches this point, stops, rotates, randomly seeks \(F=1\) and then goes back to the initial state.
(a) What are the states of the controller? (2 marks).
(b) Draw the state diagram for the controller (3 marks).
(c) Draw a truth table showing the next state logic (you may use D flip flops) (3 marks).
(d) Draw a truth table showing the output logic (3 marks).
(e) Use Karnaugh maps to minimise the logic for each of (c) and (d) (3 marks).
(f) Draw the resultant circuit schematic of the controller (3 marks).
(g) Starting from the next state and output logic tables, implement the controller in
VERILOG HDL using standard coding practice (3 marks).
Q4. VERILOG and C (20 marks; 20% Total)

(1) Write VERILOG HDL to describe a device which samples an incoming bitstream and outputs the last eight samples of the bitstream at the positive edge of the system clock. (10 marks)

**HINT:** Let the incoming bitstream (denoted `inbits`) and the system clock (denoted `sysclk`) be inputs and the output (denoted `outbyte`) be the 8-bit output.

(2) Write the C-subroutine `store()` that fills the elements of the 4 element array `arr` with the values `a,b,c,d` in sequence. Carefully explain each line of code in the final program. (10 marks)

```c
int main() {
    int a = 1;
    int b = 2;
    int c = 3;
    int d = 4;

    int *arr = malloc(4);
    store(a, b, c, d, arr);

    return 0;
}
```
Q5. MU0 and PICOBLAZE Micrprocessors  (20 marks; 20% Total)

(1) Consider the following MU0 machine language program.

```
0004 3005 1005 7000 0004 0001
```

Using the figures on the last page answer the following.
(a) Describe in words what happens on each of the lines of the MU0 assembly language program (2 marks).
(b) What is the contents of the accumulator at the completion of execution (2 marks).
(c) Draw a timing diagram showing the contents of the program counter (PC), the instruction register (IR), the accumulator (ACC) and the CLOCK over the entire duration of execution of the program. Label the timing diagram with the trace values in hexadecimal. (10 marks).

(2) Describe line by line the function of the following PICOBLAZE assembly language program (6 marks).

```
CONSTANT counter_port, 04
NAME REG sA, interrupt_counter
start: LOAD interrupt_counter, 00
        ENABLE INTERRUPT
;
        ADDRESS 3B0
int routine: ADD interrupt_counter, 01
        OUTPUT interrupt_counter, counter_port
        RETURN ENABLE
;
        ADDRESS 3FF
        JUMP int routine
```
### FSM State Transition Table

<table>
<thead>
<tr>
<th>state</th>
<th>F[2:0]</th>
<th>Next state</th>
<th>IREn</th>
<th>PCEn</th>
<th>AccEn</th>
<th>M[1:0]</th>
<th>Xsel</th>
<th>Ysel</th>
<th>Acc</th>
<th>Run</th>
<th>Wes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>xxxx</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>10</td>
<td>0</td>
<td>x</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>000</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>xxx</td>
<td>1</td>
<td>x</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>001</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>xxx</td>
<td>1</td>
<td>x</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>000</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>00</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>011</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>11</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>100</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>00</td>
<td>0</td>
<td>x</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>101</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>00</td>
<td>0</td>
<td>x</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>110</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>xxx</td>
<td>0</td>
<td>x</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>111</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>xxx</td>
<td>0</td>
<td>x</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

**Notes:**
- N and Z are the Negative and Zero state of the Accumulator, respectively.
- (used to reduce the size of the table, as drawn)
- If a value is not going to be latched it doesn’t matter what it is!
  (e.g. ALU output for STO)
- STP operates by remaining in its evaluation state.

### MU0 Instruction Set

<table>
<thead>
<tr>
<th>F</th>
<th>Mnemonic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>LDA</td>
<td>Acc := [S]</td>
</tr>
<tr>
<td>1</td>
<td>STO</td>
<td>[S] := Acc</td>
</tr>
<tr>
<td>2</td>
<td>ADD</td>
<td>Acc := Acc + [S]</td>
</tr>
<tr>
<td>3</td>
<td>SUB</td>
<td>Acc := Acc − [S]</td>
</tr>
<tr>
<td>4</td>
<td>JMP</td>
<td>PC := S</td>
</tr>
<tr>
<td>5</td>
<td>JGE</td>
<td>If Acc &gt;= 0, PC := S</td>
</tr>
<tr>
<td>6</td>
<td>JNE</td>
<td>If Acc ≠ 0, PC := S</td>
</tr>
<tr>
<td>7</td>
<td>STP</td>
<td>Stop</td>
</tr>
</tbody>
</table>