UNIVERSITY OF TORONTO
FACULTY OF APPLIED SCIENCE AND ENGINEERING

ECE253F – Digital and Computer Systems
Final Examination

December 15, 2017 2:00pm - 4:30pm
Duration: 2.5 hours

Examiner: J. Anderson

Exam Type D: Examiner specified aids: One single sheet of letter size paper (8.5 x 11 inch), both sides may be used.

Calculator Type 4: No calculators or other electronic devices are allowed.

All questions are to be answered on the examination paper. There is one extra page at the end and you may use the back of a page. If you use more than the given space, please direct the marker to the appropriate page and indicate clearly on that page which question(s) you are answering there. It is your responsibility to make sure the marker can find your solution.

The number of marks for each question are indicated.

The examination has **20 pages**, including this one.

Last Name: _____ First Name: _____

Student Number: _____ UTORID: _____

MARKS

1	2	3	4	5	6	7	8	9	10	11	12	Total
/6	/6	/6	/6	/6	/10	/10	/8	/6	/12	/6	/4	/86

Question 1 [6 Marks]

Assuming that all numbers given below are *signed* integers, fill in the following table with the appropriate number conversions. If a number conversion is not possible, state the reason why. Hexadecimal and binary are to represent 6-bit 2's complement numbers.

6-bit 2's complement	decimal	hexadecimal
101111		
	-45	
		2E

Question 2 [6 Marks]

Trace the execution of the following ARM assembly-language program.

```
.global _start
_start:
   LDR R0, =N
   LDR R1, [R0]
   MOV R2, #0
LOOP:
   CMP R1, #0
   BEQ END
   MUL R3, R1, R1
   ADD R2, R2, R3
   SUB R1, #1
   B LOOP
END:
   B END
N:
   .word 3
.end
```

Show the contents of registers R1, R2, and R3 after the execution.

R1 =

R2 =

R3 =

Question 3 [6 Marks]

[1 mark] State the purpose of the link register in the ARM computer.

[1 mark] Write a single ARM instruction that initializes register R1 to the value 7.

[1 mark] Write a single ARM instruction that performs a logical-shift left by two on the value in register R3, and places the result in register R5.

[2 marks] Consider a scenario where the content of register R1 is the value 5. If the ARM instruction,

```
CMP R5, #0x6
```

is executed, show the values of the four condition-code bits, `N, C, V, Z`:

[1 mark] Write ARM-assembly language instructions that branch to the address label `FOO` if the content of register R6 is greater than the content of register R5.

Question 4 [6 Marks]

Trace the execution of the following ARM assembly-language program.

```
.global _start
_start:
   LDR R0, =N
   LDR R1, [R0]
   LDR R3, [R0,#4]
   MOV R2, #0
LOOP:
   CMP R2, #4
   BEQ EXITLOOP
   ASR R1, #8
   ADD R2, R2, #1
   B LOOP
EXITLOOP:
   EOR R3, R3, R1
END:
   B END
N:
.word 0x80000000, 0xAAAAAAAA
.end
```

Show the contents of registers R1, R2 and R3 after the execution.

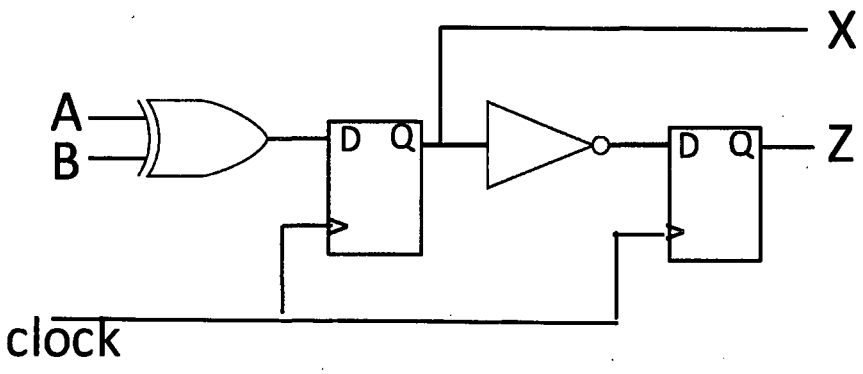
R1 =

R2 =

R3 =

Question 5 [6 Marks]

Consider the following sequential logic circuit, having inputs `A`, `B` and `clock`, and outputs `X` and `Y`:

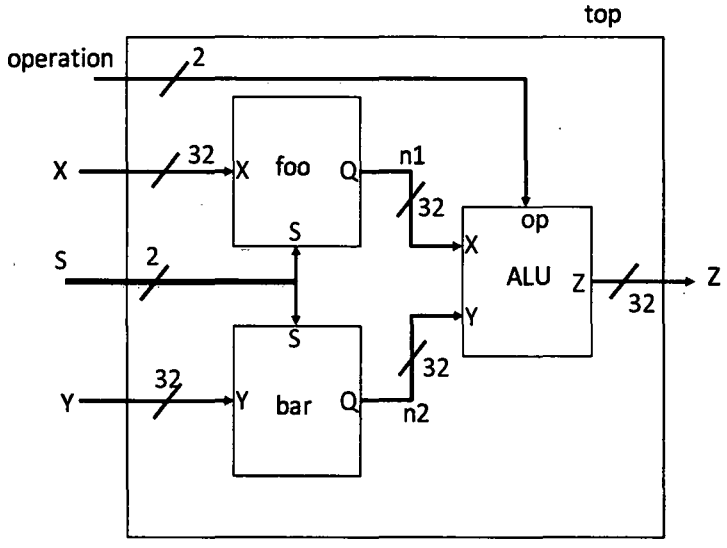


Write Verilog code for the logic circuit:

```
module q5circuit(
```

Question 6 [10 Marks]

Consider the following circuit block diagram:



[5 marks] Write structural Verilog for the circuit `top`, assuming that the modules `foo`, `bar`, and `ALU` are already specified, with the following port orderings: `foo(X,S,Q)`, `bar(Y,S,Q)`, `ALU(X,Y,op,Z)`. Use the signal and port names shown in the figure.

```
module top(
```

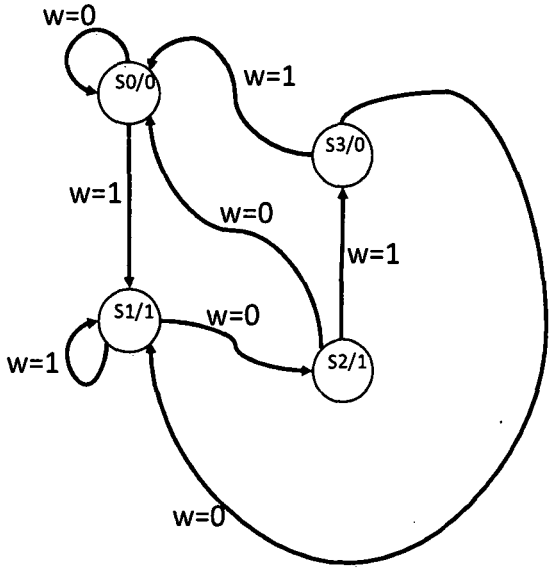
Question 6 continued ...

[5 marks] Write Verilog code for the module `ALU`. It should behave as follows: if `op` equals 00, $Z = X\ AND\ Y$; if `op` equals 01, $Z = X\ OR\ Y$; if `op` equals 10, $Z = X \oplus Y$; if `op` equals 11, $Z = X + Y$ (addition). It is a combinational circuit.

```
module ALU(
```

Question 7 [10 Marks]

Consider the following state diagram for a finite-state machine with input w and output z (z's value is shown within each state):



Given the following one-hot state codes for the states:

	y3	y2	y1	y0
S0	0	0	0	1
S1	0	0	1	0
S2	0	1	0	0
S3	1	0	0	0

[5 marks] Give the Boolean equations for the next-state logic and output logic:

Y0 =

Y1 =

Y2 =

Y3 =

z =

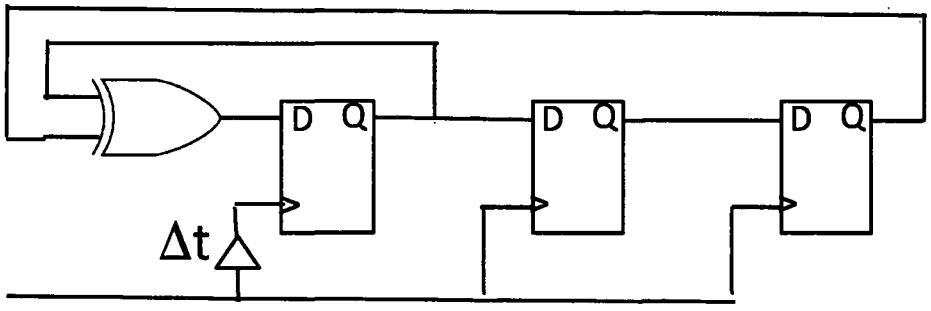
Question 7 continued ...

[5 marks] Write Verilog code for the FSM. Use the state-code assignments shown on the previous page. The FSM should also have an active-low asynchronous reset signal called `rstn` that causes the state to return to `S0`. The clock signal should be named *clk*.

```
module fsm(
```

Question 8 [8 Marks]

Timing analysis: Consider the following logic circuit, which is called a linear-feedback shift register (LFSR). Such circuits are used for random-number generation in hardware.



Assume the following circuit delays (wire delays are assumed to be 0):

	min	max
tEXOR	0.8ns	1.2ns
tCQ	0.8ns	1.0ns

tSU = 1ns (setup time)
tHOLD = 0.8ns (hold time)

2 marks Assuming $\Delta t = 0$, find the minimum clock period for the circuit, T_{min}. Show your work for full marks.

2 marks Assuming $\Delta t = 0$, is there a hold-time violation? Show your work for full marks.

4 marks Choose a value for Δt to minimize the clock period for the circuit, T_{min}, and such that there are no hold-time violations. State the value for Δt and the new T_{min}. Show your work for full marks.

Question 9 [6 Marks]

Consider the following C-language program fragment:

```
unsigned list[] = {1,5,7,13,0,0};

unsigned mul_Two_Elements(unsigned A, unsigned B)
{
 return A * B;
}

int main(void)
{
 unsigned *p = list;
 unsigned accum = 0;

 while (*p != 0)
 {
   accum += mul_Two_Elements(*p, *(p+1));
   p = p + 2;
 }

 ...
}
```

Write the equivalent functionality as an ARM assembly-language program, where the `mul_Two_Elements` function is implemented as a subroutine. Use register R0 for the subroutine return value. Use registers R0 and R1 to pass parameters to the subroutine (corresponding to `A` and `B`). The value of `accum` should be placed in register R5. The functionality of the `main` function should be located in memory at address label `_start`. The functionality of `mul_Two_Elements` should be located in memory at address label `MUL_TWO_ELEMENTS`. Fill in the skeleton code on the next page. For each variable in the C code, label the ARM register used (add labels to the C code above).

Question 9 continued ...

```
.global _start
LIST:
        .word 1,5,7,13,0,0
_start:




MUL_TWO_ELEMENTS:




.end
```

Question 10 [12 Marks]

Consider a four-channel analog-to-digital converter (ADC) as a memory-mapped I/O device connected to an ARM processor. The ADC converts up to four analog signals (e.g. from sensors) into digital values. The memory-mapped registers for this I/O device are connected to addresses 0xFF200200 to 0xFF20020C. Each channel produces an 8-bit value, accessible from the Data Register. The device also supports interrupts, using IRQ ID 91. Interrupts are generated for each channel when the Data Register value drops below the threshold value, defined for each channel in the Interrupt Thresholds register. An interrupt sets the interrupt-status bit for the corresponding channel in the Interrupt Status register. Each status bit can be cleared by writing a 1 to it.

Bits	31 ... 24	23 ... 16	15 ... 8	7 ... 4	3	2	1	0	
0xFF200200	Ch3 value	Ch2 Value	Ch1 value	Ch0 value					**Data Register**
0xFF200204	**Unused**								
0xFF200208	Ch3 Thresh	Ch2 Thresh	Ch1 Thresh	Ch0 Thresh					**Interrupt Thresholds**
0xFF20020C	**Unused**				S3	S2	S1	S0	**Interrupt Status Register**

Table 1: Analog-to-Digital Converter

[4 marks] Write the code to enable interrupts for this device for channels 1 and 2 with thresholds of 127 and 134 respectively, and to enable interrupts in the CPSR.

[4 marks] Write the section of the SERVICE_IRQ subroutine to determine which device is interrupting. Your code should call the ADC_ISR subroutine if the ADC is triggering the interrupt, and behave appropriately otherwise.

```
SERVICE_IRQ:
    PUSH R0-R5, LR
    LDR R4, =MPCORE_GIC_CPUIF
    LDR R5, [R4, #ICCIAR] // read the interrupt ID
```

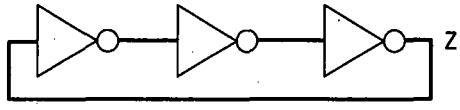
<YOUR CODE HERE>

```
EXIT_IRQ:
    STR R5, [R4, #ICCEOIR]
    POP R0-R5, LR
    SUBS PC, LR, #4
```

[4 marks] Write the `ADC_ISR` subroutine. This ISR should decrement by one the thresholds of any interrupting channels. Your code only needs to check the enabled channels, 1 and 2. Be sure you use the stack to store the previous contents of any registers used in your subroutine.

Question 11 [6 Marks]

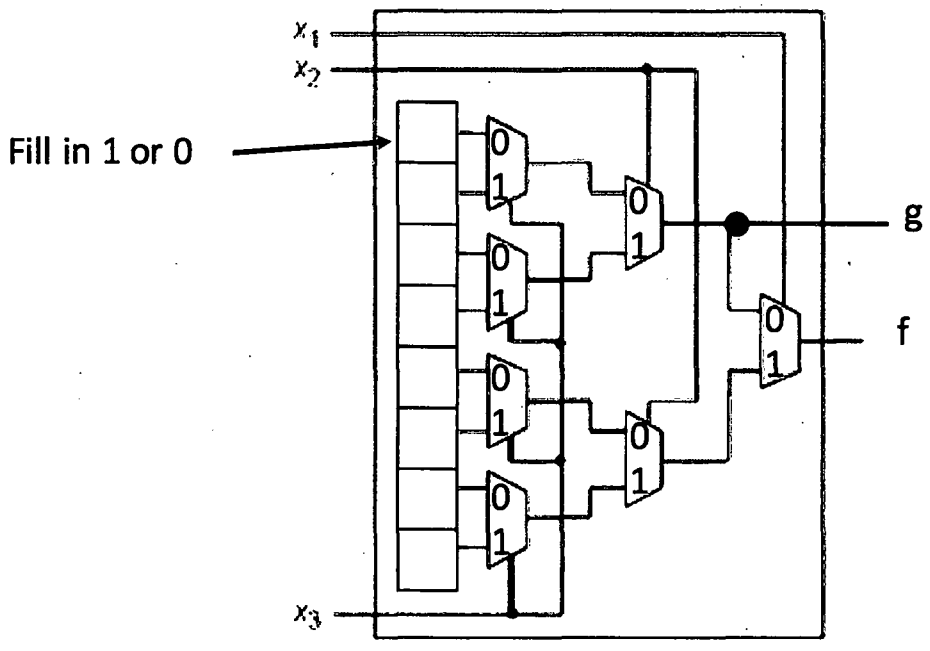
Consider the following circuit, which is called a ring oscillator. Assume each inverter has a delay of 1ns; the output Z toggles every 3ns and therefore, has a 6ns period.



Assume that inverters have a delay of 1ns; 2-input AND gates have a delay of 2ns; and 2-to-1 multiplexers have a delay of 2ns. Using inverters, 2-input AND gates, and 2-to-1 multiplexers, design a configurable ring-oscillator circuit with two inputs, w and S, and one output Z. If $w = 0$, output Z should be 0 (held constant). If $w = 1$, output Z oscillates with a period depending on S as follows: If $S = 0$, Z oscillates with a period of 10ns. If $S = 1$, Z oscillates with a period of 14ns. Draw the logic circuit.

Question 12 [4 Marks]

Consider the following dual-output look-up-table (LUT), similar to that used in modern commercial FPGAs. You are to implement the following two logic functions in the dual-output LUT: $g = x_2 \oplus x_3$ and $f = x_1x_2x_3 + (x_2 \oplus x_3)\overline{x_1}$. Fill in 0/1 values for the SRAM cells of the LUT to implement the two functions g and f.



[1 mark] BONUS: In the fictional Star Wars universe, name the individual who trained Luke Skywalker to be a Jedi Knight:

This page has been left blank intentionally. You may use it for answers to any questions.

This page has been left blank intentionally. You may use it for answers to any questions.