PRINT:

First Name: Kevin Last Name: Siu

Student Number: 995518462

University of Toronto
Faculty of Applied Science and Engineering

Final Examination – December 10, 2007

ECE253F – Digital and Computer Systems

Examiner: Parham Aarabi

1. There are 5 questions. Do ALL questions. The total number of marks is 100. The duration of the test is 2 hours 30 minutes.
2. ALL WORK IS TO BE DONE ON THESE SHEETS! Use the back of the pages if you need more space. Be sure to indicate clearly if your work continues elsewhere.
3. Open book – Exam Type D. Only authorized photocopies of copyrighted material area allowed.
4. All calculators permitted.
5. Please put your final solution in the appropriate spaces. A big space DOES NOT necessarily mean a long answer is required.
6. Place your student card on your desk.

1. (15 marks) 15
2. (20 marks) 16
3. (25 marks) 4+20
4. (15 marks) 13
5. (25 marks) 23

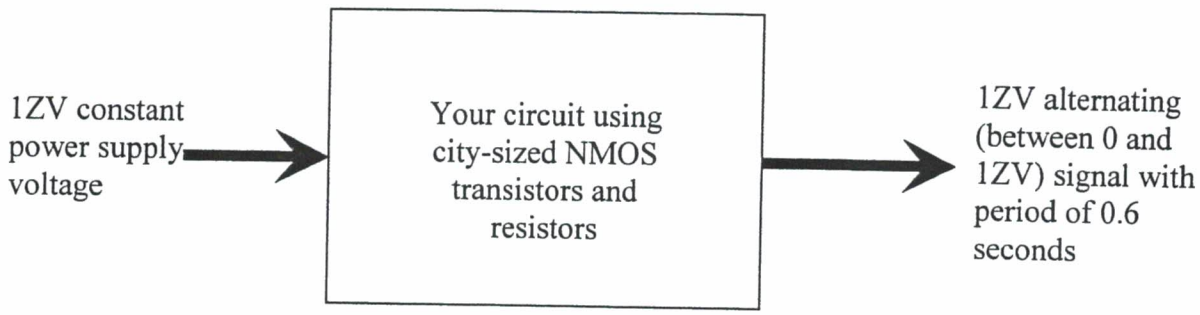
TOTAL (100 marks) 91

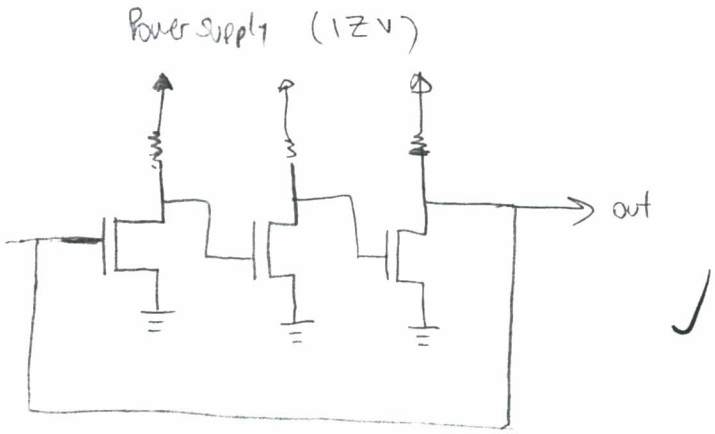
On November 9, you stole an alien space shuttle, learned the inner secrets of the alien computer, and destroyed the alien mother ship. Since then, the aliens have launched an all out assault against Earth. Given your success and experience (and your Eng. Sci. degree), you have been promoted to the rank of Chief Engineer at Area 52, a high-tech base in the mountains of Utah. This exam begins with NASA detecting a massive alien fleet approaching Earth.

1. (15 marks)

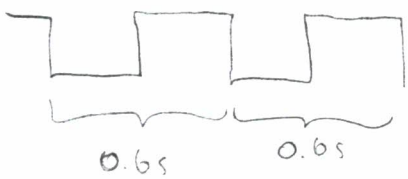
With the alien fleet just weeks away, you decide to use the specs of the alien shuttle shield to create an Earth-~~wide~~ shield. You realize that this requires a MASSIVE power supply. You come up with an ambitious plan to use Earth's inner core to power the shield. After drilling several holes directly to Earth's core, you are able to obtain a constant 1Zetta Volt (i.e. $1x10^{21}$) power supply voltage.

The alien shield, however, requires that the power coming into it be changing (oscillating) at 1.6666 Hz (i.e go up and down every 0.6 seconds). You decide to build giant pyramid-like NMOS transistors and resistors as part of an oscillating circuit. Assuming that each transistor has a delay of 0.1 seconds, show how you would connect these city-sized NMOS transistors and resistors to Earth's core and to the alien shield.





Ring oscillator with period of 0.6 s



2. (20 marks)

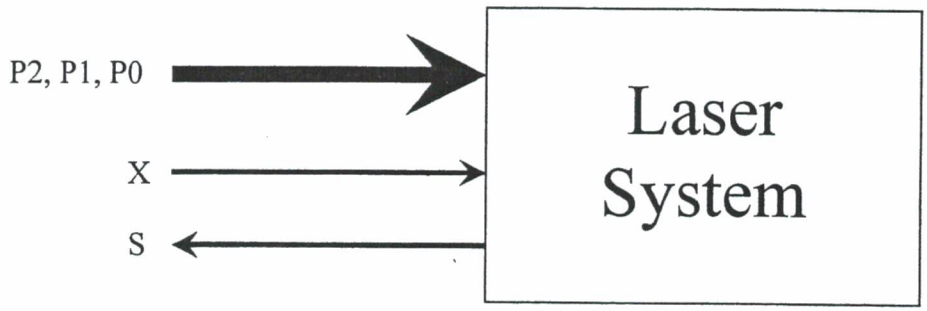
With the shield in place, you now need to equip F22 fighter jets with alien lasers. The F22s have an onboard NIOS computer with the usual bus signals that you would expect. The laser systems (copied from the lasers on the shuttle that you stole) have the following wires:

P2,P1,P0 = three bits representing a number between 0 and 7 that is the power output of the laser (7 being highest, 0 being lowest) – These bits come from the F22.

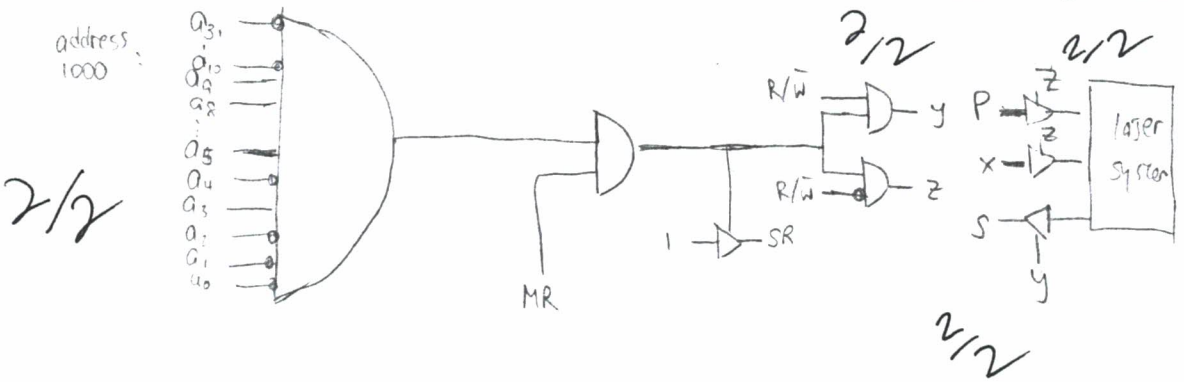
X = one bit on the positive edge of which the laser is fired – this comes from the F22.

S = one bit that will equal one when the laser power is stable. The laser's power cannot be changed or fired safely without the power first being stable – this comes from the laser system to the F22.

Address 1000 in decimal:
1000:
$2^9+2^8+2^7+2^6+2^5+2^3$
In binary:
1111 01000



a) (8 marks) Design an interface circuit between the F22 NIOS processor bus and the laser system above. Please use address 1000 to access (read and write) the laser system. You may use any number of DFFs, AND, OR, NOT, and TRI-STATE gates/buffers.



b) (12 marks) Write a NIOS assembly subroutine that, using the circuit of part a, powers up the laser from level 0 to level 7 and then fires. MAKE SURE that you increase the power level by only 1 step at a time, and after every power increase that you wait for the power to become stable before increasing again or firing.

INCREASE POWER ROUTINE

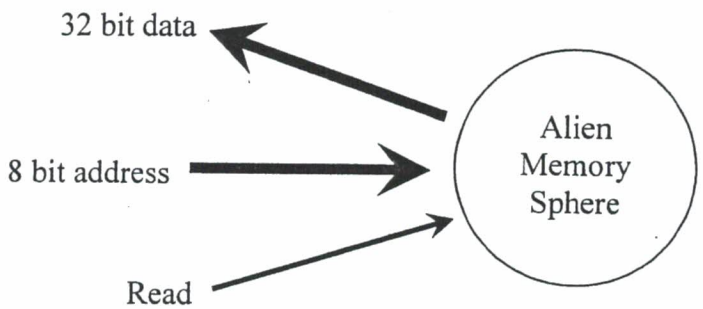
```
        addi r27,r29,-4   // save instr. location for subroutines
        stw  r31, 0(r27)

        movi r1, 0        //Power level from 0-7
        movi r2, POWERADDRESS //location of bits P2,P1,P0 } -2
        movi r3, STABLEADDRESS // location of bit S.
LOOP    stw  r1, 0(r2)    // set Power level
        addi r1, r1, 1    // increment power level.

POLL    ldw  r4, 0(r3)    // check stability of laser.
        movi r5, 1        // 1 = ready. 0 = not ready.
        bne  r4, r5, POLL // if not ready, go back to POLLing the laser.

        movi r5, 7 (ok)   // fully powered?
        bleu r1, r5, LOOP //if laser is not fully powered yet, go increment some more
→ fire!                   -2
        ldw  r31, 0(r27)  //restore return address from stack, etc.
        addi r29, r29, 4

        ret               //done subroutine.
```

3. (25 marks)

Amid a massive space battle between Earth's F22s and the alien fleet, you and your deputy sneak onboard one of the alien command ships with plans to hack into the alien computer.

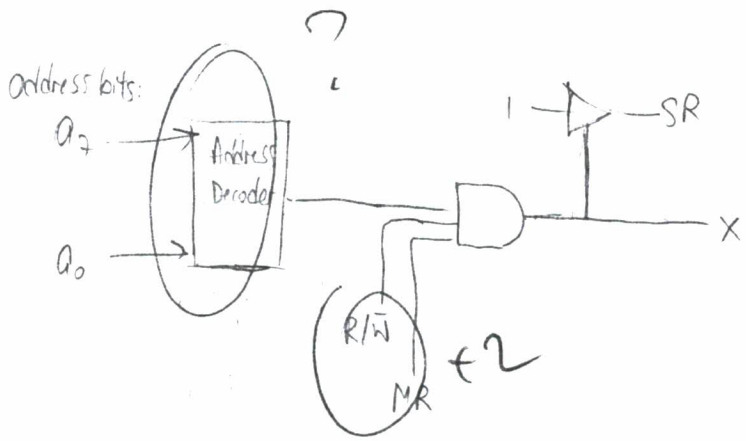
When you get to the computer room, you realize that there is an alien memory sphere with 256 memory locations. These locations are linked to all command ships, enabling you to hack into all ships.

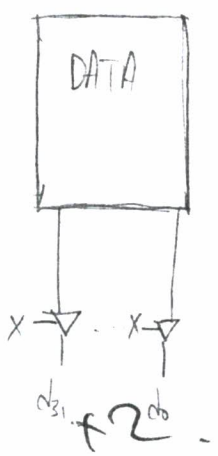
The alien memory sphere has 8 address bits, 32 data bits, and one Read signal on the positive edge of which data from a single memory location is read. This memory sphere behaves in an odd fashion, however. Prior to reading any location, the value at that particular location is first incremented by 1, and then read. There is no direct write to this memory, but the values obviously change (increase by 1) through successive readings.



4

a) (10 marks) Design a circuit to interface the alien memory sphere to your hand-held NIOS computer. Please use address locations 256-511 for accessing the memory sphere.





b) (15 marks) Write a NIOS subroutine that would WRITE the contents of register R1 into the memory sphere location defined by R2 (i.e. the location ranging from 256-511). 15

```
WRITEROUTINE
    addi r27,r27,-4  //Store return address in stack as usual.
    stw r31, 0(r27)  ✓

LOOP  ldw r4, 0(r2)  ✓ // keep incrementing the memory by 1. Into reading
      bne r4, r1, LOOP ✓ //until it reaches the same value as r1.

      ldw r31, 0(r27) //restore return address from stack
      addi r27, r27, 4
                      ✓
      ret.  ✓
```

Note: program assumes memory data will wrap around when it reaches the max value plus 1.

ie. 111111...111 + 1 = 000000...000

etc.

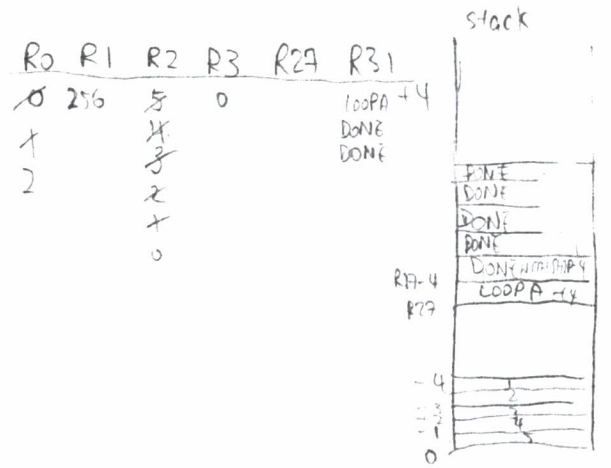
c) (5 marks) Write a NIOS assembly program that would set all memory sphere locations to 0. You may use the subroutine in part b. 5

```
      movi r1, 0  // set to zero  ✓
      movi r5, 0  //counter       ✓
      movi r2, 256  // lower bound  ✓
      movi r6, 512  //upper bound  ✓

LOOP  call WRITEROUTINE  ✓✓
      addi r2, r2, 1
      blt r2, r6, LOOP // keep writing for 256-511 mem location.
                                   ✓
```

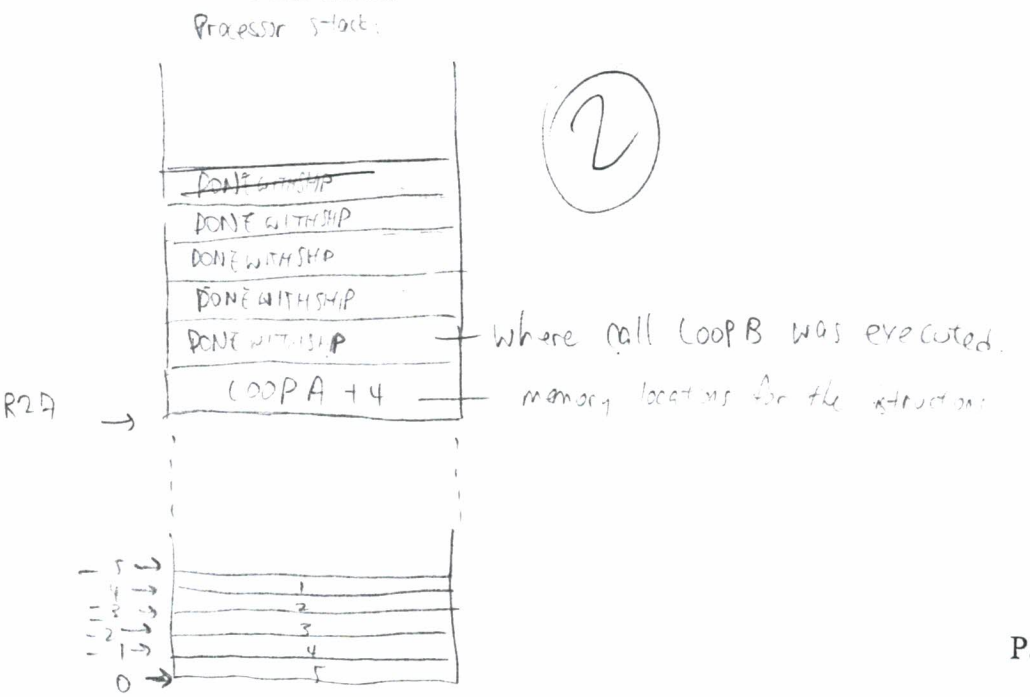
4. (15 marks)

While you were working on interfacing with the alien computer, your deputy has been busy writing a NIOS assembly program that would explode the alien ships. He, a recent Waterloo graduate, has come up with the following code:

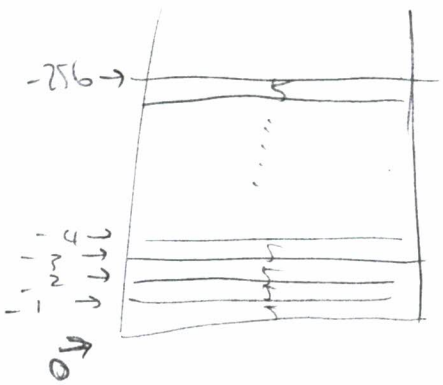
```
                MOVI R0,0
                MOVI R1,256
LOOPA           CALL EXPLODEASHIP
                ADDI R0,R0,1
                BNE R0,R1,LOOPA
STUCK           BEQ R0,R1,STUCK
                .
                .
                .
EXPLODESHIP     MOVI R2,5
LOOPB           ADDI R27,R27,-4
                STW R31,0(R27)
                STW R2,0(R0)
                ADDI R2,R2,-1
                MOVI R3,0
                BEQ R2,R3,DONEWITHSHIP
                CALL LOOPB
DONEWITHSHIP    LDW R31,0(R27)
                ADDI R27,R27,4
                RET
```



a) (4 marks) What are the contents of the stack the first time that the processor gets to the DONEWITHSHIP label?



b) (4 marks) What are the contents of the stack the first time that the processor gets to the STUCK label?



4

c) (7 marks) Rewrite your deputy's code to do the exact same thing without using any subroutines.

```
        movi r0, 0
        movi r1, 256
LOOPA   bra  EXPLODESHIP
NEXT    addi r0, r0, 1
        bne  r0, r1, LOOPA
STUCK   beq  r0, r1, STUCK

EXPLODESHIP movi r2, 5
LOOPB       stw  r2, 0(r0)
            addi r2, r2, -1
            movi r3, 0
            bne  r2, r3, LOOPB
            bra  NEXT.
```

7

5. (25 marks)

Realizing that your deputy's code is no good, you write the program yourself. The explode-all-ships program along with the English-translated version of the instructions is as follows (please note that the alien computers only have a single register R):

English meaning	Alien Instruction
0. Pause 1 minute	∞∞∞
1. Set R to 0	OOO
2. Connect with ship R	ΔΔΔ
3. Explode ship R	♥♥♥
4. Increment R by 1	ΩΩΩ
5. Goto step 2	▼▼▼

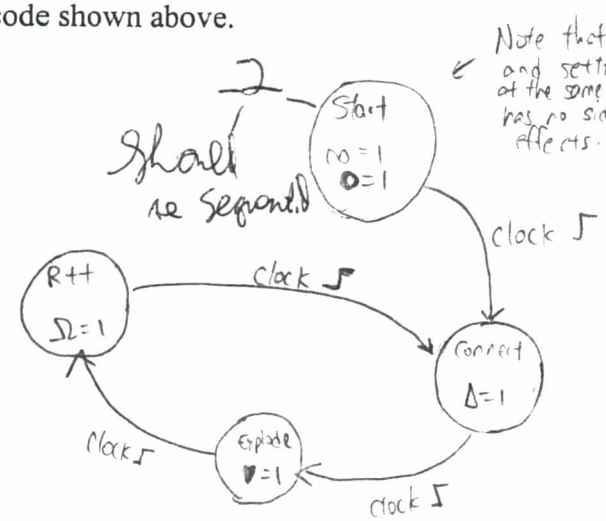
As you are about to run this program, an alien zaps your NIOS computer, your deputy, as well as the '▼' button on the alien computer. After a small fire-fight, you regain control of the computer room. But all you now have is a destroyed NIOS computer with a whole bunch of transistors. You can't even use the ▼▼▼ or 'GOTO STEP 2' instruction on the computer anymore.

Since you cannot sit there and type the program yourself, your only hope is to create a Finite State Machine that can type the program for you. The only input to this FSM is a clock signal. The outputs are:

∞	which if set to 1	Pauses for 1 minute
O	which if set to 1	Sets the computer register R to 0
Δ	which if set to 1	Connects with ship # R
♥	which if set to 1	Explodes ship # R
Ω	which if set to 1	Increments R by 1

Build your FSM, as follows:

a) (10 marks) Draw the State Diagram that would do the EXACT same thing as the original code shown above.



Note that pausing and setting R=0 at the same time has no side effects.

Assume for all states, on clock negative edge, it remains in the same state.

18/10

b) (5 marks) From your State Diagram, obtain the State Transition Table.

State bits S_1, S_0

State	S_1	S_0	Clock	∞	0	Δ	♥	Ω	S_1^+	S_0^+
				Outputs						
Start	0	0	⊓	1	1	0	0	0	0	0
									0	1
Connect	0	1	⊓	0	0	1	0	0	0	1
									1	0
Explode	1	0	⊓	0	0	0	1	0	1	0
									1	1
Rtt	1	1	⊓	0	0	0	0	1	1	1
									0	1

5/5

c) (10 marks) Obtain the next state update equations and the output logic.

Outputs: $\infty = \bar{S}_1\bar{S}_0$

$O = \bar{S}_1\bar{S}_0$

$\Delta = \bar{S}_1 S_0$

$\blacksquare = S_1\bar{S}_0$

$\Omega = S_1 S_0$ ✓

S_1^+:

S_1 \ S_0	0	1
0	0	1
1	1	0

S_0^+:

S_1 \ S_0	0	1
0	1	0
1	1	0

$S_1^+ = S_1\bar{S}_0 + \bar{S}_1 S_0$ ✓ $\quad S_0^+ = \bar{S}_0$ ✓

10/10