Homework 3

● Graded

Student

Vidit Dharmendra Pokharna

Total Points

94 / 100 pts

Question 1

Datapath Tracing — **30** / 30 pts

1.1 **LW microcode** — **10** / 10 pts

- ✔ **+ 3 pts** Asserts DrREG, LdA, LdB, RegSel=01, BRSel=1
- ✔ **+ 3 pts** Asserts DrALU, LdMAR (func=00)
- ✔ **+ 3 pts** Asserts DrMEM, WrREG, (RegSel=00)
- ✔ **+ 1 pt** Completed correctly in three clock cycles

+ 4 pts Incorrect: Working LW except does not utilize BRSel

− 1 pt Wrong bitsize of some signal

+ 0 pts Incorrect/blank/no answer

1.2 **BEQ microcode** — **20** / 20 pts

- ✔ **+ 20 pts** Correct

+ 4 pts Asserts DrREG, LdA, RegSel = 00

+ 4 pts Asserts DrREG, LdB, RegSel = 01, BRSel = 0

+ 4 pts Asserts DrALU, LdCR, func = 01

+ 0 pts Asserts ChkCmp = 1 (Correct, but we are not requiring this triggered)

+ 4 pts Asserts DrPC, LdA, LdB, BRSel = 1

+ 4 pts Asserts DrALU, LdPC, func = 00

+ 0 pts Works and utilizes BRSel, but still uses more than 5 microstates (ignoring any ChkCmp only microstates)

+ 10 pts Works, but does not utilize BRSel at all

− 2 pts Second to last microstate broken up into 2 microstates and is thus inefficient

− 1 pt Wrong bitsize of some signal

+ 0 pts Incorrect/Empty

Question 2

Datapath Design **16** / 16 pts

2.1 **LC 2200 Bus Design** **4** / 4 pts

✔ **+ 4 pts** Correct

+ 0 pts Incorrect

2.2 **Single-Bus vs Dual-Bus Design** **12** / 12 pts

✔ **+ 12 pts** Correct

+ 6 pts Mentions a benefit of single-bus design (easier/simpler to implement, more flexibility to extend datapath because all components are connected to one another/can load multiple components at the same time, etc.)

+ 6 pts Mentions a drawback of single-bus design (more clock cycles because only one component can be writing at any given moment, etc.)

+ 0 pts Incorrect

+ 0 pts Incorrect/blank/no answer

– 3 pts Answer is not specific enough

Question 3

Microcontroller Design **18** / 20 pts

3.1 **Fetch microstates** **Resolved** **2** / 4 pts

– 0 pts **Correct**
fetch3: 0000000011
nand2 : 0001000001

– 0 pts Correct

– 2 pts Incorrect fetch

✔ – 2 pts Incorrect NAND

– 1 pt Answer is reversed

– 1 pt Incorrect address size

– 1.5 pts Incorrect fifth bit (Z = 0)

– 4 pts Incorrect

Regrade Request **Submitted on: Oct 03**

Sorry this was a mistake

Reviewed on: Oct 18

3.2 **Next-State Bits** **Resolved** **6** / 6 pts

✔ – 0 pts Correct

– 3 pts One correct answer not selected

– 3 pts One incorrect answer selected

– 6 pts Two incorrect answers selected

– 6 pts Incorrect

Regrade Request **Submitted on: Oct 03**

I believe this question was supposed to be given full credit? Did that change?

Reviewed on: Oct 18

3.3 M bit — **10** / 10 pts

- ✔ **+ 5 pts** Describes that the M bit enables multiway branching for the end of fetch
- ✔ **+ 5 pts** Describes that the M bit allows the microcontroller to read the OpCode

+ 2 pts Confuses the selection functionality with that of the T bit, where we append the Z-bit instead of the opcode.

+ 0 pts Blank/no answer

+ 0 pts Incorrect

Question 4

Synchronous vs Asynchronous — **22** / 24 pts

4.1 (no title) — **4** / 4 pts

- ✔ **+ 4 pts** Correct
- **+ 2 pts** Only explains synchronous event correctly
- **+ 2 pts** Only explains asynchronous event correctly
- **+ 0 pts** Incorrect

4.2 (no title) — **Resolved** 2 / 4 pts

- **+ 4 pts** Correct
- **+ 2 pts** Only Sync
- ✔ **+ 2 pts** Only Exception
- **+ 0 pts** Incorrect

Regrade Request — **Submitted on: Oct 03**

I put exception, would that get half of the points?

resolved

Reviewed on: Oct 18

4.3 (no title) — **4** / 4 pts

- ✔ **+ 4 pts** Correct
- **+ 2 pts** Only Sync
- **+ 2 pts** Only Trap
- **+ 0 pts** Incorrect

4.4 (no title) — **4** / 4 pts

- ✔ **+ 4 pts** Correct
- **+ 2 pts** Only Async
- **+ 2 pts** Only Interrupt
- **+ 0 pts** Incorrect

4.5 **(no title)** **4** / 4 pts

✔ **+ 4 pts** Correct

+ 2 pts Only Sync

+ 2 pts Only Exception

+ 0 pts Incorrect

4.6 **(no title)** **4** / 4 pts

✔ **+ 4 pts** Correct

+ 2 pts Only Sync

+ 2 pts Only Trap

+ 0 pts Incorrect

Question 5

System vs. User Stack **8** / 10 pts

5.1 **User Program** **2** / 2 pts

✔ **+ 2 pts** Correct

+ 1 pt Only User (Mode)

+ 1 pt Only User (Stack)

+ 0 pts Incorrect

5.2 **Handler 1** **2** / 2 pts

✔ **+ 2 pts** Correct

+ 1 pt Only Kernel

+ 1 pt Only System

+ 0 pts Incorrect

5.3 **Handler 2** **2** / 2 pts

✔ **+ 2 pts** Correct

+ 1 pt Only Kernel(Mode)

+ 1 pt Only System (Stack)

+ 0 pts Incorrect

5.4 **After RTI(1)** **Resolved** **1** / 2 pts

+ 2 pts Correct

+ 1 pt Only Kernel(Mode)

✔ **+ 1 pt** Only System (Stack)

+ 0 pts Incorrect

Regrade Request **Submitted on: Oct 03**

I put system for stack, would that be half of the points?

resolved

Reviewed on: Oct 18

5.5 **After RTI(2)** Resolved 1 / 2 pts

+ 2 pts Correct

✔ **+ 1 pt** Only User (Mode)

+ 1 pt Only User (Stack)

+ 0 pts Incorrect

Regrade Request **Submitted on: Oct 03**

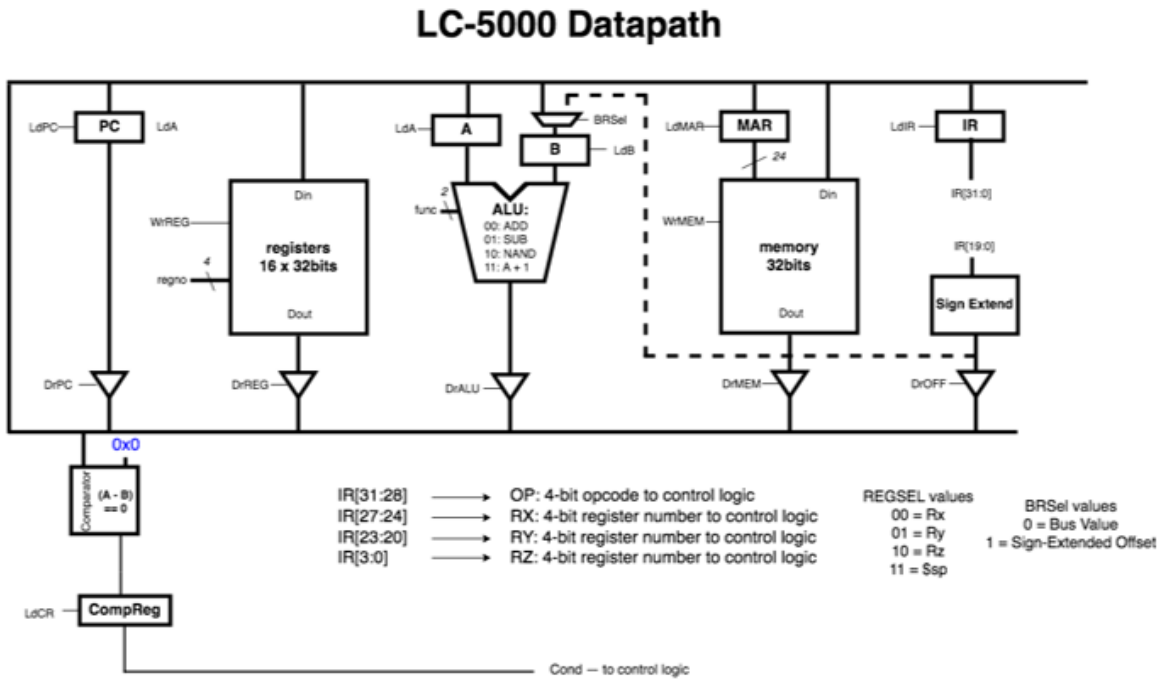
I put user for mode, would that be half of the points?

resolved

Reviewed on: Oct 18

Q1 Datapath Tracing

30 Points



The above is the datapath of the LC-5000, a modified version of LC-2200. Notice the extra MUX north of the “B” register and a wire connecting the Sign Extend output to the new B mux. Also, note that the new mux uses control signals BRSel.

Q1.1 LW microcode
10 Points

Write out the microstates for an efficient `LW` instruction that makes use of the modifications on the LC-5000 datapath. This `LW` instruction accomplishes the same goals, i.e. loads a 32-bit word from memory into DR using the Base Register and offset.

For each microstate, write the control signals used. Signals irrelevant to the state can be omitted and will be assumed to be zero. **You will lose points for an inefficient answer!** An example answer can be found below. *Note the use of RegSel instead of regno!*

Example: `ADD` instruction
ADD0: DrREG, LdA, RegSel=01
ADD1: DrREG, LdB, RegSel=10, BRSel = 0
ADD2: DrALU, WrReg, func=00, RegSel=00

Enter your microstates of the `LW` instruction for the LC-5000 below:

LW1: DrREG, LdA, RegSel=01, LdB, BRSel=1
LW2: DrALU, LdMAR, func=00
LW3: DrMEM, WrREG, RegSel=00

Q1.2 BEQ microcode
20 Points

Write out the microstates for an efficient `BEQ` instruction that makes use of the modifications on the LC-5000 datapath. For each microstate, write the control signals used. Signals irrelevant to the state can be omitted and will be assumed to be zero. **You will lose points for an inefficient answer!**

You should write out the full logic for `BEQ`; this means including the microstates for when a branch is taken. You should assume that asserting `ChkCmp` at the correct time will select the correct next state for the branch; and you can either assume that the comparison output will be directly forwarded to the microcontroller or that it will be stored in CmpReg before it can be read.

Enter your microstates of the `BEQ` instruction for the LC-5000 below:

BEQ1: DrREG, LdA
BEQ2: DrREG, LdB, RegSel=01
BEQ3: DrALU, LdCmp, ALU=01
BEQ4: ChkCmp
BEQ5: DrPC, LdA, LdB, BRSel=1
BEQ6: DrALU, LdPC

Q2 Datapath Design

16 Points

In datapath design, some approaches are to use a single bus design or dual-bus design.

Q2.1 LC 2200 Bus Design

4 Points

What type of bus design does the LC-2200 have?

- (●) Single-bus
- ( ) Dual-bus

Q2.2 Single-Bus vs Dual-Bus Design

12 Points

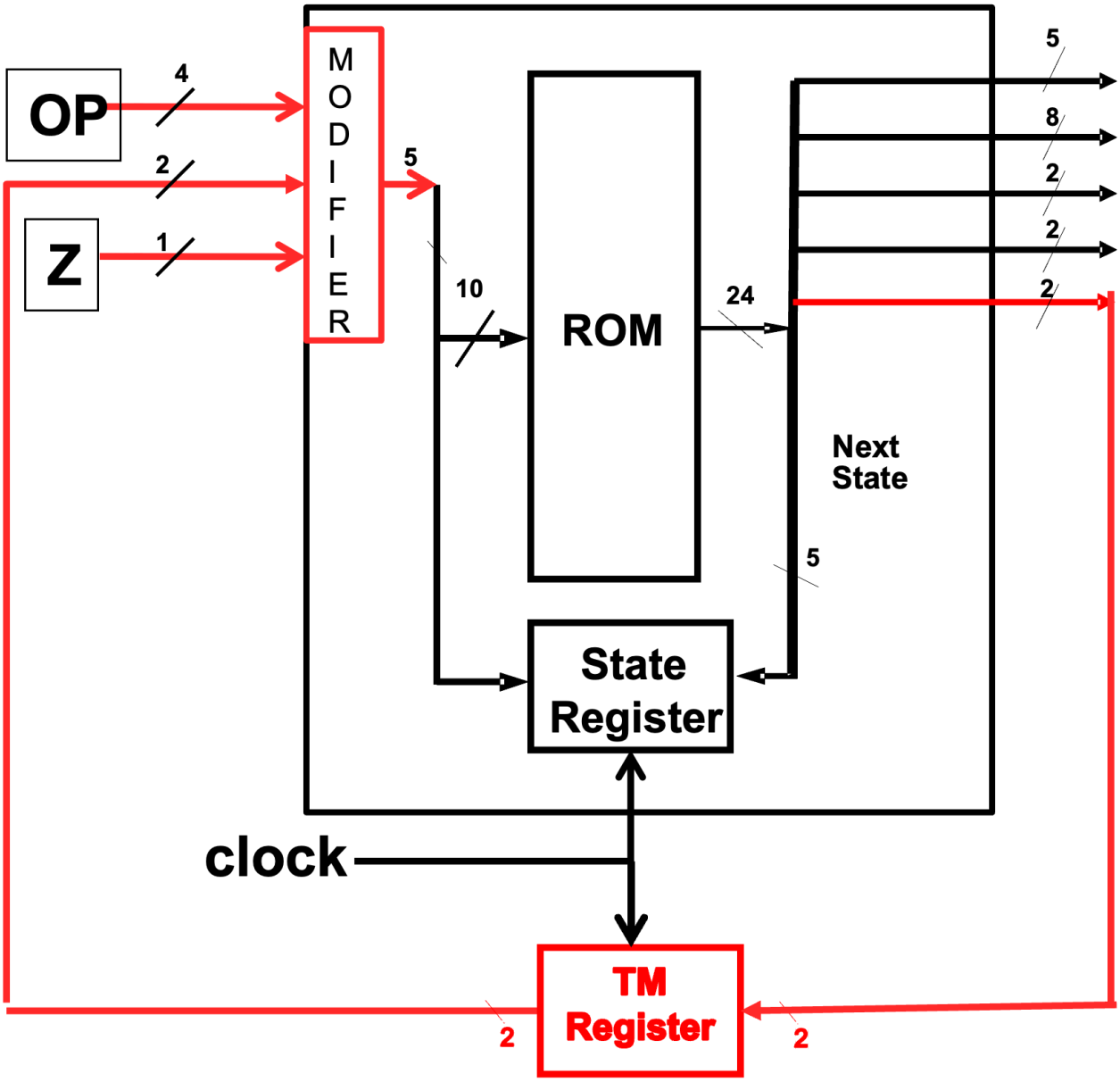
Describe one benefit of having a single-bus design AND one drawback of having a single-bus design.

Single-bus designs will likely consume less power compared to more complex designs (such as a dual-bus design), as there is less overhead, wiring, etc. associated with controlling multiple buses.

Because of the limited bandwidth in a single-bus design, data transfer rates may be slower, as only one signal can utilize the bus at any given point in time.

Q3 Microcontroller Design

20 Points



Consider the Flat ROM from the LC-2200 microcontroller. The Z-bit is equivalent to CmpOut in the LC-5500 (i.e. it tells the microcontroller whether or not to branch). We use the below as our "Next State" code. This combination of Opcode, Z bit, and state bits all determine where in our Flat ROM we will jump to next. **This is different than the LC-2222 in your project.**

The bit layout of the input to the rom looks like this:

```
MSB | --- 4 bit OP --- | --- 1 bit Z --- | --- 5 bit State --- | LSB
```

Q3.1 Fetch microstates
4 Points

Suppose fetch contains 3 microstates (fetch1 starts at *00001*), and the base address for the generic execute macrostate is *00000*. What is the address in ROM that stores the fetch3 and nand2 microstates respectively?

fetch3

0000000011

nand2

0001000010

Q3.2 Next-State Bits
6 Points

Given the current state, the next state bits, and the T bit, choose the next state(s) that are possible.

Current State: 18
Next-State Bits: 10011
T Bit: 1

- [ ] 19
- [x] 36
- [ ] 50
- [ ] 51

Q3.3 M bit
10 Points

Describe the reason **why** we have an M bit and briefly explain **how** it works.

The M bit enhances conditional branching capabilities, moving from one state to another. If M is set (M=1), it indicates that the instruction involves memory access so the top 4 bits of the address are taken from the opcode currently in the IR. If M is not set (M=0), the ROM generates the next state by taking the bits from the NextState field of the current state.

Q4 Synchronous vs Asynchronous

24 Points

Q4.1

4 Points

Explain the difference between a synchronous and asynchronous event.

A synchronous event is an occurrence that occurs at predetermined moments in time synchronized with the system's planned activities. In contrast, an asynchronous event is an occurrence that happens unexpectedly and independently of other ongoing activities within the system.

Q4.2

4 Points

Arithmetic Overflow is a(n) ____ event.

- ○ Synchronous
- ◉ Asynchronous

This would be an example of a(n) ____.

- ◉ Exception
- ○ Trap
- ○ Interrupt

Q4.3
4 Points

Making a System Call is a(n) ____ event.

- (●) Synchronous
- ( ) Asynchronous

This would be an example of a(n) ____.

- ( ) Exception
- (●) Trap
- ( ) Interrupt

Q4.4
4 Points

Receiving a printer finished message is a(n) ____ event.

- ( ) Synchronous
- (●) Asynchronous

This would be an example of a(n) ____.

- ( ) Exception
- ( ) Trap
- (●) Interrupt

Q4.5
4 Points

Calling an attribute of variable that is Null is a(n) ____ event.

- (●) Synchronous
- ( ) Asynchronous

This would be an example of a(n) ____.

- (●) Exception
- ( ) Trap
- ( ) Interrupt

Q4.6
4 Points

Writing to a file is a(n) ____ event.

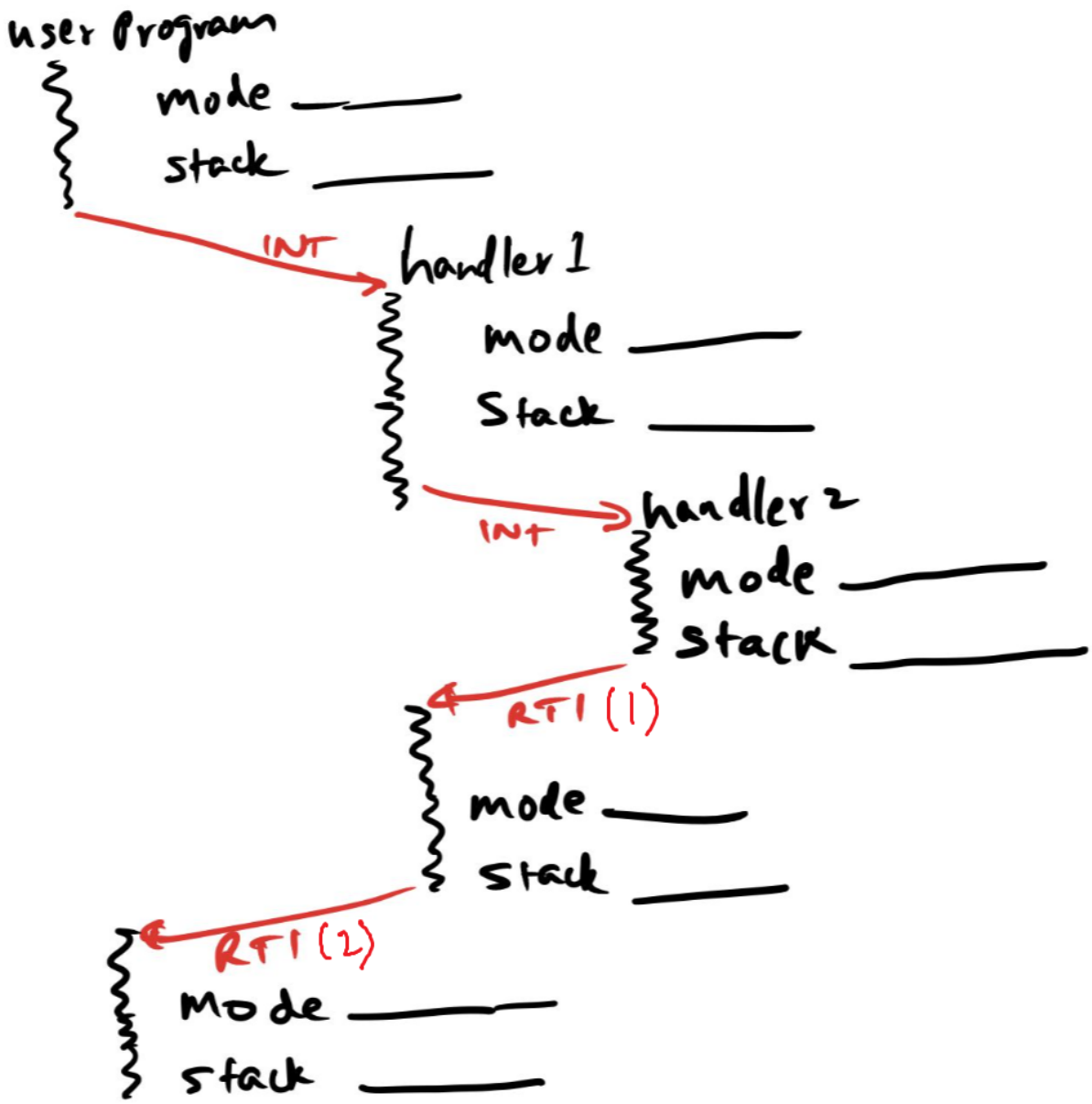
- (●) Synchronous
- ( ) Asynchronous

This would be an example of a(n) ____.

- ( ) Exception
- (●) Trap
- ( ) Interrupt

Q5 System vs. User Stack

10 Points



Fill in the blanks (in questions 5.1 to 5.5) to indicate the mode (**user, kernel**) of the processor and the state (**user, system**) of the stack. The squiggly black lines indicate a program, and the red arrows indicate a change in what program is executing (with a label indicating what operation is happening).

Q5.1 User Program
2 Points

What mode is the processor in?

- (•) User
- ( ) Kernel

What state is the stack in?

- (•) User
- ( ) System

Q5.2 Handler 1
2 Points

What mode is the processor in?

- ( ) User
- (•) Kernel

What state is the stack in?

- ( ) User
- (•) System

Q5.3 Handler 2
2 Points

What mode is the processor in?

- ○ User
- ◉ Kernel

What state is the stack in?

- ○ User
- ◉ System

Q5.4 After RTI(1)
2 Points

What mode is the processor in?

- ◉ User
- ○ Kernel

What state is the stack in?

- ○ User
- ◉ System

Q5.5 After RTI(2)
2 Points

What mode is the processor in?

- (●) User
- ( ) Kernel

What state is the stack in?

- ( ) User
- (●) System