Exam 2 - 2 EECE 344 Spring 2010

1. Information representation. We gotta have some question about number representation. Consider number system(s) that utilize a single byte - 8 bits, with the radix point right in the middle (xxxx.xxxx). For that arrangement of bits, fill in the missing elements of the following table. (Remember that Maximum is right-most on the number line; minimum is left-most on the number line.)

Value	Unsigned binary bit pattern	Twos-complement bit pattern
Maximum	1111.1111 ✓	0111.1111 ✓
Minimum	0000.0000 ✓	1000.0000 ✓
$^{11}/_{16}$	0000.1011 ✓	~~1111.0101~~ -2 0000.1011
$-^{5}/_{8}$	N/A 0000.1010 1111.0101	1111.0110 ✓
~~$12^{5}/_{8}$~~	~~1000.1010~~	N/A

$\frac{1}{2}$ $\frac{1}{4}$ $\frac{1}{8}$ $\frac{1}{16}$
$\frac{4}{8}$ $\frac{2}{8}$ $\frac{1}{8}$

2. General information question:
   a) When an interrupt occurs, what happens to the MSR[EE] bit?

The MSR[EE] bit goes to 0.

   b) The EVPR register holds 32 bits. What is the restriction on the contents of this register for normal interrupt operation?

32 bits is the maximum it can process.
(-3) 16 LSB must be 0

   c) Suppose that register R10 contains the value 0x00100000. Is it possible to use this register as a pointer to access the word address 0x000FA874? If not, why not? If so, what is the offset value used in the load instruction to reach the address?

(-3) ~~No~~ can't. To offset a pointer the max it can offset is 0xffff and 0x000FA874 is further than 0xffff from R10's value.
yes, 0000578C is the offset.

   d) Suppose R10 is as defined in part c, and R11 contains 0x00123456. Give the bit pattern found in the most significant 4 bits of the Condition Register after a `cmp 0,0,r10,r11` instruction.

R10 < R11

LT	GT	EQ	Summary Overflow
1	0	0	0

i.e. CR = 1000

   e) The counter register (CTR) is automatically decremented by some conditional branch instructions. However, the user can purposely increment or decrement the value in CTR. What steps need to be taken in order to increment CTR by 0x0100?

Basic steps to be taken is to send the counter register a pre incremented value.

count: mtctr R1
addi R1, R1, 0x100 (-2)
b count ?

must be sent to a general purpose register.

mf ctr r31
addi r31, r31, 0x0100
~~b~~ mt ctr r31

Exam 2 - 3 EECE 344 Spring 2010

3. A stack question. Most PPC systems utilize R1 as the stack pointer. And stacks are an accepted method for doing subroutine linkage. In the space provided below, provide code for subroutine linkage using a stack. Be careful with the division of responsibility between calling routines and called routines. Make sure that the method is capable of recursion. Use R31 as the place to locate the return address. Make sure that you use a minimal number of instructions, and that the instruction order is correct.

```
.org  0x5000
li   R1, 0x2000
li   R10, 0x7000

lwz  R4, 0(R10)
lwz  R5, 8(R10)
b    loop1
```

```
loop2: subf R4, R4, R4   # second routine
       mflr r1
       bl   loop3
       mflr r1

loop3: add  R5, R3, R4   # 3rd routine
       blr

loop1: add  R3, R3, R3   # first routine
       bl   loop2

FML:   b    FML
```

```
.org 0x3000
nop
nop
nop
nop
nop
mflr r31
addi r1, r1, -16
```

```
stmw r28, 0(r1)
bl   subrtn
lmw  28, 0(r1)
addi r1, r1, 16
mtlr r31
nop
nop
nop
nop
```

4. Let's try this again – that is, let's look at the interrupt system. This question deals with initialization, and the next, with steady state. In the space below, give code that will perform all of the steps needed to initialize the system to respond to two interrupts – the Fixed Interval Timer (FIT) and the Programmable Interval Timer. The clock speed is as we have it in the lab – 200 MHz. Set the FIT for its longest interval. Set the PIT to interrupt every 20 microseconds. Also clear any already set FIT or PIT flags, as well as the two mailboxes – at 0x7000 for the PIT and 0x7004 for the FIT.

```
li: r4, 0x3DA #TCR
li: r1, 0x7000
li: r2, 0x0400
li: r3, 0x0700
li: r5, 0x0300 #TSR
li: r6, 0
mt PIT, r2
stw r3, 0(r4) not this way #send rs1 to TCR
stw r6, 0(r1) # clear mailbox
stw r6, 4(r1)
```

2,000,000 = 10 ms
40,000,000 = 20 ms
PIT = 0x7000 ?
FIT = 0x7004

0x0700 = Bit mod. to turn on PIT & FIT

evpr?
PIT contents?
evpr?
EE?

PIT ← 4,000
TCR 0000|0111|1100|——
0 > < 0 → 0

200 MHz = 5 ns
20 000 / 5 = 4,000

TSR 0000|1100|0000|——
0 < 00 —

```
org 0x5000
li r8, 4000
mtpit r8
li r9, 0x700
mttcr r9
li r10, 0x0C00
mttsr r10
li r11, 0x7000
li r0, 0
stw r0, 0(r11)
stw r0, 4(r11)
mtevpr
```

Exam 2 - 5 EECE 344 Spring 2010

5. This question is a follow-on to the previous question. The intent is to provide the necessary interrupt service activity to service the two interrupts. The activity is to be the simple model that we used in class. Save current values of R28-R31 going into the ISR and restore at the end of the ISR. The register storage area starts at location 0x7F00. And the activity of the ISR is to reset the appropriate interrupt flag and also set a non-zero mailbox flag. Flag mailboxes are stored in the area starting at 0x7000. The PIT flag is located at 0x7000 and the FIT flag is located at 0x7004.

PRS
FIS

```
.org
li  R1, 0x7F00
li  R2, 0X7000
li  R3, 0x8
wset IAR, (0X1C)   Not sure what the offset is for the IAR
li  R4, 0x0
.org 0X500 ?
```


```
.org 0x1000
b PRstub
.org 0x1010
b FITstub
.org 0x4000
```

```
stw  R28, 0x0(R1)
stw  R29, 0x4(R1)
stw  R30, 0x8(R1)
stw  R31, 0xC(R1)
```

stmw!

```
stw R3, IAR(# ISR mem location)
stw R4, 0(R2)
stw r4, 4(R2)
lwz R31, 0xC(R1)
lwz R30, 0x8(R1)
lwz R29, 0x4(R1)
lwz R28, 0x0(R1)
rfi
```

PRstub:
```
li  r27, 0x7F00
stmw r28, 0(r27)
lmw  r28, 16(r27)
li's 28, 0x0E00
mtspr r28
li  r27, 0x7000
stw  r27, 0(r29)
stmw r28, 11(r27)
lmw  r28, 0(r27)
rfi
```

FITstub:
```
li  r27, 0x7F00
stmw r28, 0(r27)
lmw  r28, 32(r27)
li  r28, 0x0400
mtspr r28
li  r29, 0x7000
stw r27, 4(r29)
stmw r28, 32(r27)
lmw r28, 0(r27)
rfi
```