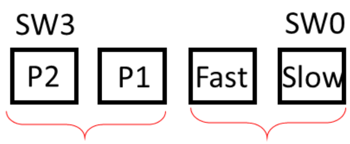
EE 306 Microprocessors Spring 2017 – 2018

Tuba Ayhan

Final Exam

4.06.2018

1. A prototype for a Christmas tree decoration is built on DE1-SoC board. The system can control 8 red LEDs on the board and uses 4 switches (SW3 to SW0) to select the blinking mode. Mode selection is given in Figure 1. A blinking mode means the pattern (P) and speed. There are two speed modes: Fast and Slow, which can be chosen with switch 1 or switch 0. One of two patterns can be selected by switch 3 and switch 4. Blinking patterns are given in Figure 2.



Select pattern, speed
Example: 1001→ Pattern 2, at Slow pace.

Figure 1: Selecting a blinking mode

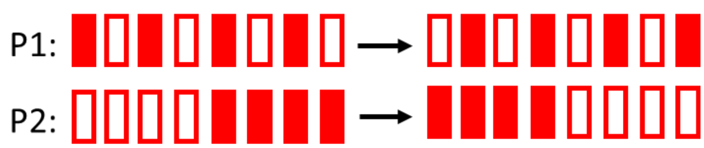


Figure 2: Blinking patterns

Below is an assembly program for this light decoration. Complete the program.

a. Define binary templates for patterns P1 and P2.
b. READSW: Read the switch values to R0 and R1. Then get the pattern and speed information by help of two constants, “speedmask” and “patternmask”. Pattern information should be on R0 and speed information should be on R1. Assume only one pattern and one speed is selected.
c. Write the selected pattern (P1 or P2) to R2.
d. According to the speed selection, write #10 or #20 to R3. This register will be used in delay generation.
e. Display the selected pattern on LEDR.
f. Wait for a while. Use a DODELAY1 subroutine to generate delay. The delay time depends on selection on R3 and you will need this value on the second half of the blink (check h).
g. Bitwise complement R2 to obtain the second half of the blinking pattern. Display new R2.
h. Wait for a while. Use a DODELAY2 subroutine to generate delay.
   Then you can go to b. and continue, as given in the assembly template.

```
.equ SW , 0x FF200040 // Switch base address
.equ LEDR, 0xFF200000  // LEDR base address
.equ P1, _______
.equ P2, _______
.equ speedmask, _______
.equ patternmask, _______
_start:
// read the pattern and speed:
      LDR R5, =SW
      LDR R6, =LEDR
READSW:
      // answer b-h ..........................
      B READSW
.end
```

Final Exam

4.06.2018

2. Assume that your CPU on DE1-Soc board is busy with a complex algorithm: copying a very big data from one source to a destination. Every 5 seconds, we want to flash the amount of data that is already copied to the destination, using a seven-segment display. A timer is used to generate an interrupt, via GIC.

a. Write the assembly code to configure the timer for generating an interrupt every 5 seconds. The timer is driven by a 100-MHz clock. Configuration registers for the timer are given in the figure.

Address	31 ... 16 | 15 ... 3	2	1	0	Register name
0xFFC08000	Load value				Load
0xFFC08004	Current value				Counter
0xFFC08008	Unused	I	M	E	Control
0xFFC0800C	Unused			F	End-of-Interrupt
0xFFC08010	Unused			S	Interrupt status

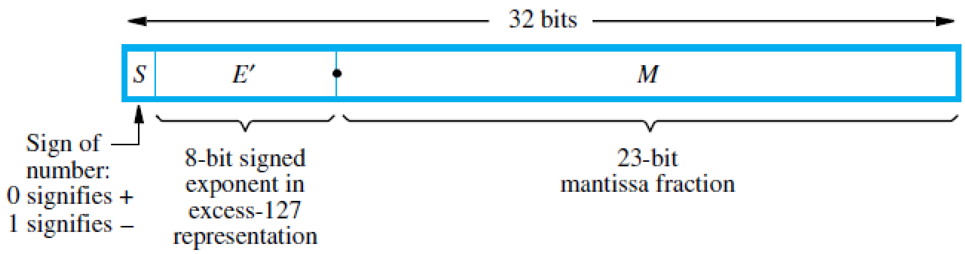
b. In order to make interrupts available for your processor, you need to do the following in order. Fill in the blanks

1) _____________ IRQ interrupts.
2) Configure __________. Interrupts for each I/O peripheral device are identified by a unique _______________________.
3) Configure each ___________________________, so that it can send an ___________________ to the GIC.
4) _____________ IRQ interrupts.

c. Fill the SERVICE_IRQ subroutine:

```
SERVICE_IRQ:
     ______________________________          // firstly, save the
                                             //variables you are using in
                                             //main
/* Read the ICCIAR from the CPU interface */
     LDR        R4, =MPCORE_GIC_CPUIF
     LDR        R5, [R4, #ICCIAR]            // read the interrupt ID
     __________________________________      // check if this interrupt is
                                             //from HPS timer.
UND: BNE        UND                          // stay here for undefined irq
     BL         TIMER_ISR
     B          EXIT_IRQ
EXIT_IRQ:   // Write to the End of Interrupt Register (ICCEOIR)
     STR        R5, [R4, #ICCEOIR]
     _________________________________       // bring the variables back
      _________________________________      // Go to the last instruction
                                             //in the main. Do not forget
```

3. You are asked to compare two floating point numbers given in the excess-127 representation. Propose a comparison way without using floating point co-processor. Plot the flowchart. Watch out the sign bit!



Value represented $= \pm 1.M \times 2^{E'-127}$

Final Exam

26.05.2017

4. *There are "fill in the blanks" questions about pipelining, memory, and memory management unit.*

Example:

____ is the time that elapses between the initiation of an operation to transfer a word of data to/from a memory and the completion of that operation.

SOLUTIONS

Solution 1 - 23 ticks in total→ 23 point

Solutions…..

```
.equ SW , 0x FF200040 // Switch base address
.equ LEDR, 0xFF200000  // LEDR base address
.equ P1, 0b10101010                //                    a
.equ P2, 0b11110000                //                    a
.equ speedmask, 3                  //                    b
.equ patternmask, 12               //                    b
_start:
// read the pattern and speed:
     LDR R5, =SW
     LDR R6, =LEDR
READSW:
     LDR R0, [R5] // read pattern and speed on R0 and    b
     MOV R1, R0   // copy it to R1                       b
     AND R1,R1, #speedmask //                            b
     AND R0,R0, #patternmask //                          b
     CMP R0,#0                                           c
     MOVEQ R2,#P1                                        c
     MOVNE R2,#P2                                        c
     CMP R1,#0                                           d
     MOVEQ R3,#20                                        d
     MOVNE R3,#10                                        d
     STR R2,[R6]                                         e
     MOV R7,R3                                           f
DODELAY1: SUBS R7,R7,#1                                  f
     BNE DODELAY1                                        f
     EOR R2,R2,#FF                                       g
     STR R2,[R6]                                         g
     MOV R7,R3                                           h
DODELAY2: SUBS R7,R7,#1                                  h
     BNE DODELAY1                                        h
     B READSW
.end
```

Solution 2: a→ 2 pnt for each assembly line, 1 pnt for correct timeout calculation. = 2x5+1=11

```
LDR R1, =0Xffc08000       // timer base address
LDR R3, =500000000        // timeout = 1/(100MHz)x 500e6 = 5 sec
STR R3, [R1]              // write to timer load register
MOV R3, #0b011            // set bits: mode = 1 (auto), enable = 1
STR R3, [R1, #0x8]        // write to timer control register
```

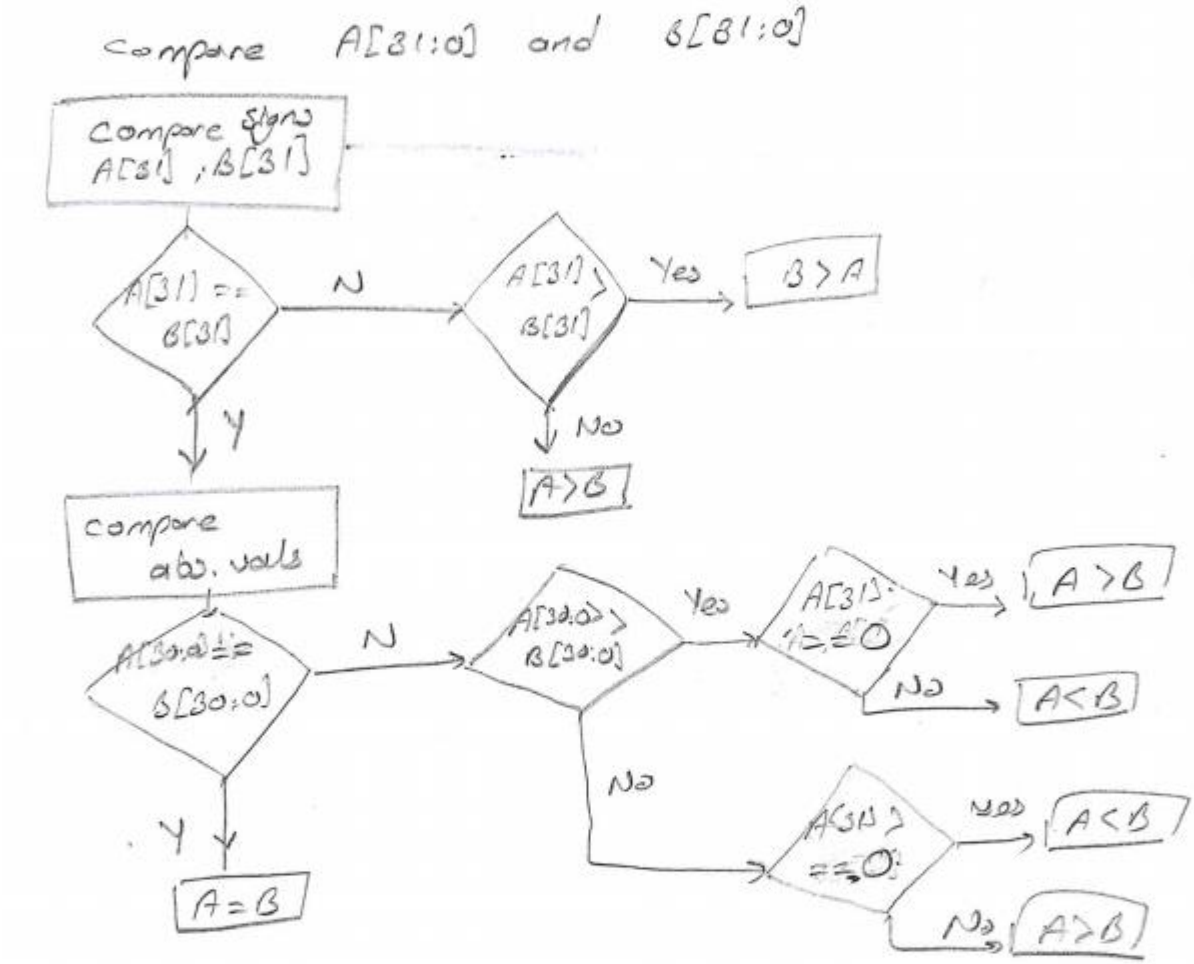
b. 6 pnt per blank.

1) Disable IRQ interrupts.
2) Configure GIC. Interrupts for each I/O peripheral device are identified by a unique Interrupt ID.
3) Configure each I/O (peripheral) , so that it can send an interrupt request to the GIC.
4) Enable IRQ interrupts.

c. 2 pnt x 4 instructions = 8 pnt

```
SERVICE_IRQ:
     PUSH {R0-R7(Rx), LR}                   // firstly, save the
                                            //variables you are using in
                                            //main
/* Read the ICCIAR from the CPU interface */
     LDR       R4, =MPCORE_GIC_CPUIF
     LDR       R5, [R4, #ICCIAR]            // read the interrupt ID
     CMP       R5, #TIMER_IRQ               // check if this interrupt is
                                            //from HPS timer.
UND: BNE       UND                          // stay here for undefined irq
     BL        TIMER_ISR
     B         EXIT_IRQ
EXIT_IRQ:   // Write to the End of Interrupt Register (ICCEOIR)
     STR       R5, [R4, #ICCEOIR]
     POP       {R0-R7(Rx), LR}              // bring the variables back
     SUBS   PC, LR, #4                      // Go to the last instruction
                                            //in the main. Do not forget
                                            //pipelining!
```

Solution 3: 28 points → 1 point for each box, decision and directed arrow.



Solution 4-9: (total 24 point)

(2pnt) 7. <u>Memory access time</u>____________________________ is the time that elapses between the initiation of an operation to transfer a word of data to/from a memory and the completion of that operation.