CS1S Computer Systems: Questions and Solutions
May 2017

Duration: 1 hour
Rubric: Answer both questions
This examination is worth a total of 50 marks

1. **(a)** Convert 1001 0110 to a decimal number, assuming binary representation.

[2]

128 + 16 + 4 + 2 = 150 by adding the powers of 2 corresponding to the positions where there is a 1 bit in the word.
[Problem solving.]

(b) Convert 1001 0110 to a decimal number, assuming two's complement representation.

[3]

Since the leftmost bit is 1, this is a negative number. Negate it to get a nonnegative number. To negate, first invert it giving 0110 1001. Then increment it, giving 0110 1010. Now this result is nonnegative so its binary representation is the same as its two's complement value; this is 64 + 32 + 8 + 2 = 106. Since the negation of the original word is 106, the answer is -106.
[Problem solving. 1 mark for identifying it as negative; 2 marks for negation.]

(c) Translate the statement `a = b + x[i]` into Sigma16 assembly language, assuming that `a`, `b`, and `i` are integer variables and `x` is an array. You do not need to write data statements to define the variables or the array.

[4]

```
   load   R1,b[R0]   ; R1 = b
   load   R2,i[R0]   ; R2 = i
   load   R3,x[R2]   ; R3 = x[i]
   add    R1,R1,R3   ; R1 = b + x[i]
   store  R1,a[R0]   ; a = b + x[i]
```

[Problem solving; requires understanding addressing, memory accesses, and integers. 1 mark for variable accesses, 1 mark for array access, 1 mark for calculation, 1 mark for storing result.]

(d) Show the sequence of steps performed by the processor as it executes the instruction `load R2,x[R1]`.

[4]

(1) The instruction is fetched: ir = mem[pc], pc++. (2) Then the opcode is examined and the control jumps to the control for a load. The second word of the instruction is fetched: adr = mem[pc]; this places the value of x in adr. (3) The effective address is calculated: adr = adr + R1 = x + R1. (4) The data is fetched and placed in the destination

register: R2 = mem[adr + R1] = x[R1].
[Bookwork and synthesis. 1 mark for each step.]

(e) There is an array named `x` that contains `n` integers, where `n` is an integer variable in memory. Write a Sigma16 assembly language program that calculates the sum of the positive elements of the array (i.e. the elements that are greater than 0) and stores this sum in the variable `SumPos`. The program should define `n = 6` and define the initial elements of the array as 6, -9, 7, 4, -1, 2. After the program runs the variable `SumPos` should have the value 19. The program must work correctly for any nonnegative `n` and initial array elements.

[12]

```
; R1 = i = loop index
; R2 = n = array size
; R3 = constant 1
; R4 = running sum of positives
; R5 = temp

; Initialise the variables in registers
       add    R1,R0,R0      ; i = 0
       load   R2,n[R0]      ; R2 = n
       lea    R3,1[R0]      ; R4 = 1
       add    R4,R0,R0      ; R4 = 0 (SumPos)

; Traverse the array and calculate sum of positives
loop   cmp    R1,R2         ; compare i with n
       jmpge  done[R0]      ; if i >= n then goto done
       load   R5,x[R1]      ; temp = x[i]
       cmp    R5,R0         ; compare temp with 0
       jmple  skip[R0]      ; if x[i] <= 0 then goto skip
       add    R4,R4,R5      ; SumPos = SumPos + x[i]
skip   add    R1,R1,R3      ; i = i + 1
       jump   loop[R0]      ; goto loop
; Save result and terminate
done   store  R4,SumPos[R0] ; save SumPos
       trap   R0,R0,R0      ; terminate

; Define variables
n      data   6
SumPos data   0
x      data   6
       data   -9
       data   -7
       data   4
       data   -1
       data   2
```

[Problem solving, requires understanding of the instruction set, conditionals, indexed addressing and loops. 3 marks for initialization, 3 marks for loop, 3 marks for conditional, and 3 marks for result and termination.]

2. **(a)** Draw the circuit diagram for a multiplexer (the mux1 circuit). The circuit takes inputs `c`, `x`, `y` and produces output `z`, where `z = (if c=0 then x else y)`. Use Boolean algebra to show that if `c=1` then `z=y`.

[5]

The circuit is `mux1 c x y = or2 (and2 (inv c) x) (and2 c y)`. Either the corresponding diagram or this HDL description of the circuit is acceptable, and the simplifications may be shown either as annotations on the diagram or using algebraic notation. Assume `c=1`. Then

```
  mux1 c x y
  = or2 (and2 (inv c) x) (and2 c y)  {def of mux1}
  = or2 (and2 (inv 1) x) (and2 1 y)  {substituting c=1}
  = or2 (and2 0 x) (and2 1 y)        {inv 1 = 0}
  = or2 0 y                          {and2 0 x = 0,and2 1 y = y}
  = y                                {or2 0 y = y}
```

[Bookwork and calculation (seen previously). 2 marks for circuit, 3 marks for calculation.]

(b) Explain why the speed of a synchronous circuit depends on the longest path through logic gates. Describe how a suitable clock speed for the circuit is determined, and explain what can happen if the clock speed is too fast.

[5]

The path depth of a combinational circuit is the number of logic gate delays required for the circuit's outputs to become valid, after the inputs have all become valid. The critical path of a circuit is the path with the maximal path depth. The clock must run slowly enough so that the outputs of the critical path are valid before the next clock tick; thus the critical path determines how fast the circuit can run. Shorter paths are irrelevant because their outputs will become valid before the critical path's outputs become valid. If the clock is too fast, then a tick may reach a flip flop before that flip flop's input is valid, and this can result in an incorrect value being latched into the flip flop. This is a fatal and unrecoverable error.
[Bookwork plus synthesis. 1 mark for definition, 2 for speed determination, 2 for explanation of excessively fast clock.]

(c) Define the terms *interrupt* and *process*. Describe how the operating system uses interrupts to implement concurrent processes.

[5]

An interrupt is a jump executed by the processor in response to an event, but not as the result of executing a jump instruction. The event may be caused by an external signal, for example the timer going off or an I/O device needing service. A process is a running program with evolving state. The operating system allows multiple processes to run concurrently, although at any point in time the processor is executing only one instruction. The OS maintains a table of processes, including information about which portion of memory each process is using. A process has a state, either ready, running, or blocked. When an interrupt occurs, the machine jumps from a running process to the OS, which selects a ready process. The OS grants it a time slice by setting the timer to a

fixed period and jumping to the process, which will run until the timer goes off or the process is interrupted by I/O. Over a long period of time (a second) all the processes will be given time slices, so they all make progress.
[Bookwork. 2 marks for definitions, 3 marks for explanation.]

(d) Define the terms *circuit switching* and *packet switching*, and state which technique is used in the Internet. Give two advantages of using packet switching.
[5]

In circuit switching, a route from one host to another is established before communication begins, and switches on that route are dedicated to this connection for the duration of the communication. All information between the hosts goes through the same path, and other data cannot go through the same switches. In packet switching, all data is organized into packets of a fixed size, which are sent in a sequence of steps from one router to another. There is no permanent connection between the hosts, and different packets could take different paths. The Internet uses packet switching, which has several advantages, including the following. Packet switching is more efficient for "bursty" traffic, where a large amount of data is sent and then there is no transmission for a relatively long time (in circuit switching this wastes bandwidth). Packet switching is more flexible for error recovery: missing or corrupted packets can be detected (using timeouts or checksums) and then retransmitted, and it doesn't matter if packets arrive out of order because they can be reassembled by the receiver. Packet switching allows a larger number of hosts to communicate, as switching hardware is never allocated exclusively to one connection.
[Bookwork plus synthesis. 2 marks for definitions, 1 for Internet, 2 marks for advantages of packet switching.]

(e) Give two services provided by the TCP protocol in the Internet, and describe how TCP recovers from errors. Explain what the IP protocol does, and the purpose of the DNS protocol.
[5]

The TCP protocol delivers an incoming packet to the appropriate application, e.g. browser or email client. It also provides reliable service using an unreliable network; this is achieved by using checksums and timers to detect missing or corrupted packets, and requesting those to be retransmitted. The IP protocol sends a packet from one router to the next, along a path from the sending host to the receiving host. Each router examines the IP address of the packet and determines which router to send to packet on to. The DNS protocol converts a symbolic internet address (e.g. www.bbc.co.uk) into the corresponding IP address, which is just a number and hard for users to remember.
[Bookwork. 3 marks for TCP, 2 for IP.]