G51CSA Exam sample answers

These are sample answers, they are not necessarily the only correct solution, neither are they necessary the best solution. Text in italics are commentary on the question and/or answers — like this.

Question 1

a) *The answers for these can easily be found by using the Google calculator (search for* `123 in binary`*, or* `0o123 in hex` *). In terms, of calculating these by hand its useful to remember that the binary version is easily converted to and from hex and octal (there's a one to one mapping between a hex or octal digit and four or three sequential binary digits respectively) so it probably makes sense where its not given to calculate that first.*

b) m = 44 or `00101100`
n = -56. 56 = `00111000`, therefore we invert and add one to get -56. Invert: `11000111` Add One: `11001000`
o = 77 or `01001101`

c) Answers for the sums using the above values

m+n

```
00101100 +
11001000
11110100
```
No overflow

n - o

-o = `10110011` *(same procedure as above)*

```
11001000 +
10110011
01111011
```
(and carry 1) Overflow

m + o

```
00101100 +
01001101
01111001
```
No overflow

Remember overflow occurs when the result of a 2's complement addition means the sign bit (the MSB) has the wrong sign for the calculation.

Question 2

*The key with this type of question is given in the hint in the **Note**, don't try and test more address lines than you need to. So it's just a matter of finding a logic equation that matches the bits in the highest address lines.*

a)

The AND gate and OR gate both combine two (or more) logic signals to produce one output. the difference is that an AND gate's output is only true if **all** of it's inputs are true while an OR gate's output is true if any of it's inputs are true.

b)

```
VIDCS = ~A15 . ~A14 . ~A13 . ~A12. ~A11. ~A10 . ~A9 . ~A8
```

(remember this uniquely decodes the addresses for VIDEO I/O, even though it also decodes lots of other addresses)

```
RAMCS = ~A15 . ~A14 . ~A13 . ~A12. ~A11. ~A10 . ~A9 . A8
IOCS  = ~A15 . ~A14 .  A13 . ~A12
ROMCS =  A15 .  A14 .  A13 .  A12
```

c)

```
INVALID = ~VIDCS . ~RAMCS . ~IOCS . ~ROMCS
```

or

```
INVALID = ~(VIDCS+RAMCS+IOCS+ROMCS)
```

Question 3

a) `a = 3, b = 100, c =75`. This routine multiplies R1 and R2 by doing a repeated addition.

b)

Removing `for` loop

```
y = 1;
i = 1;
goto while_cond;
while_loop:
    y = y * i;
    i++;
while_cond:
if(i <= x)
    goto while_loop;
```

Add `goto`s

```
y = 1;
i = 1;
goto while_cond;
while_loop:
    y = y * i;
    i++;
while_cond:
if(i <= x)
    goto while_loop;
```

Convert to ARM assembler

```
B main
x   DEFW 42
main
    MOV R0, #1  ; y
    MOV R1, #1  ; i
    LDR R2, x
    B while_cond
while_loop
    MUL R0, R0, R1
   ADD R1, R1, #1
while_cond
    CMP R1, R2
    BLE while_loop
```

c)

```
x DEFW 42
```

```
LDR R0, #x
ADD R1, R0, #1
MUL R2, R0, R1
MOV R0, R2 ASR #1 ; divide by two…
SWI 4
SWI 2
```