Exam Solution

Programme DT228-1
Module: Microprocessors
Sitting: Summer 2014
Examiner: Frank Duignan

S228/999

DUBLIN INSTITUTE OF TECHNOLOGY
KEVIN STREET DUBLIN 8

BSc. (Honours) Degree in Computer Science

Year 1

Semester 2 Examinations 2013/2015

Microprocessors

Mr. F. Duignan
Dr. D. Lillis

Wednesday 24th May 4.00 p.m. – 6.00 p.m.

SOLUTIONS

Exam Solution

Programme DT228-1
Module: Microprocessors
Sitting: Summer 2014
Examiner: Frank Duignan

Question 1

(a)

(i) 0xb3314a5b AND 0x2417ffce	=	0x20114a4a	[3]
(ii) 0x8a293dce OR 0x753e2170	=	0xff3f3dfe	[3]
(iii) 0x7231a3bf XOR 0x1248a8e2	=	0x9726b595	[3]

(b)
What is the hexadecimal representation of the 32 bit number -5?
0xFFFFFFFB [4]

(c)
8 bit signed range: -128 to +127 [2]
12 bit signed range: -2048 to +2047 [2]
24 bit signed range: -8388608 to +8388607 [2]

(d)

```
void IntToString(unsigned int val, char *String)
{
        // maximim value approx 4 billion
        // this implies that there are 10 digits plus a NULL
        int index = 10;
        String[index]=0; // terminate the string with a NULL
        while (index)
        {
                index--;
                String[index]=(val % 10) + '0';
                val = val / 10;
        }
}
```

Loop not essential – can do long hand also. Looking for:
Understanding of ASCII
Understanding of Modulo arithmetic
Understanding of Integer divide
Understanding of how strings reside in memory.
[9]

Exam Solution

Programme DT228-1
Module: Microprocessors
Sitting: Summer 2014
Examiner: Frank Duignan

Question 2
(i)
GPIO0DIR: This is the direction control register for port 0. A 1 at a particular bit location makes the corresponding port bit an output. A 0 makes a bit an input. [2]
GPIODATA : This is the register that contains the state of the port pins. Output bits can be written here, input bits can be read here. [2]

(ii)
Set BIT5 of General Purpose IO Port 0 without affecting the other bits
GPIO0DATA = GPIO0DATA | (1<<5); [4]

(ii)
Clear BIT2 of General Purpose IO Port 0 without affecting the other bits
GPIO0DATA = GPIO0DATA & ~(1<<2); [4]

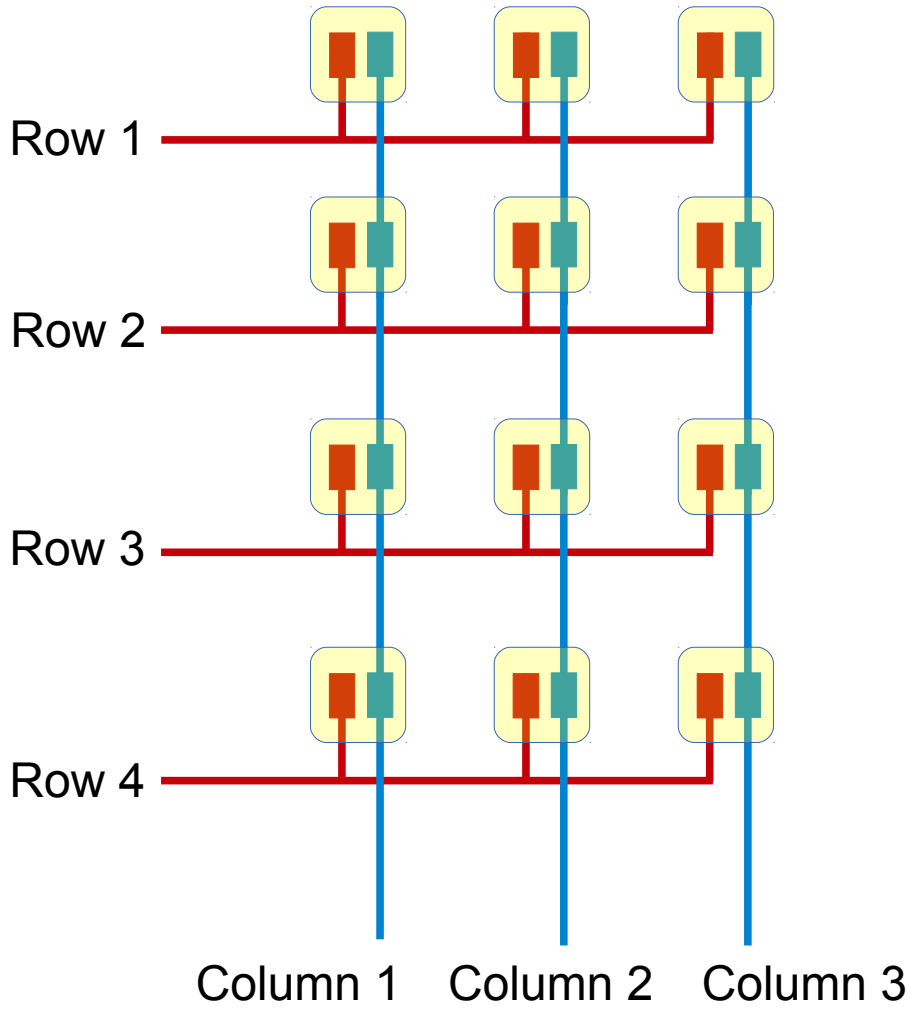
(iii) Bits 2 and 5 of General Purpose IO Port 0 to be outputs, all other bits to be inputs
GPIO0DIR = (1<<5) + (1<<2); [4]

(iv) Wait for bit 6 of General Purpose IO Port 0 become logic 1
while ((GPIO0DATA & (1 << 6)) == 0); [4]

Exam Solution

Programme DT228-1
Module: Microprocessors
Sitting: Summer 2014
Examiner: Frank Duignan

(c)
(i)



Looking for diagram similar to above with a description of how each column is asserted in turn (driven low in this case) and how the micro checks for zeros on the row pins. An NxM matrix keyboard requires only N+M I/O pins using this method as opposed to 2*M*N without multiplexing. [8]

(ii) Missing code: [5]

```
GPIO1DATA |= COL_1 | COL_2 | COL_3;
GPIO1DATA &= ~COL_3;
if ((GPIO1DATA & ROW_1) == 0)
      return '3';
if ((GPIO1DATA & ROW_2) == 0)
      return '6';
if ((GPIO1DATA & ROW_3) == 0)
      return '9';
if ((GPIO1DATA & ROW_4) == 0)
      return '#';
```

Exam Solution

Programme DT228-1
Module: Microprocessors
Sitting: Summer 2014
Examiner: Frank Duignan

Question 3.

```
// global variable
int i;                  i DCD 0
int k;                  k DCD 0                                   [2]
:
k=1;                    LDR R0,=1
                        LDR R1,=k
                        STR R0,R1


for (i=0;i<10;i++)                                                [5]
(i=0)
                        LDR R0,=0
                        LDR R1,=i
                        STR R0,[R1]
k=k*2
                        LDR R1,=k
                        LDR R0,[R1]
                        LSLS R0,R0,#1
                        STR R0,[R1]


(i++)
                        LDR R1,=i
                        LDR R0,[R1]
                        ADDS R0,R0,#1
                        STR R0,[R1]

(i<10)
                        LDR R1,=i
                        LDR R0,[R1]
                        CMP R0,#10
                        BLT Loop_Start
        k = k * 2;                                                [3]
```

Exam Solution

Programme DT228-1
Module: Microprocessors
Sitting: Summer 2014
Examiner: Frank Duignan

Question 4
(a)
PC = Program Counter. Contains the address of the next instruction to be executed [2]
LR = Link Register. Contains the return address during a fucntion call [2]
SP = Stack Pointer. Points to the last value placed on the stack. It is used by push and pop instruction when removing/placing data on the stack [2]
PSR = Program Status Register. Contains arithmetic statis flags as well as control flags for the processor [2]

(b)
1-1 : Z=1, C/NB = 1 [2]
0xffffff9+12 : C = 1 [2]
1-2 : N flag (Note: NB=0 implies there was a borrow) [2]
0x7fffffff+2 : N=1, V=1 [2]

(c)
(i) Which instruction is executed first following reset?
BL main [2]

(ii) An assembler directive is an instruction to the assembler which controls the assembly process. e.g. **end**, **DCD**, **AREA**
[3]

(iii)
Line A: Push registers R0,R1, R2 and the Link Register (LR) on to the stack. [3]
Line B: Load R1 with the value represented by the symbol Num1 [3]
Line C: Add R0 and R1 together, placing the result in R0. The flags are updated. [3]
Line D: Pull the registers R0,R1 R2 and PC from the stack. This has the effect of transferring control back to the calling function. [3]