Computer Organisation and Architecture – End of topic test 2

Q1. Explain the role of any 3 control lines available to the control bus (3 marks)

Q2. Explain, using an example, the relationship between the width of the address bus and the memory capacity of the system. (2 marks)

Q3. State a system that may benefit from using the Harvard architecture instead of the Von Neumann architecture and explain why. (2 marks)

Q4. Describe the use of any two dedicated special registers in the CPU. (4 marks)

Q5. A computer has instructions that are set out with 7 bits for the op code, 1 bit for the addressing mode and 8 bits for the operand.

a) Explain what an instruction set is and what the maximum number of instructions this computer can handle is. (2 marks)

_____

_____

_____

b) 1 bit has been used to determine the addressing mode. Explain the two possible addressing modes you need to be familiar with. (2 marks)

_____

_____

_____

_____

Q6. Explain what an interrupt is and how it is handled by a CPU. (4 marks)

_____

_____

_____

_____

_____

_____

_____

_____

Q7. Explain what a computer word is and how it relates to the address bus. (2 marks)

Q8. Aside from word length and bus widths, describe 3 factors that affect processor performance. (6 marks)

Q9. Write the assembly code instructions which are equivalent to the following high-level code. Comment each line of code.

```
x -> 0
WHILE x < 1000
        x -> x + 1
ENDWHILE
```

(6 marks)

LDR Rd, <memory ref>	Load the value stored in the memory location specified by <memory ref> into register d.
STR Rd, <memory ref>	Store the value that is in register d into the memory location specified by <memory ref>.
ADD Rd, Rn, <operand>	Add the value specified in <operand> to the value in register n and store the result in register d.
SUB Rd, Rn, <operand>	Subtract the value specified by <operand> from the value in register n and store the result in register d.
MOV Rd, <operand>	Copy the value specified by <operand> into register d.
CMP Rn, <operand>	Compare the value stored in register n with the value specified by <operand>.
B <label>	Always branch to the instruction at position <label> in the program.
B<condition> <label>	Conditionally branch to the instruction at position <label> in the program if the last comparison met the criteria specified by the <condition>. Possible values for <condition> and their meaning are: EQ: Equal to, NE: Not equal to, GT: Greater than, LT: Less than.
AND Rd, Rn, <operand>	Perform a bitwise logical AND operation between the value in register n and the value specified by <operand> and store the result in register d.
ORR Rd, Rn, <operand>	Perform a bitwise logical OR operation between the value in register n and the value specified by <operand> and store the result in register d.
EOR Rd, Rn, <operand>	Perform a bitwise logical exclusive or (XOR) operation between the value in register n and the value specified by <operand> and store the result in register d.
MVN Rd, <operand>	Perform a bitwise logical NOT operation on the value specified by <operand> and store the result in register d.
LSL Rd, Rn, <operand>	Logically shift left the value stored in register n by the number of bits specified by <operand> and store the result in register d.
LSR Rd, Rn, <operand>	Logically shift right the value stored in register n by the number of bits specified by <operand> and store the result in register d.
HALT	Stops the execution of the program.