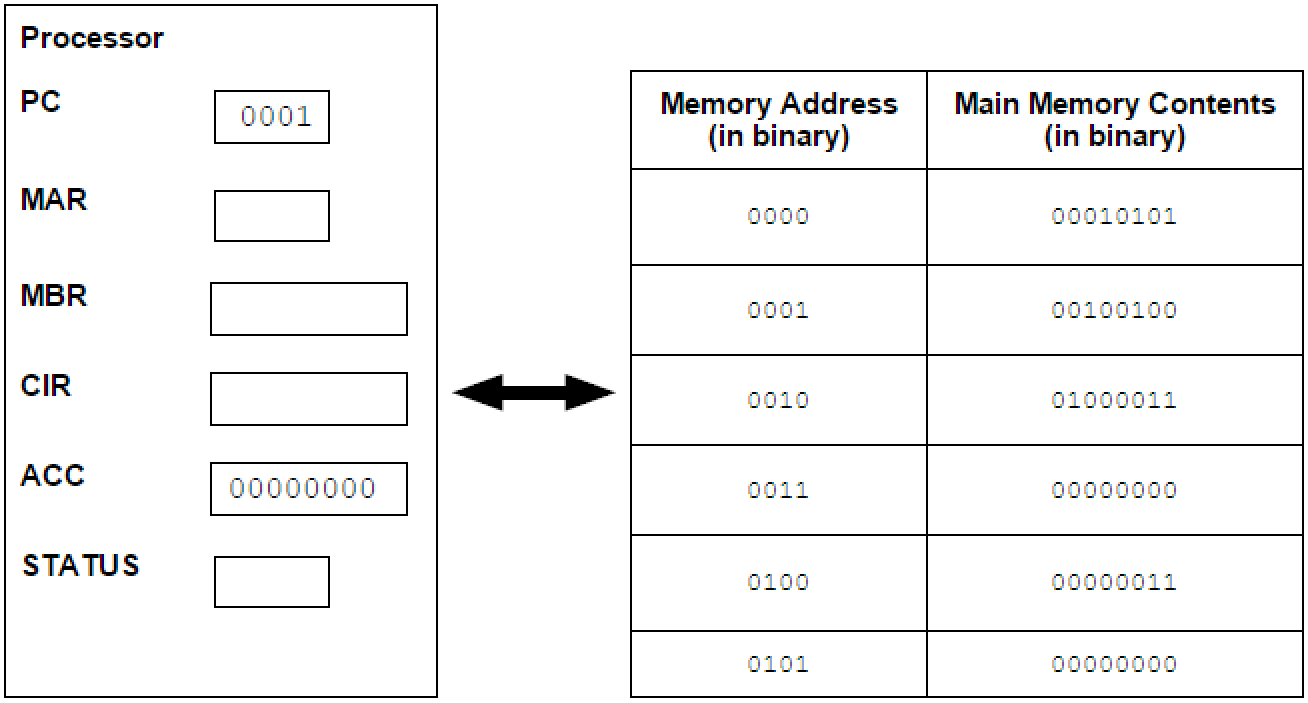
1. The control unit is an important component of a processor.

Describe the role of the control unit.

(Total 3 marks)

2. The diagram below shows some of the registers used in the fetch-execute cycle of a simple processor and the contents of a small section of main memory that it is connected to by the system bus ($\longleftrightarrow$).



Memory Address (in binary)	Main Memory Contents (in binary)
0000	00010101
0001	00100100
0010	01000011
0011	00000000
0100	00000011
0101	00000000

OPCODE	INSTRUCTION	DESCRIPTION
0001	LOAD	Load the contents of the provided memory location into the accumulator
0010	ADD	Add the contents of the provided memory location to the current contents of the accumulator, storing the result in the accumulator
0100	STORE	Copy the contents of the accumulator into the provided memory location

(a) In the diagram above the first 4 bits of an instruction represent the opcode and give the type of instruction to be executed.

What name is given to the second 4 bits of an instruction?

____

(1)

(b) (i) Currently the value in the Program Counter (PC) is example `0001`.

Complete the table below by writing the values, expressed in binary, in the following registers after completing the fetch part of the fetch-execute cycle.

Register	**Value**
PC	
MAR	
MBR	

(3)

(ii) Describe what will happen during the decode and execute part of the cycle.

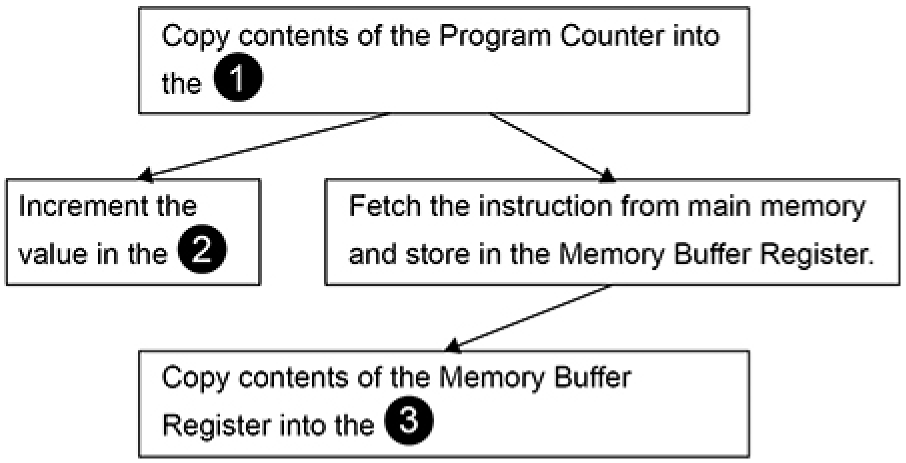
(3)

(c) What would be the outcome of executing the instruction `01000011?`

(1)

(Total 8 marks)

3. The diagram below describes the fetch part of the Fetch-Execute cycle. Some of the names of registers have been omitted from the figure and replaced with the numbers ❶ to ❸


Copy contents of the Program Counter into the 1
Increment the value in the 2
Fetch the instruction from main memory and store in the Memory Buffer Register.
Copy contents of the Memory Buffer Register into the 3

State the **full names** of the registers that should appear in the diagram where the numbers are.

Number	**Full Name of Register**
❶	
❷	
❸	

(Total 2 marks)

4.

(a) State the full names of **two** of the special purpose registers that are used in the fetch part of the fetch-execute cycle.

Register 1 ____

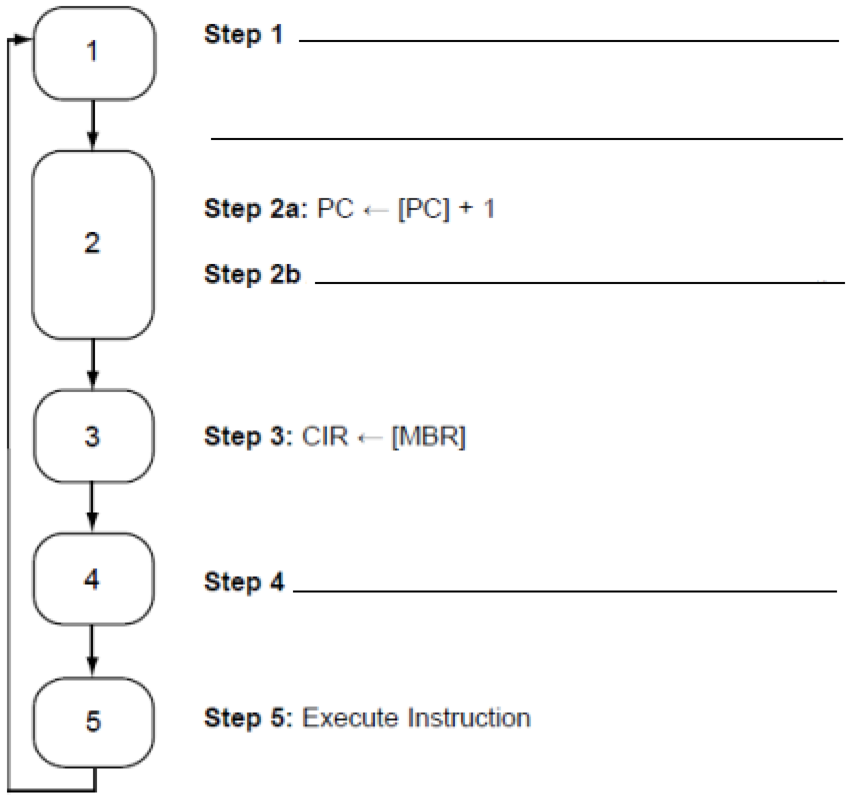
Register 2 ____

(2)

(b) **Figure 1** below is an incomplete diagram of the fetch-execute cycle.

Describe the missing steps 1, 2b and 4 using either register transfer notation or a written description. Steps 2a and 2b occur at the same time.

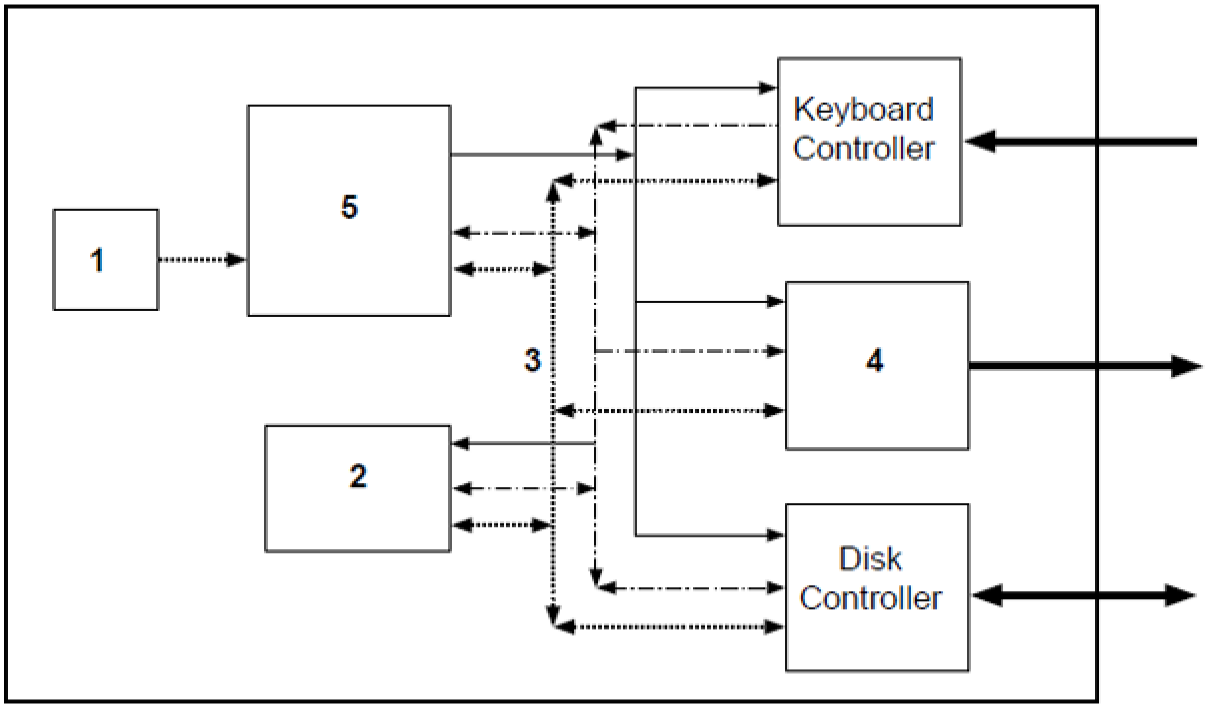
Figure 1



(3)

(Total 5 marks)

5. The diagram below shows some of the components of a computer system.



(a) Suggest names for the components numbered **1** to **5** in diagram above by completing the table below.

Number	**Component Name**
1	
2	
3	
4	
5	

(5)

(b) In the first step of the Fetch-Execute Cycle the contents of the Program Counter arecopied into another register

State the **full name** of this register.

____

(1)

(c) An item of data or an instruction fetched into the processor is initially loaded into a register.

State the **full name** of this register.

____

(1)

(d) Modern computers often have a *64-bit address bus*.

Explain what this means.

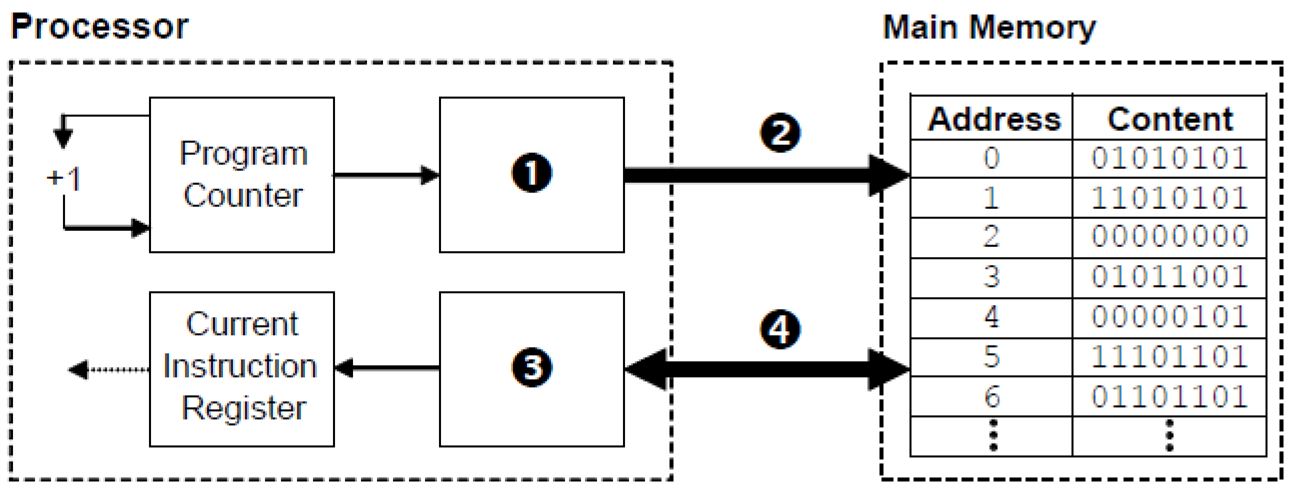
____

____

(1)

(Total 8 marks)

6. The diagram below shows the processor registers and busses that are used during the fetch part of the fetch-execute cycle, together with the main memory. The values stored in memory locations 0 to 6 in the main memory are machine code instructions.



(a) Name the components that are labelled with the numbers 1 to 4. In the case of register names, the full names must be stated.

Number	**Component Name**
❶	
❷	
❸	
❹	

(4)

(b) Explain what happens during the decode and execute stages of the fetch-execute cycle.

____

____

____

____

____

____

(3)

(c) The machine code instructions in the main memory in the diagram above are shown in binary.
When programmers look at machine code instructions they usually prefer to view them in hexadecimal.

State **one** reason why this is the case.

____

____

(1)

(d) The machine code instructions in the main memory in the diagram above were produced when an assembly language program was translated into machine code.

(i) What type of program translator was used to do this?

____

(1)

(ii) Most computer programs are initially written in an imperative high level language rather than assembly language.

Explain why this is the case.

____

____

____

____

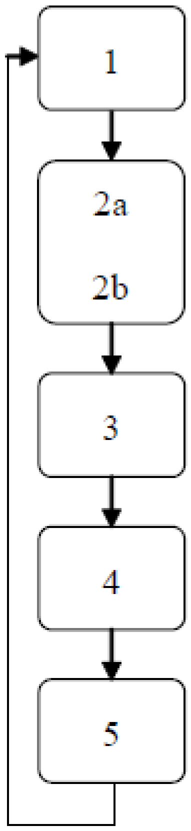
____

____

(3)

(Total 12 marks)

7. The figure below shows the fetch-execute cycle. Steps 2a and 2b occur at the same time.



Step 1: MAR ← [PC]

Step 2a: PC ← [PC] + 1

Step 2b: MBR ← $[\text{Memory}]_{\text{addressed}}$

Step 3: CIR ← [MBR]

Step 4: Decode Instruction

Step 5: Execute Instruction

(a) State the full names of **two** of the special purpose registers that are used in the fetch part of the fetch-execute cycle.

Register 1: ____

Register 2: ____

(2)

(b) Explain the role of the address bus, data bus and main memory during Steps 1 and 2b.

____

____

____

____

____

(2)

(c) Give **one** reason why Steps 2a and 2b are able to occur at the same time.

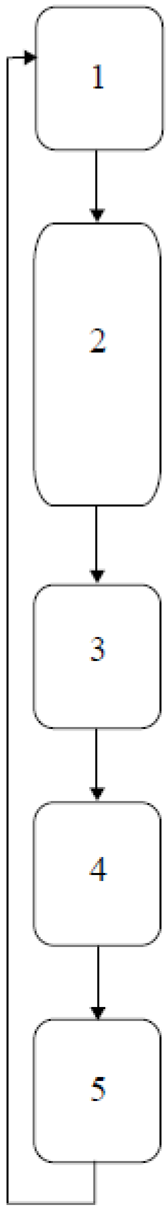
____

____

(1)
(Total 5 marks)

8. The diagram below shows the fetch-execute cycle. Some of the steps have been described.

(a) Describe the missing steps 1, 2b and 4 using either register transfer notation or a written description. Steps 2a and 2b occur at the same time.



Step 1:____

____

Step 2a: PC ← [PC] + 1
(Increment contents of Program Counter Register)

Step 2b: ____

____

Step 3: CIR ← [MBR]
(Transfer contents of Memory Buffer Register into Current Instruction Register)

Step 4:____

____

Step 5: Execute Instruction

(3)

(b) What would be the effect on the performance of the computer system of increasing the

(i) width of the data bus? ____

____

____

(ii) width of the address bus? ____

____

____

(iii) clock speed? ____

____

____

(3)

(Total 6 marks)

9. The Fetch-Execute cycle can be described in register-transfer language as follows:

MAR$\leftarrow$ [PC]

PC $\leftarrow$[PC] + 1; MBR $\leftarrow$[Memory]$_{\text{addressed}}$

CIR$\leftarrow$ [MBR]

[CIR] decoded and executed

where [ ] means *contents of*.

Describe the Fetch-Execute cycle in your own words.

____

____

____

____

____

____

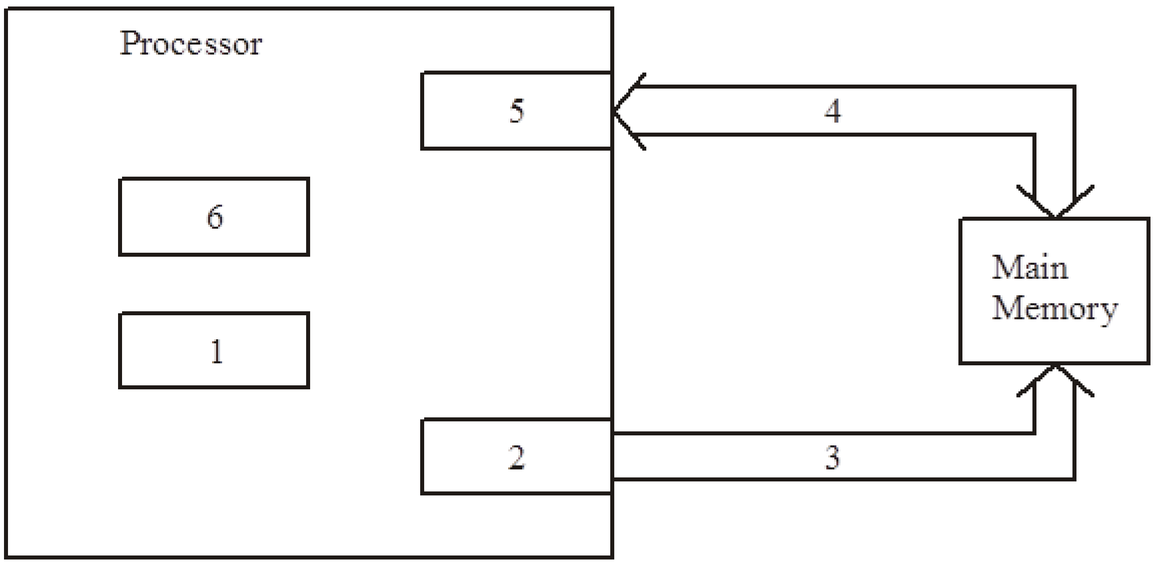
____

____

____

(Total 6 marks)

10.

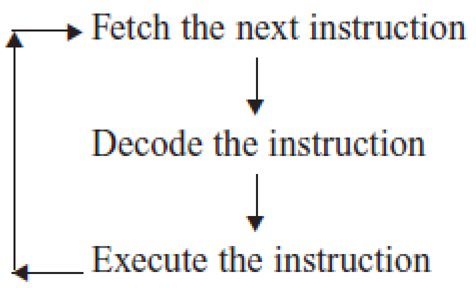


As part of the fetch-execute cycle of a computer system the processor has to fetch the next instruction. The figure above shows the main components used. They are used in the sequence 1, 2, 3, 4, 5, 6 to fetch the next instruction. Name the components by completing the table below.

Component	Name
1	Program Counter
2	
3	
4	
5	
6	

(Total 5 marks)

11. The fetch execute cycle may be described as:



(a) Name **four** registers that are used in the Fetch Decode part of the cycle.

1. ______________________________

2. ______________________________

3. ______________________________

4. ______________________________

(4)

(b) (i) What additional steps would be required if the computer system had an interrupt mechanism?

____

____

(2)

(ii) Where would they be placed in the above cycle?

____

(1)

(c) (i) Describe the vectored interrupt mechanism.

____

____

____

____

(3)

(ii) How does this mechanism make the use of interrupts more flexible?

____

____

(1)

(Total 11 marks)

12. A processor with an instruction format of 16 bits and a word length of 16 bits is being used.

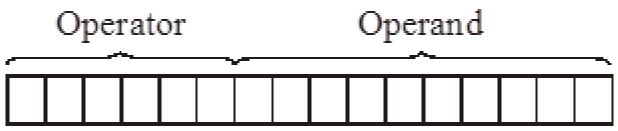
(a) Integers are stored in 2's complement form. What is the possible range of integers that can be stored in a 16-bit word?

____

____

(2)

(b) The instruction format uses 6 bits for the operator and 10 bits for the operand.



If direct addressing is used, what is the highest address possible?

____

____

(1)

(c) The main registers involved in the fetch-execute cycle are the Program Counter (PC), the Current Instruction Register (CIR), Memory Address Register (MAR) and the Memory Data Register (MDR). List the **steps** of the fetch-execute cycle, including how the above registers are used.

____

____

____

____

____

____

(6)
(Total 9 marks)

13. (a) Some of the basic components of a computer system are processor, main memory, and secondary storage.

(i) What connects the processor and main memory?

____

(1)

(ii) What is the purpose of secondary storage?

____

____

(1)

(iii) Describe what happens during the fetch-execute cycle.

(2)

(b) (i) Machine code is the first generation programming language. What is the second generation?

(1)

(ii) A programmer writes a program in a second generation programming language. What has to be done to this program before it can be executed?

(2)

(iii) Some high level languages are classified as *imperative*. What is meant by imperative?

(1)

(iv) Give an example of an imperative high level language.

(1)

(v) What is the relationship between an imperative high level language statement and its machine code equivalent?

(1)

(vi) Give **two** disadvantages of programming in first and second generation programming languages compared with imperative high level languages.

1. ____

____

2. ____

____

(2)

(Total 12 marks)

14. Registers are involved in the fetch part of the fetch-execute cycle.

Name **three** of these registers, describe what each will store, and give **one** further detail about its role.

1 Name the register ______________________________

What does it store? ______________________________

Further detail ______________________________

2 Name the register ______________________________

What does it store? ______________________________

Further detail ______________________________

3 Name the register ______________________________

What does it store? ______________________________

Further detail ______________________________

(Total 9 marks)

15. Name **one** register involved in the fetch part of the fetch-execute cycle.

____

____

(Total 1 mark)

Mark schemes

1.

All marks AO1 (knowledge)

To marshal / control operation of fetch-execute cycle;
Controls fetching / loading / storing operations; **NE.** fetches instructions Determines the type of an instruction; **A.** decodes instructions
To execute (some) instructions;
To synchronise operation of processor;
To send control signals / commands to other components;
To control the transfer of data between registers;
To handle interrupts;

Max 3

[3]

2.

(a) operand;
R operand code

1

(b) (i)

PC	0010;
MAR	0001;
MBR	00100100;

3

(ii) The instruction is held in the CIR // instruction in CIR is decoded;
A IR

The control unit / instruction decoder decodes the instruction;
NE the processor decodes the instruction

Instruction will be split into opcode and operand;
R if it is implied that a register will do this splitting / decoding

Relevant part of processor / CPU executes instruction // using ALU to perform calculations;
A instruction executed by the control unit / ALU
NE processor executes instruction

Further memory fetches / saves carried out if required;

Result of computation stored in accumulator / register / written to main memory;

Status register updated;
If jump / branch instruction PC is updated;

By example:
Will ADD contents memory location 0100 to accumulator;

MAX 3

(c) The current value in the accumulator would be stored in (memory) address / location 0011 / 3;

Number 011 / 3 stored in (memory) address / location 0011 / 3;

MAX 1

[8]

3. **2 marks for AO1 (knowledge)**

Number	Full Name of Register
❶	Memory Address Register **NE.** MAR
❷	Program Counter **NE.** PC
❸	Current Instruction Register **NE.** CIR, IR **A.** Instruction Register

1 mark: Two registers correctly named OR
2 marks: All three registers correctly named

If student has used initialisms instead of full register names (or a mixture of both) then award **1 mark** if all three registers are given the correct abbreviated name.

[2]

4.

(a) Program Counter / Sequence Control Register;
Memory Address Register;
Memory Buffer Register / Memory Data Register;
Current Instruction Register;
R Abbreviations

Max 2

(b) **Step 1:** MAR $\longleftarrow$ [PC] / Contents of program counter transferred to MAR;

R MAR $\longleftarrow$ PC

R [MAR] $\longleftarrow$ PC (see note about **DPT**)
R PC sends / transfers

Step 2b: MBR $\longleftarrow$ [Memory]$_{\text{addressed}}$ / Contents of addressed memory location loaded into MBR; (must have concept of data coming from address in memory, not just going into MBR)

Step 4: Decode instruction;
A Contents of CIR decoded
A Instruction is split into opcode and operand
R Data for instruction
R CIR decoded, CIR decodes instruction
Note: A. [CIR] decoded

1 mark for each correct step

For PC accept Program Counter / SCR / Sequence Control Register
For MAR accept Memory Address Register
For MBR accept Memory Buffer Register / MDR / Memory Data Register

A Other means of indicating transfer e.g. [PC] $\longrightarrow$
MAR
A [Memory] for [Memory]$_{\text{addressed}}$
DPT – no / incorrect square bracket use for register transfer notation

3

[5]

5.

(a) 1 – clock;
2 – (Main) memory / IAS; **A** RAM **R** ROM
3 – Control bus;
4 – VDU controller / output controller; **A** controller for other named output device
5 – Processor; **R** Central Processing Unit / CPU

5

(b) Memory address register;
R abbreviations

1

(c) Memory buffer register / memory data register;
R abbreviations

1

(d) Address bus has 64 lines / tracks/ wires // there are 2 ^64 memory locations available;
NE 64 bits wide, moves 64 bits of data

1

[8]

6. (a)

Number	**Component Name**
1	Memory Address Register
2	Address Bus
3	Memory Data / Buffer Register
4	Data Bus

4

(b) The instruction is held in the CIR;
A IR
The control unit / instruction decoder decodes the instruction;
The opcode identifies the type of instruction it is;
Relevant part of CPU / processor executes instruction;
A ALU
Further memory fetches / saves carried out if required;
Result of computation stored in accumulator / register / written to main memory;
Status register updated;
If jump / branch instruction, PC is updated;
A SCR

Max 3

(c) Can be displayed in less space;
R takes up less space **NE**
Easier to remember / learn / read / understand;
Less error prone;

Max 1

(d) (i) Assembler;

1

(ii) HLLs are problem oriented;
HLL programs are portable // machine / platform independent ;
English like **keywords / commands/ syntax / code**;
R closer to English
Less code required // less tedious to program //
one to many mapping of HLL statements to machine code commands;
Quicker/easier to understand / write / debug /learn / maintain code;
R just quicker/easier
HLLs offer extra features e.g. data types / structures // structured statements // local variables // parameters // named variables/constants;
R procedures / modular
A example of a data structure
NE "extra features" without example
Speed of execution not crucial for most tasks so faster execution of assembly language not required;
Most computer systems have a lot of (main) memory / RAM so compact object code not essential;
A converse points for Assembly Language

3

[12]

7.

(a) Program Counter;
A Sequence Control Register
R Next Instruction
Register
Current Instruction Register;
A Instruction Register
Memory Buffer Register;
A Memory Data Register
Memory Address Register;

Max 2

(b) Address in MAR/address to fetch instruction from, sent down Address Bus to Main Memory;
R address in PC (program counter)
Contents of address accessed in Main Memory;
A by implication if
contents of address location referred to during data transfer
Contents of address location//instruction//data passed down Data Bus into MBR/to processor;
A MDR instead of MBR
A RAM for Main Memory

Max 2

(c) Order of execution unimportant/one step does not rely on prior completion of the other;
Steps carried out by different (hardware) devices/components;
A operations are independent
A operations use different registers
R using different buses

Max 1

[5]

8.

(a) **Step 1**: MAR ←[PC] / Contents of program counter transferred to MAR;

Step 2b: MBR ← $[\text{Memory}]_{\text{addressed}}$ / Contents of addressed memory location loaded into MBR; (must have concept of data coming from address in memory, not just going into MBR)

Step 4: Decode instruction;
A Contents of CIR decoded
R Data for instruction
R CIR decoded, CIR decodes instruction

1 mark for each correct step

For PC accept Program Counter / SCR / Sequence Control Register
For MAR accept Memory Address Register
For MBR accept Memory Buffer Register / MDR / Memory Data Register
For CIR accept Current Instruction Register / IR / Instruction Register
A Other means of indicating correct transfer e.g. [PC] → MAR or MAR:=PC
A Missing square brackets or alternative types of brackets
A Answers that miss out reference to "contents of"
A [Memory] for $[\text{Memory}]_{\text{addressed}}$

3

(b) (i) Increases the number of bits (**A** amount of data) that can be transferred at one time // increase rate of data transfer;

1

(ii) Increases the number of memory addresses / /Increase the maximum amount of primary store/memory (possible);

1

(iii) Instructions performed more quickly // Instructions executed at faster rate;
A Calculations for instructions (this time only)
A Operations for instructions
NE Speeds the computer up
R Processes, tasks for instructions

1

[6]

9.

1. address of <u>next</u> instruction to be executed/fetched;
2. (contents of Program Counter) copied into <u>Memory Address Register</u>;
3. Contents of <u>Program Counter</u> incremented (by 1);
   A incrementing by more than 1
4. ...at the same time....; *(only give a mark if between correct statements)*
5. instruction/data held at that address is placed in the <u>Memory Buffer Register</u>;
6. Contents of Memory Buffer Register copied into <u>Current Instruction Register</u>;
7. <u>Instruction</u> held in Current Instruction Register is decoded;
8. If necessary data is fetched;
9. (and) instruction is executed by processor/ALU;
10. Address sent/transferred over address bus;
11. Data/instruction transferred to processor on data bus;
12. Result stored in accumulator;

Max 6

[6]

10.

Component	Name
1	Program Counter
2	Memory Address Register; **A** MAR
3	Address Bus;
4	Data Bus;
5	Memory Data Register/ Memory Buffer Register; **A** MDR/MBR
6	Current Instruction Register; A Instruction Register/IR/CIR

[5]

11.

(a) Program Counter; Sequence Control Register/Instruction Pointer
Instruction Register// Current Instruction Register;
Memory Buffer Register// Memory Data Register;
Memory Address Register;
Penalise initials once only

4

(b) (i) Test for an interrupt/ check priority of interrupt
Identify the <u>source</u> of the interrupt;
Save and/or restore the volatile environment/registers;
<u>Service</u> the interrupt; <u>A</u> handle the interrupt
Disable (lower priority) interrupts

Max 2

2

(ii) Placed between Execute and Fetch;
A before Fetch/ after execute <u>R</u> at end of cycle

1

(c) (i) Interrupting device/ source supplies;
An offset/vector;
A index/indexed address added to the base address;
A base register

Gives the start address of interrupt service routine/ ISR//
Address vector table cell contains start address of ISR/
R Interrupting device supplies start address of ISR

3

(ii) A different routine can be easily introduced//routine can be relocated/ dynamically loaded; *or words to this effect*
A The interrupting device only needs to supply a new offset

1

[11]

12.

(a) From -2^{15}/-32,768; to 2^{15}-1/32767; **R** *binary;* **R** *one number unqualified*

2

(b) 2^{10}-1 / 1023; **A** 1111111111_2; **A** &3FF; **A** $3FF_{16}$;

1

(c) Step 1: (Content of) PC copied into MAR // [PC] ® MAR; *accept {} for []*
Step 2: (Content of) PC incremented (by 1) // [PC] incremented (by 1);
Step 3: content of Memory location addressed by MAR; loaded into MDR;// MAR addresses a memory location; whose content is loaded into MDR;
Step 4: Content of MDR copied into CIR // [MDR] ® CIR;
Step 5: Content of CIR decoded // [CIR] decoded // opcode/instruction/data decoded;
Step 6: instruction executed;

Step 2 could be after step 3,4 or step 5 but otherwise order of steps must be as above, step(s) may be missed out steps do not have to be one per line

P1 of it is difficult to pick out the steps from prose
A loaded/moved/stored for copied
A MBR for MDR
A IR for CIR
A SCR for PC

Max 6

[9]

13. (a) (i) (Data/address/control/internal/system) bus;
R just a description of a bus
R names of buses which don't exist e.g. memory bus

1

(ii) Store programs and/or data/files when not in use/
When computer is off permanent/long term storage
Of programs and/or data; save programs/data;
R offline/backup **R** ROM **R** temporary storage
A save on magnetic disk/ tape storage;
A information instead of data

1

(iii) (Machine code) instruction/data is fetched from main memory;
A what is fetched or from where
Instruction is decoded;
Instruction is executed (by the processor); **R** data executed

Max 2

(b) (i) Assembly language; mnemonic code; mnemonics; assembly code;
R low level language **A** assembler;

1

(ii) Translated/assembled/converted/decoded; into machine code (instructions);
R compiled **R** interpreted **A** object/target code;
A binary instructions;

2

(iii) Computer executes instructions in programmer defined sequence;
A the programmer tells the computer how to do it;
R user *instead of* programmer

1

(iv) Pascal /Visual Basic/Basic/C/C++/Cobol/Fortran/Ada/Delphi/Lylix/Modula /or any other imperative HLL
R Prolog
R Lisp
R Pop11

1

(v) One statement/instruction/command in a high level language translates into several machine code instructions; 1 to many;

1

(vi) Laborious/time-consuming to write; hard to debug; harder to program; easier to make mistakes; more difficult to understand/ learn; difficult to maintain; different assembler/instruction set for different type of computer; machine dependent; low level programs not portable;

Max 2

[12]

14. *Any 3 from:*
PC/SCR - Address of next instruction; - To be fetched;
MAR - Address of data &/or instructions; - That are to be read from - (main) memory;
MBR / MDR - Data &/or instruction; - Read from (main) - memory;
IR/CIR - Holds current instruction; - While it is being - decoded and executed;

Correct name/acronym required. Accept other correct 'Further details' – but must refer to role of register being described. May be incorporated in 'what stored' and carried forward.

[9]

15. One from: PC/SCR, MAR, MBR(MDR). LR/CIR(not address/data register)

[1]