


CMR INSTITUTE OF TECHNOLOGY		USN <input type="text"/>						 <small>CMRIT</small> <small>THE INSTITUTE OF TECHNOLOGY, MUMBAI</small> <small>ACCREDITED WITH A GRADE BY MAAC</small>		
First Internal Test										
Sub:	COMPUTER ORGANIZATION						Code:	18CS44		
Date:	07/09/2019	Duration:	90 mins	Max Marks:	50	Sem:	III	Branch:	ISE	
Answer Any FIVE FULL Questions										
								Marks	OBE	
									CO	RBT
1(a)	Explain the steps involved in instruction fetching from memory and execution in the processor with the help of block diagram.						[5]	CO1	L2	
(b)	Define the Little Endian and Big Endian assignments. Explain with relevant diagrams.						[5]	CO1	L2	
2(a)	What are the operations of DMA controller? Explain the centralized arbitration and distributed bus arbitration schemes.						[10]	CO1	L2	
3(a)	Define Subroutine. How to pass parameters to subroutine? Explain with your own Example.						[10]	CO1	L3	
4(a)	Explain any five addressing modes with example of each mode.						[10]	CO1	L2	

5 a)	Calculate the SPEC rating for the program suite under test. Running times of the program suite for reference PC and PC under test are given below:																										
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: left;">Programs</th> <th style="text-align: center;">Running time for reference PC</th> <th style="text-align: center;">Running time for PC under test</th> </tr> </thead> <tbody> <tr> <td>P1</td> <td style="text-align: center;">10</td> <td style="text-align: center;">20</td> </tr> <tr> <td>P2</td> <td style="text-align: center;">100</td> <td style="text-align: center;">50</td> </tr> <tr> <td>P3</td> <td style="text-align: center;">40</td> <td style="text-align: center;">20</td> </tr> <tr> <td>P4</td> <td style="text-align: center;">10</td> <td style="text-align: center;">5</td> </tr> <tr> <td>P5</td> <td style="text-align: center;">60</td> <td style="text-align: center;">30</td> </tr> </tbody> </table>						Programs	Running time for reference PC	Running time for PC under test	P1	10	20	P2	100	50	P3	40	20	P4	10	5	P5	60	30	[5]	CO1	L3
Programs	Running time for reference PC	Running time for PC under test																									
P1	10	20																									
P2	100	50																									
P3	40	20																									
P4	10	5																									
P5	60	30																									
(b)	Explain the operation of stack with example. Write the line of code for implementing the same.						[5]	CO1	L2																		
6 (a)	What is an interrupt? List the sequence of events involved in handling an interrupt request from a single device.						[5]	CO2	L2																		
(b)	Write about shift and rotate instructions with neat diagram and example of each.						[5]	CO1	L3																		
7 (a)	Explain the following methods of handling interrupts from multiple devices a) Interrupt Nesting/Priority Structure b) Daisy Chain Method						[10]	CO2	L2																		
8 (a)	With a neat diagram, explain I/O interface for an I/O devices. Explain various registers involved in it.						[5]	CO2	L2																		
(b)	What are Condition Code flags? Explain the four commonly used flags.						[5]	CO1	L2																		

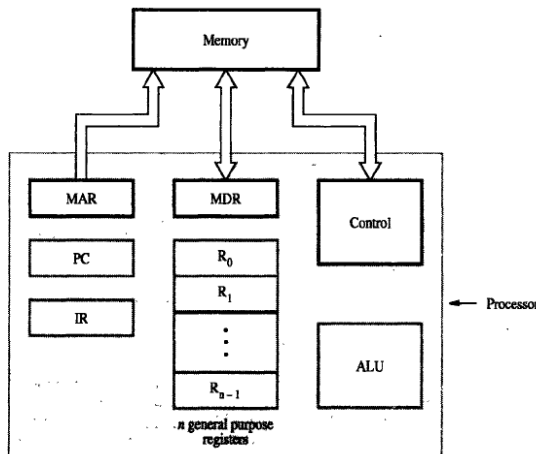
Scheme of Solution

Internal Assessment Test 1 – September 2019

Sub:	COMPUTER ORGANIZATION	Code:	18CS44
Date:	07/09/2019	Duration:	90 mins
		Max Marks:	50
		Sem:	III
		Branch:	ISE
Answer Any FIVE FULL Questions			

1 (a) Explain the steps involved in instruction fetching from memory and execution in the processor with the help of block diagram. 5 Marks

- Transfer the contents of register PC to register MAR
 - Issue a Read command to memory, and then wait until it has transferred the requested word into register MDR
 - Transfer the instruction from MDR into IR and decode it
 - Transfer the address LOCA from IR to MAR
 - Issue a Read command and wait until MDR is loaded
 - Content of the PC are incremented
- 3 Marks



2 Marks

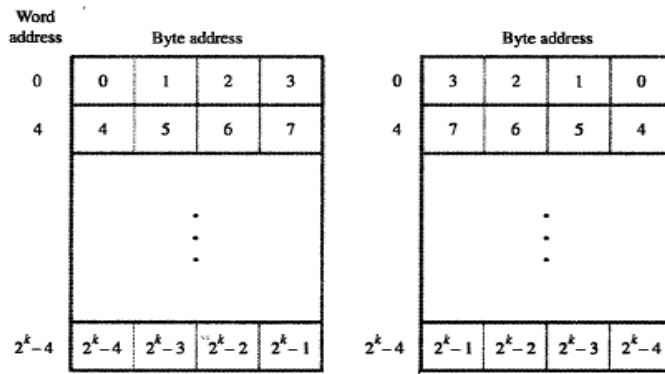
1(b) (Define the Little Endian and Big Endian assignments. Explain with relevant diagrams

Two ways – Big Endian and Little Endian

Big Endian – used when lower byte addresses used for MSB of the word

Little Endian – used when lower byte addresses used for LSB of the word

2 Marks



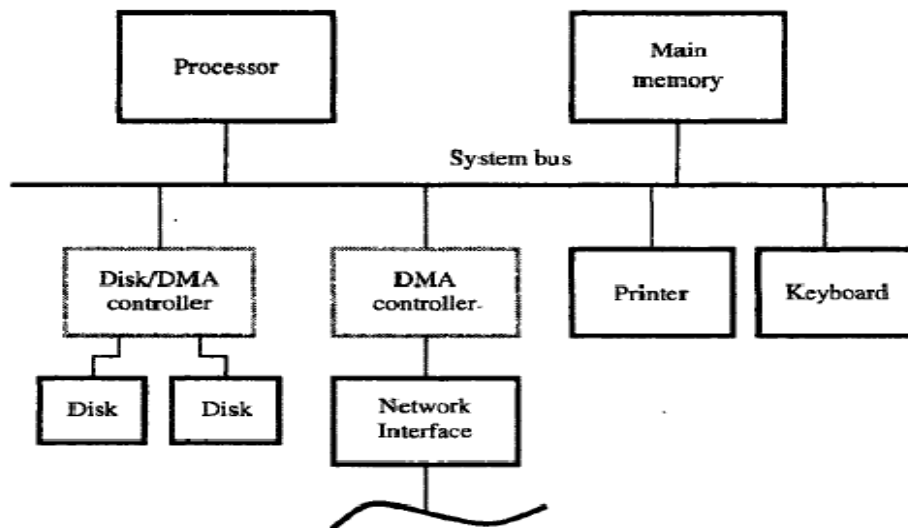
(a) Big-endian assignment

(b) Little-endian assignment

3 Marks

2(a) What is the functional operation of DMA controller? Explain the centralized arbitration and distributed bus arbitration schemes. 10 Marks

- To transfer large blocks of data at high speed, an alternative approach is used
- A special control unit may be provided to allow transfer of a block of data directly between an external device and the main memory, without continuous intervention by the processor. The approach is called Direct Memory Access (DMA)
- DMA controller performs the functions that would normally be carried out by the processor when accessing the main memory. 2 Marks



3 Marks

Two approaches

- Centralized Arbitration – A single bus arbitration performs the required arbitration
- Distributed Arbitration – All devices participate in the selection of the next bus master

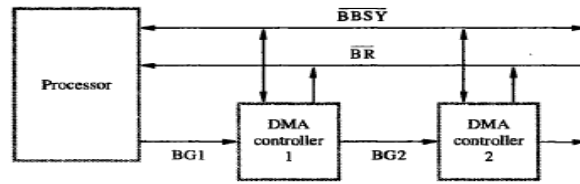


Figure 4.20 A simple arrangement for bus arbitration using a daisy chain.

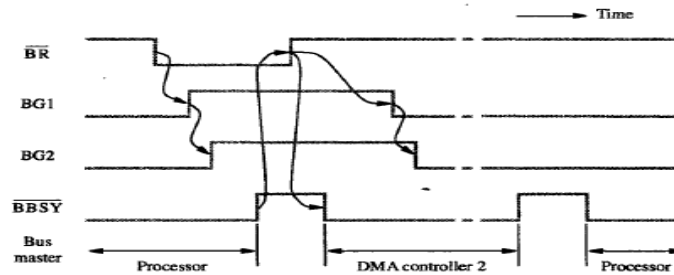
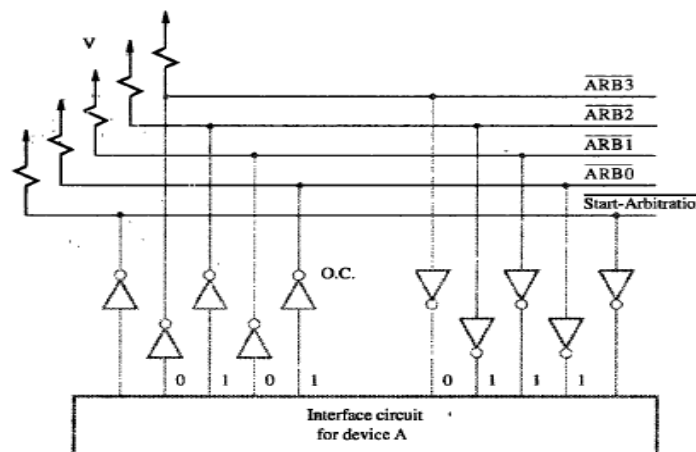


Figure 4.21 Sequence of signals during transfer of bus mastership for the devices in Figure 4.20.

3 Marks



2 Marks

3(a) Define Subroutine. How to pass parameters to subroutine? Explain with Example.
10 Marks

In a program subtasks that are repeated on different data values are usually implemented as subroutines. When a program requires the use of a subroutine, it branches to the subroutine. Branching to the subroutine is called as “calling” the subroutine.

Instruction that performs this branch operation is Call.

After a subroutine completes execution, the calling program continues with executing the instruction immediately after the instruction that called the subroutine.

Subroutine is said to “return” to the program.

Instruction that performs this is called Return.

Subroutine may be called from many places in the program.

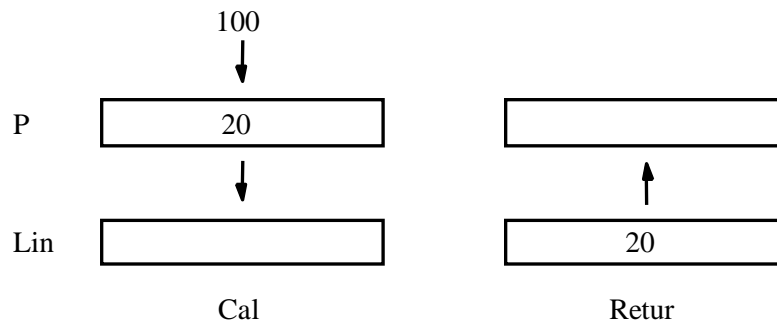
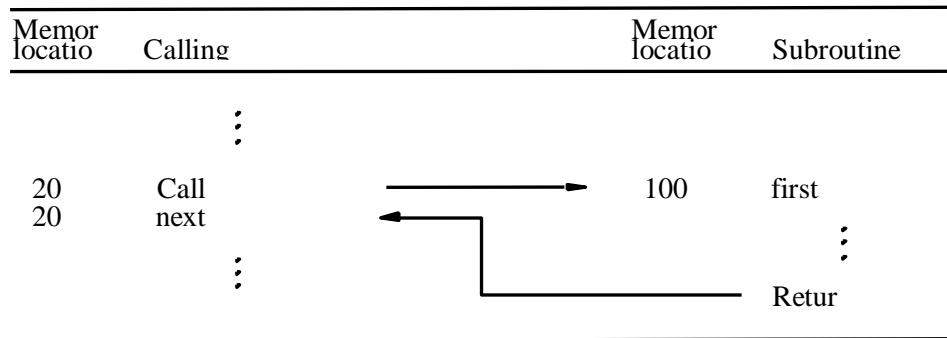
3 Marks

Example:

- Calling program calls a subroutine, whose first instruction is at address 1000.
- The Call instruction is at address 200.
- While the Call instruction is being executed, the PC points to the next instruction at address 204.

- Call instructions stores address 204 in the Link register, and loads 1000 into the PC.
- Return instruction loads back the address 204 from the link register into the PC.

2 Marks



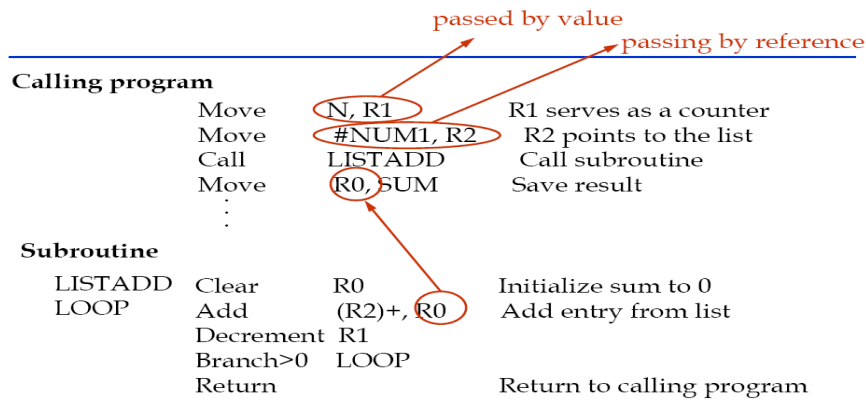
Parameter Passing:

When calling a subroutine, a program must provide to the subroutine the **parameters**, that is, the **operands** or their addresses, to be used in the computation. Later, the subroutine **returns** other parameters, in this case, the **result of computation**. The exchange of information between a calling program and a subroutine is referred to as parameter passing

Parameter passing approaches

- ◆ The parameters may be placed in registers or in memory locations, where they can be accessed by the subroutine
- ◆ The parameters may be placed on the processor stack used for saving the return address

3 Marks



2 Marks

4(a) Explain any five addressing modes with example of each mode. 10 Marks

Name	Assembler syntax	Addressing function
Immediate	#Value	Operand=Value
Register	Ri	EA=Ri
Absolute (Direct)	LOC	EA=LOC
Indirect	(Ri)	EA=[Ri]
	(LOC)	EA=[LOC]
Index	X(Ri)	EA=[Ri]+X
Base with index	(Ri, Rj)	EA=[Ri]+[Rj]
Base with index and offset	X(Ri, Rj)	EA=[Ri]+[Rj]+X
Relative	X(PC)	EA=[PC]+X
Autoincrement	(Ri)+	EA=[Ri]; Increment Ri
Autodecrement	-(Ri)	Decrement Ri; EA=[Ri]

Any five addressing modes

5 Marks

Example of each addressing modes

5 Marks

5(a) Calculate the SPEC rating for the program suite under test. Running times of the program suite for reference PC and PC under test are given below:

5 Marks

Programs	Running time for reference PC	Running time for PC under test
P1	10	20
P2	100	50
P3	40	20
P4	10	5
P5	60	30

$P1 = 10/20 = 0.5$ 1 mark

$P2 = 100/50 = 2$ 1 mark

$P3 = 40/20 = 2$ 1 mark

$P4 = 10/5 = 2$ 1 mark

$P5 = 60/30 = 2$ 1 mark

Overall SPEC rating = 1.517

5 marks

5(b) Explain the operation of stack with example. Write the line of code for implementing the same. 5 Marks

A stack is a list of data elements, usually words or bytes with the accessing restriction that elements can be added or removed at one end of the stack. End from which elements are added and removed is called the “top” of the stack. Other end is called the “bottom” of the stack. Also known as: Push down stack. Last in first out (LIFO) stack. *Push* - placing a new item onto the stack. *Pop* - Removing the top item from the stack. 3 Marks

Example:

- Processor with 65536 bytes of memory.
- Byte addressable memory.
- Word length is 4 bytes.
- First element of the stack is at BOTTOM.
- SP points to the element at the top.
- Push operation can be implemented as:

Subtract #4, SP

Move A, (SP)

- Pop operation can be implemented as:

Move (SP), B

Add #4, SP

- Push with autodecrement:

Move A, -(SP)

- Pop with autoincrement:

Move (SP)+, A

2 Marks

6 (a). What is an interrupt? List the sequence of events involved in handling an interrupt request from a single device. 5 Marks

An approach for the I/O device to alert the processor when it becomes ready. It is done by sending a hardware signal called an interrupt to the processor. At least one of the bus control lines, called an interrupt-request line is dedicated for this purpose. 2 Marks

The device raises an interrupt request

The processor interrupts the program currently being executed

Interrupts are disabled by changing the control bits in the Processor Status (PS) register

The device is informed that its request has been recognized, and in response, it deactivates the interrupt request signal

The action requested by the interrupt is performed by the interrupt-service routine

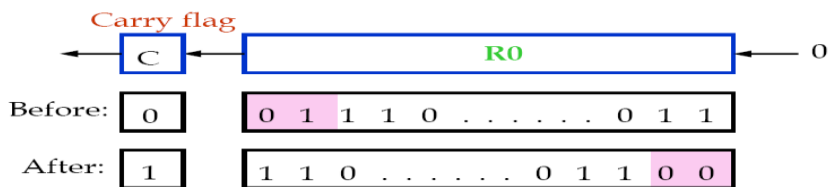
Interrupts are enabled and execution of the interrupted program is resumed 3 Marks

6 (b) Write about shift and rotate instruction with neat diagram and example of each. 5 Marks

➤ Logical shifts

Logic shift left

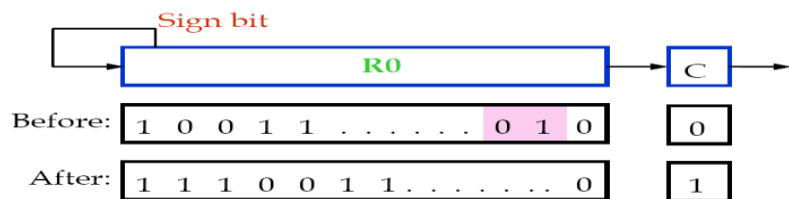
LShiftL #2, R0



➤ Arithmetic shifts

shift right

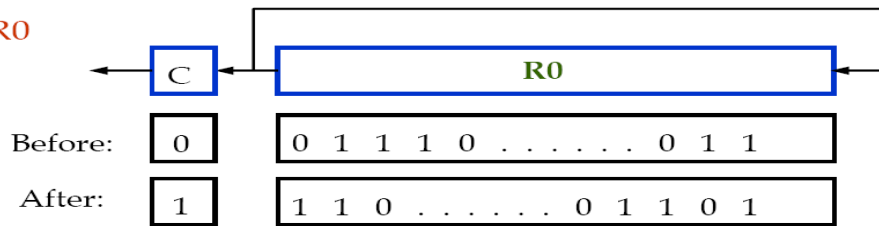
AShiftR #2, R0



3 Marks

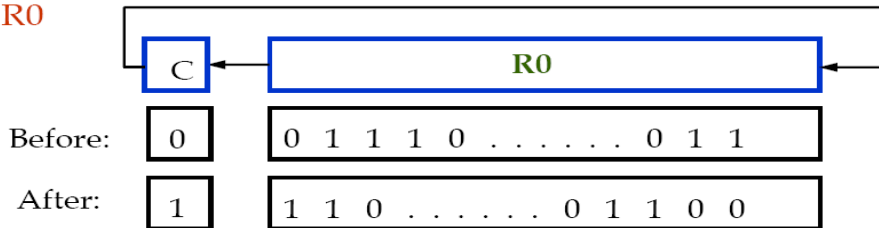
➤ Rotate left without carry

RotateL #2, R0



➤ Rotate left with carry

RotateLC #2, R0



2 Marks

7 (a) Explain the following methods of handling interrupts from multiple devices

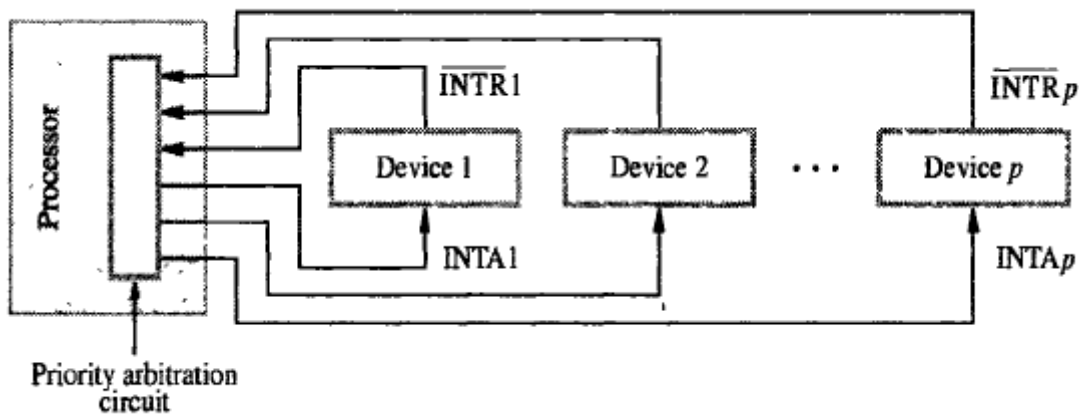
Interrupt Nesting/Priority Structure b) Daisy Chain Method

10 Marks

Interrupt Nesting

- A multiple level priority organization means that during execution of an interrupt service routine, interrupt will be accepted from some device but not from others, depending upon the devices priority

- There are privileged instructions which can be executed only while the processor is running in the supervisor mode. 3 Marks



2 Marks

Daisy Chain

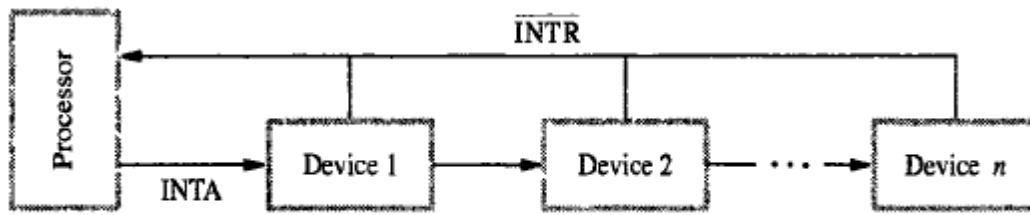
- Widely used scheme to connect the device is called Daisy Chain

- The interrupt request is common to all devices.

- The interrupt acknowledge line INTA is connected serially through the device

- It is electrically closest to the processor has the highest priority

3 Marks



(a) Daisy chain

2 Marks

8(a) With a neat diagram, explain I/O interface for an I/O device. Also, explain various registers involved in it. 5 Marks

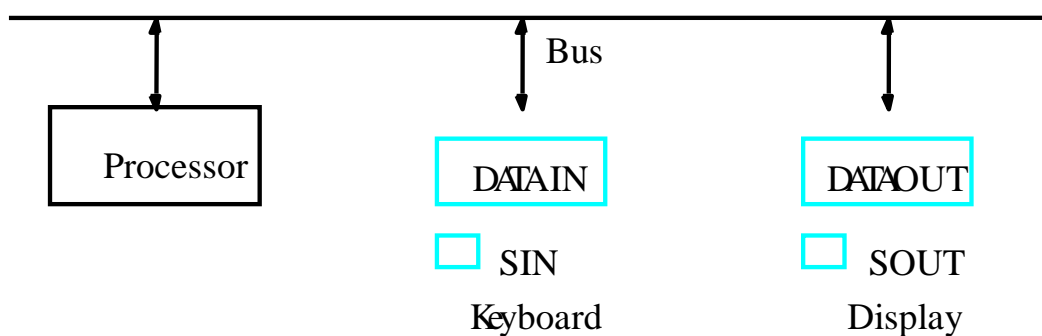
Input:

- When a key is struck on the keyboard, an 8-bit character code is stored in the buffer register DATAIN.
- A status control flag SIN is set to 1 to indicate that a valid character is in DATAIN.
- A program monitors SIN, and when SIN is set to 1, it reads the contents of DATAIN.
- When the character is transferred to the processor, SIN is automatically cleared.
- Initial state of SIN is 0.

Output:

- When SOUT is equal to 1, the display is ready to receive a character.
- A program monitors SOUT, and when SOUT is set to 1, the processor transfers a character code to the buffer DATAOUT.
- Transfer of a character code to DATAOUT clears SOUT to 0.
- Initial state of SOUT is 1.

3 Marks



2 Marks

8 (b) What are Condition Code flags? Explain the four commonly used flags. 5 Marks

- The processor keeps track of information about the results of various operations for use by subsequent conditional branch instructions.

- This is accomplished by recording required information in individual bits, often called condition code flags
- Four commonly used flags are
 - N (negative): set to 1 if the results is negative; otherwise, cleared to 0
 - Z (zero): set to 1 if the result is 0; otherwise, cleared to 0
 - V (overflow): set to 1 if arithmetic overflow occurs; otherwise, cleared to 0
 - C (carry): set to 1 if a carry-out results from the operation otherwise, cleared to 0
- N and Z flags caused by an arithmetic or a logic operation,
- V and C flags caused by an arithmetic operation

5 Marks
