Instruction Codes:-

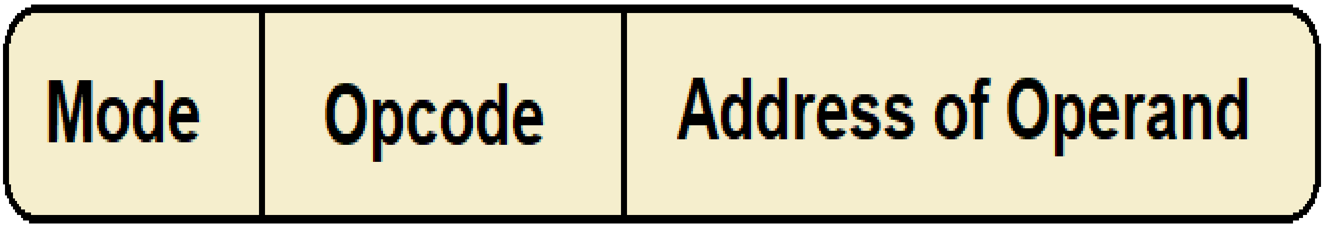
Computer instruction is a binary code that specifies the micro-operation s in a sequence stored in the computer's memory and the data. Instruction codes and addresses are unique to each computer.

Generally, we can categorize instruction codes and addresses into operation codes (Opcodes) and addresses. Opcodes specify how to perform a specific instruction. An address specifies which we should use- a register or an area, for a particular action. Operands are precise computer instructions that show what information the computer requires to function.

Instruction code

Instruction codes are bits that instruct the computer to execute a specific operation. An instruction comprises groups called fields. These fields include:

- The Operation code (Opcode) field determines the process that needs to perform.
- The Address field contains the operand's location, i.e., register or memory location.
- The Mode field specifies how the operand locates.



Mode	Opcode	Address of Operand

Opcodes:-

An opcode is a collection of bits representing the basic operations, including add, subtract, multiply, complement, and shift. The number of bits required for the opcode is determined by the number of functions the computer gives. For '2^n' operations, the minimum bits accessible to the opcode should be 'n', where n is the number of bits. We implement these operations on information saved in processor registers or memory.

Types of Opcodes:-

There are three different types of instruction codes on the main computer. The instruction's operation code (opcode) is 3 bits long, and the remaining 13 bits are determined by the operation code encountered. There are three types of formats:

1. Memory Reference Instruction: It specifies the address with 12 bits and the addressing mode with 1 bit (I). For direct addresses, I equal 0, while for indirect addresses, I equal 1.
2. Register Reference Instruction: The opcode 111 with a 0 in the leftmost bit of the instruction recognizes these instructions. The remaining 12 bits specify the procedure to be carried out.
3. Input-Output Instruction: The operation code 111 with a 1 in the leftmost bit of instruction recognizes these instructions. The input-output action is specified using the remaining 12 bits.

Computer Registers:-

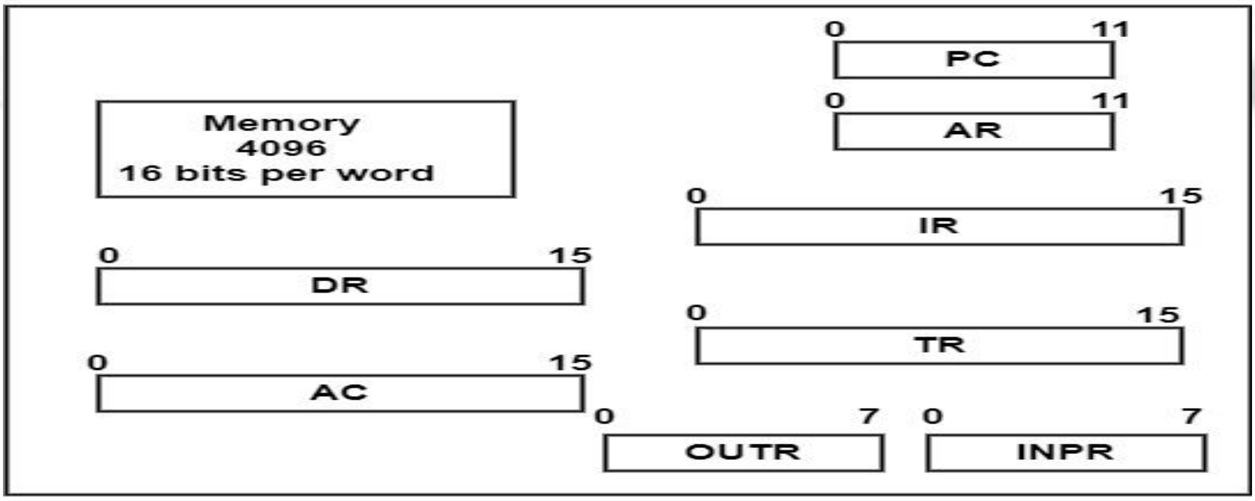
Computer registers are high-speed memory storing units. It is an element of the computer processor. It can carry any type of information including a bit sequence or single data.

A register should be 32 bits in length for a 32-bit instruction computer. Registers can be numbered relies upon the processor design and language rules.

The instructions in a computer are saved in memory locations and implemented one after another at a time. The function of the control unit is to fetch the instruction from the memory and implement it. The control does the similar for all the instructions in the memory in sequential order.

A counter is needed to maintain a path of the next instruction to be implemented and evaluate its address. The figure shows the registers with their memories. The memory addresses are saved in multiple registers. These requirements certainly state the use for registers in a computer.

Basics Registers and Memory



The following table shows the registers and their functions.

Register Symbol	Number of Bits	Register Name	Function
OUTR	8	Output register	It holds output character.
INPR	8	Input register	It holds input character.
PC	12	Program Counter	It holds the address of the instruction.
AR	12	Address Register	It holds an address for memory.
DR	16	Data Register	It holds memory operand.
AC	16	Accumulator	It's a processor register.
IR	16	Instruction Register	It holds an instruction code.
TR	16	Temporary Register	It holds temporary data.

The description for each of the registers determined in the figure is as follows −

- The data register holds the operand read from the memory.
- The accumulator is a general-purpose register need for processing.
- The instruction register holds the read memory.
- The temporary data used while processing is stored in the temporary register.

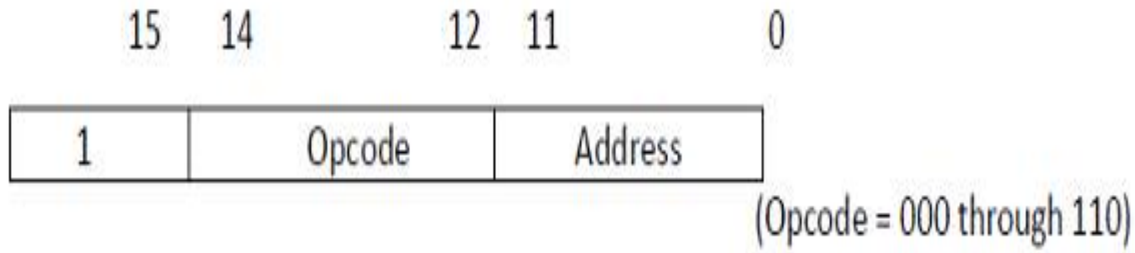
- The address register holds the address of the instruction that is to be implemented next from the memory.
- The input register (INPR) and output register (OUTPR) are the registers used for the I/O operations. The INPR receives an 8-bit character from the input device. It is similar to the OUTPR.

Computer Instructions:-

A computer has programs stored in its RAM in the form of 1s and 0s that are interpreted by the CPU as instructions. One word of RAM includes one instruction in the machine language. These instructions are loaded to the CPU one at a time, where it receives decoded and implemented. A basic computer has three instruction code formats such as the memory reference instruction, the register reference instruction, and the input-output instruction format.

Memory Reference Instruction

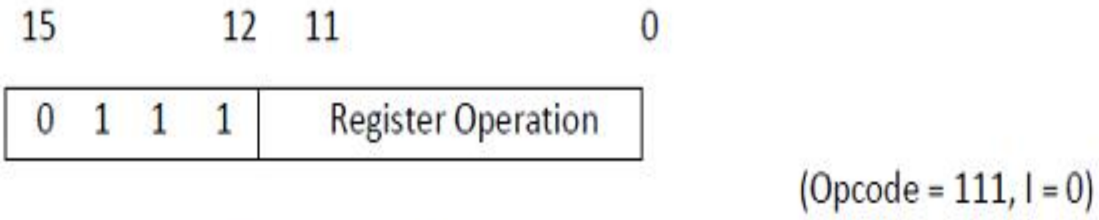
A memory-reference instruction uses 12 bits to specify an address and one bit to determine the addressing mode I. I is the same as 0 for direct address and to 1 for indirect address.



(a) Memory Reference Instruction

Register Reference Instruction:-

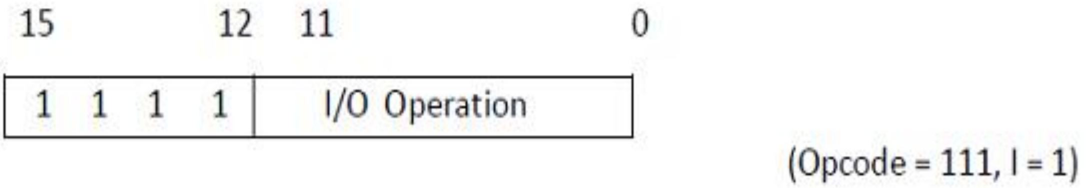
The register reference instructions are identified by the operation code 111 with a 0 in the leftmost bit (bit 15) of the instruction. It determines an operation on or a test of the AC register. An operand from memory is not required because the additional 12 bits are used to determine the operation or test to be implemented.



(b) Register Reference Instruction

Input-Output Instruction

An input-output instruction does not require a reference to memory and is identified by the operation code 111 with a 1 in the leftmost bit of the instruction. The remaining 12 bits can determine the type of input-output operation or test implemented.



(c) Input-Output Instruction

The type of instruction is identified by the computer control from the four bits in positions 12 through 15 of the instruction. If the three opcode bits in positions 12 through 14 are not similar to 111, the instruction is a memory-reference type and the bit in position 15 is taken as the addressing mode I.

If the 3-bit opcode is similar to 111, the control then examines the bit in position 15. If this bit is 0, the instruction is a register-reference type. If the bit is 1, the instruction is an input-output type.

Timing and Control

The timing for all registers in the basic computer is controlled by a master clock generator. The clock pulses are applied to all flip-flops and registers in the system, including the flip-flops and registers in the control unit. The clock pulses do not change the state of a register unless the register is enabled by a control signal. The control signals are generated in the control unit and provide control inputs for the multiplexers in the common bus, control inputs in processor registers, and micro operations for the accumulator.

There are two major types of control organization:

- Hardwired control and
- Micro programmed control.

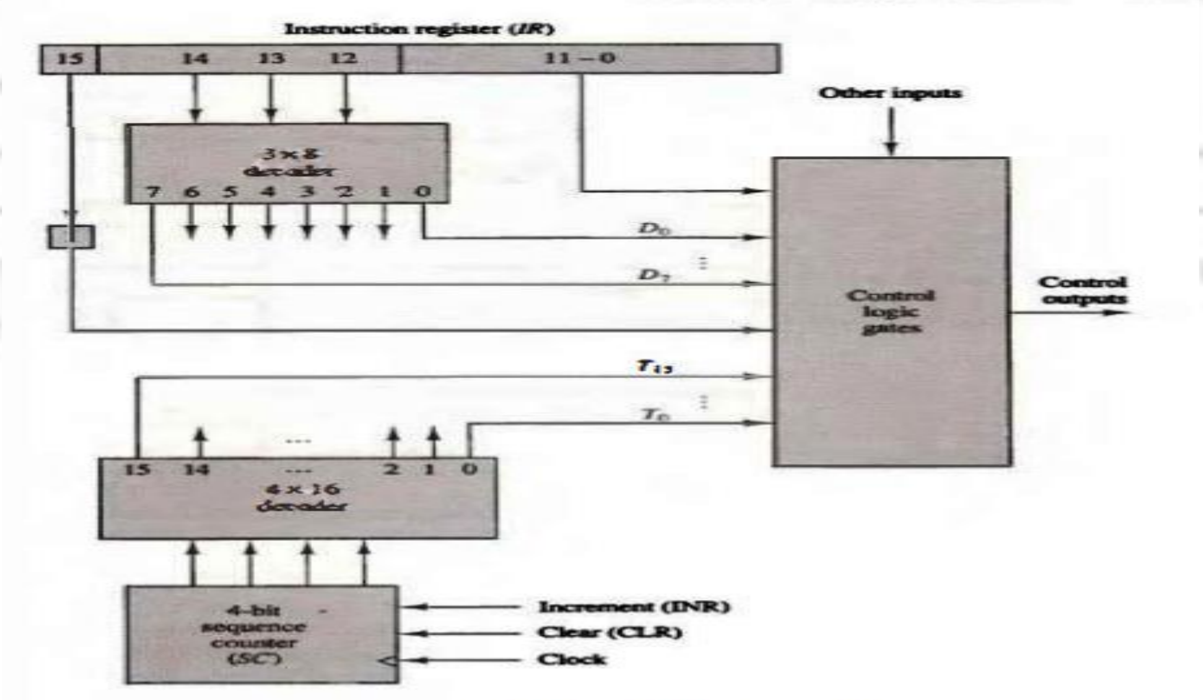
In the hardwired organization, the control logic is implemented with gates, flip-flops, decoders, and other digital circuits. It has the advantage that it can be optimized to produce a fast mode of operation. In the micro programmed organization, the control information is stored in a control memory. The control memory is programmed to initiate the required sequence of micro operations. A hardwired control, as the name implies, requires changes in the wiring among the various components if the design has to be modified or changed.

In the micro programmed control, any required changes or modifications can be done by updating the micro program in control memory.

It consists of two decoders,

1. A sequence counter, and
2. A number of control logic gates.

An instruction read from memory is placed in the instruction register (IR).position of this register in the common bus system is indicated in the block diagram of the control unit is shown in Fig.



Control unit of basic computer.

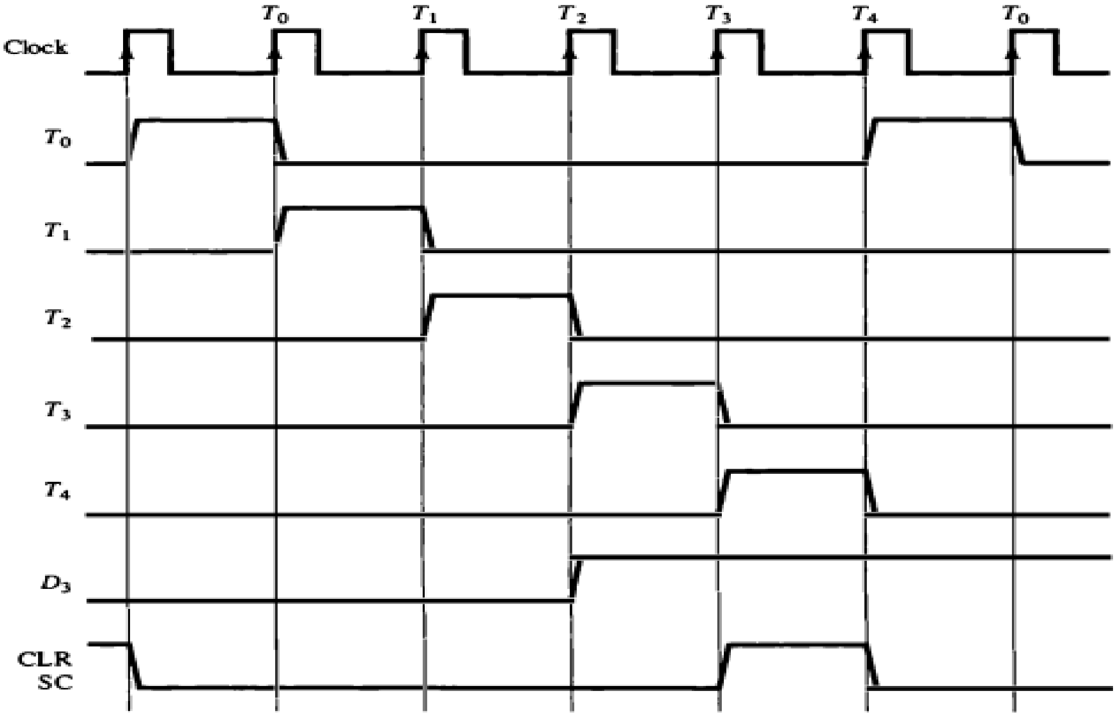
The instruction register is shown again in Fig. where it is divided into three parts:

1. The 1 bit,
2. The operation code, and
3. Bits 0 through 11.

The operation code in bits 12 through 14 are decoded with a 3 x 8 decoder. The eight outputs of the decoder are designated by the symbols D0 through D7. The subscripted decimal number is equivalent to the binary value of the corresponding operation code. Bit 15 of the instruction is transferred to a flip-flop designated by the symbol I. Bits 0 through 11 are applied to the control logic gates. The 4-bit sequence counter can count in binary from 0 through 15. The outputs of the counter are decoded into 16 timing signals T0 through T15.

The sequence counter SC can be incremented or cleared synchronously. Most of the time, the counter is incremented to provide the sequence of timing signals out of the 4 x 16 decoder. Once in a while, the counter is cleared to 0, causing the next active timing signal to be T0.

As an example, consider the case where SC is incremented to provide timing signals T0, T1, T2, T3, and T4 in sequence. At time T4, SC is cleared to 0 if decoder output D3 is active. This is expressed symbolically by the statement



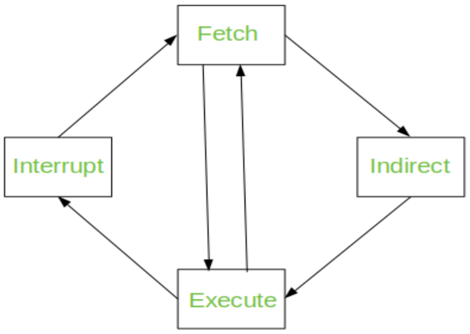
Example of control timing signals.

Instruction cycle:-

A program consisting of the memory unit of the computer includes a series of instructions. The program is implemented on the computer by going through a cycle for each instruction.

In the basic computer, each instruction cycle includes the following procedures –

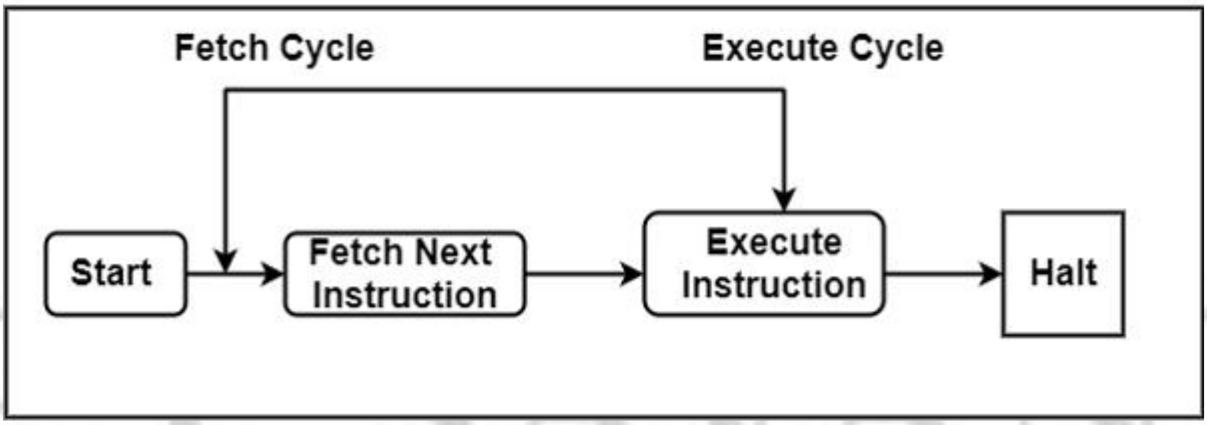
1. It can fetch instruction from memory.
2. It is used to decode the instruction.
3. It can read the effective address from memory if the instruction has an indirect address.
4. It can execute the instruction.



The Instruction Cycle

After the following four procedures are done, the control switches back to the first step and repeats the similar process for the next instruction. Therefore, the cycle continues until a Halt condition is met. The figure shows the phases contained in the instruction cycle.

Instruction Cycle



As display in the figure, the halt condition appears when the device receive turned off, on the circumstance of unrecoverable errors, etc.

Fetch Cycle

The address instruction to be implemented is held at the program counter. The processor fetches the instruction from the memory that is pointed by the PC.Next; the PC is incremented to display the address of the next instruction. This instruction is loaded onto the instruction register. The processor reads the instruction and executes the important procedures.

Execute Cycle

The data transfer for implementation takes place in two methods is as follows −

Processor-memory − the data sent from the processor to memory or from memory to processor.

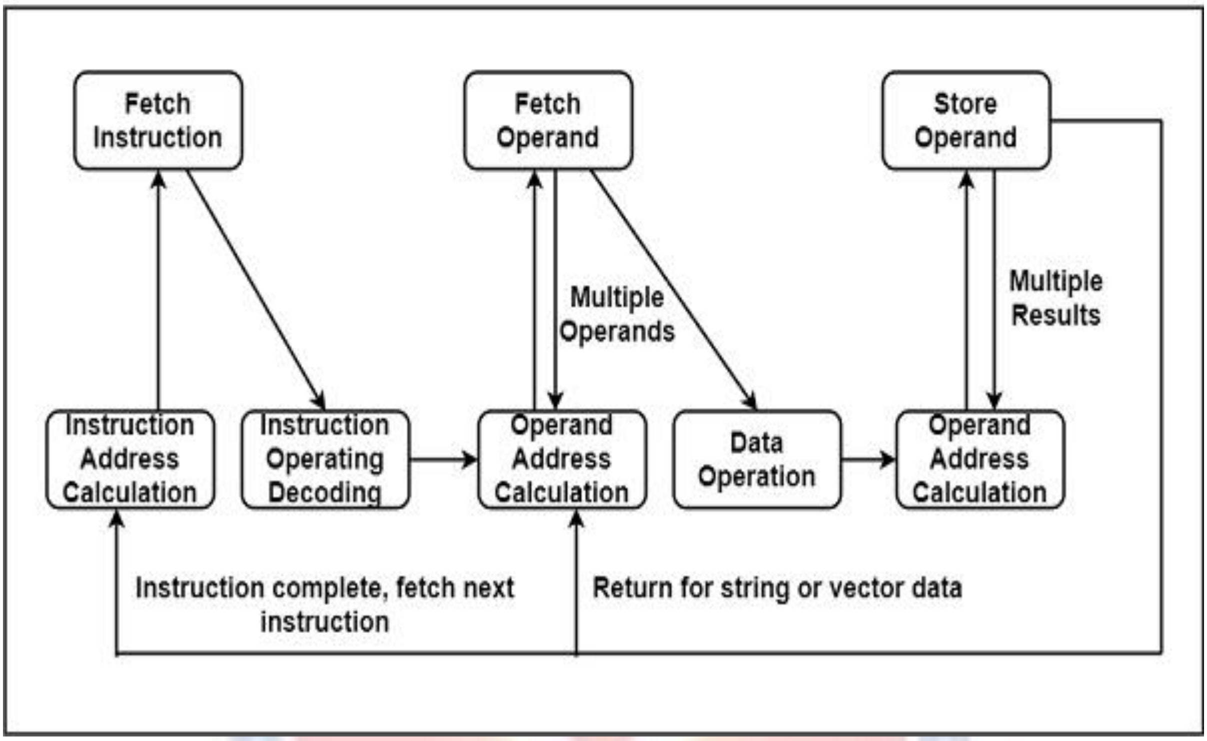
Processor-Input/output − the data can be transferred to or from a peripheral device by the transfer between a processor and an I/O device.

In the execute cycle, the processor implements the important operations on the information, and consistently the control calls for the modification in the sequence of data implementation. These two methods associate and complete the execute cycle.

State Diagram for Instruction Cycle

The figure provides a large aspect of the instruction cycle of a basic computer, which is in the design of a state diagram. For an instruction cycle, various states can be null, while others can be visited more than once.

Instruction Cycle State Diagram


Fetch Instruction
Fetch Operand
Store Operand
Multiple Operands
Multiple Results
Instruction Address Calculation
Instruction Operating Decoding
Operand Address Calculation
Data Operation
Operand Address Calculation
Instruction complete, fetch next instruction
Return for string or vector data

Instruction Address Calculation − the address of the next instruction is computed. A permanent number is inserted to the address of the earlier instruction.

Instruction Fetch − the instruction is read from its specific memory location to the processor.

Instruction Operation Decoding − the instruction is interpreted and the type of operation to be implemented and the operand(s) to be used are decided.

Operand Address Calculation − the address of the operand is evaluated if it has a reference to an operand in memory or is applicable through the Input/output.

Operand Fetch − the operand is read from the memory or the I/O.

Data Operation − the actual operation that the instruction contains is executed.

Store Operands − It can store the result acquired in the memory or transfer it to the I/O.

Memory Reference Instructions:-

Memory reference instructions are those commands or instructions which are in the custom to generate a reference to the memory and approval to a program to have an approach to the commanded information and that states as to from where the data is cache continually. These instructions are known as Memory Reference Instructions.

There are seven memory reference instructions which are as follows &

AND: - The AND instruction implements the AND logic operation on the bit collection from the register and the memory word that is determined by the effective address. The result of this operation is moved back to the register.

ADD: - The ADD instruction adds the content of the memory word that is denoted by the effective address to the value of the register.

LDA (Load Accumulator): - The LDA instruction shares the memory word denoted by the effective address to the register.

STA (STore Accumulator):- STA saves the content of the register into the memory word that is defined by the effective address. The output is next used to the common bus and the data input is linked to the bus. It needed only one micro-operation.

Note- **STA is for copying data from accumulator to memory location, LDA is for copying data from memory location to accumulator.**

BUN (Branch unconditionally): - The Branch Unconditionally (BUN) instruction can send the instruction that is determined by the effective address. They understand that the address of the next instruction to be performed is held by the PC and it should be incremented by one to receive the address of the next instruction in the sequence. If the control needs to implement multiple instructions that are not next in the sequence, it can execute the BUN instruction.

BSA: - BSA stands for Branch and save return Address. These instructions can branch a part of the program (known as subroutine or procedure). When this instruction is performed, BSA will store the address of the next instruction from the PC into a memory location that is determined by the effective address.

ISZ: - he Increment if Zero (ISZ) instruction increments the word determined by effective address. If the incremented cost is zero, thus PC is incremented by 1. A negative value is saved in the memory word through the programmer. It can influence the zero value after getting incremented repeatedly. Thus, the PC is incremented and the next instruction is skipped.

Input-output and interrupt:-

The method that is used to transfer information between internal storage and external I/O devices is known as I/O interface. The CPU is interfaced using special communication links by the peripherals connected to any computer system. These communication links are used to resolve the differences between CPU and peripheral. There exists special hardware components between CPU and peripherals to supervise and synchronize all the input and output transfers that are called interface units.

Mode of Transfer:-

The binary information that is received from an external device is usually stored in the memory unit. The information that is transferred from the CPU to the external device is originated from the memory unit. CPU merely processes the information but the source and target is always the memory unit. Data transfer between CPU and the I/O devices may be done in different modes.

Data transfer to and from the peripherals may be done in any of the three possible ways

1. Programmed I/O.
2. Interrupt- initiated I/O.

3. Direct memory access (DMA).

Programmed I/O:- It is due to the result of the I/O instructions that are written in the computer program. Each data item transfer is initiated by an instruction in the program. Usually the transfer is from a CPU register and memory. In this case it requires constant monitoring by the CPU of the peripheral devices.

Interrupt- initiated I/O:- Since in the above case we saw the CPU is kept busy unnecessarily. This situation can very well be avoided by using an interrupt driven method for data transfer. By using interrupt facility and special commands to inform the interface to issue an interrupt request signal whenever data is available from any device. In the meantime the CPU can proceed for any other program execution. The interface meanwhile keeps monitoring the device. Whenever it is determined that the device is ready for data transfer it initiates an interrupt request signal to the computer. Upon detection of an external interrupt signal the CPU stops momentarily the task that it was already performing, branches to the service program to process the I/O transfer, and then return to the task it was originally performing.

Note:

Both the methods programmed I/O and Interrupt-driven I/O require the active intervention of the processor to transfer data between memory and the I/O module, and any data transfer must transverse a path through the processor. Thus both these forms of I/O suffer from two inherent drawbacks.

1. The I/O transfer rate is limited by the speed with which the processor can test and service a device.

2. The processor is tied up in managing an I/O transfer; a number of instructions must be executed for each I/O transfer.

Direct Memory Access:- The data transfer between a fast storage media such as magnetic disk and memory unit is limited by the speed of the CPU. Thus we can allow the peripherals directly communicate with each other using the memory buses, removing the intervention of the CPU. This type of data transfer technique is known as DMA or direct memory access. During DMA the CPU is idle and it has no control over the memory buses. The DMA controller takes over the buses to manage the transfer directly between the I/O devices and the memory unit.

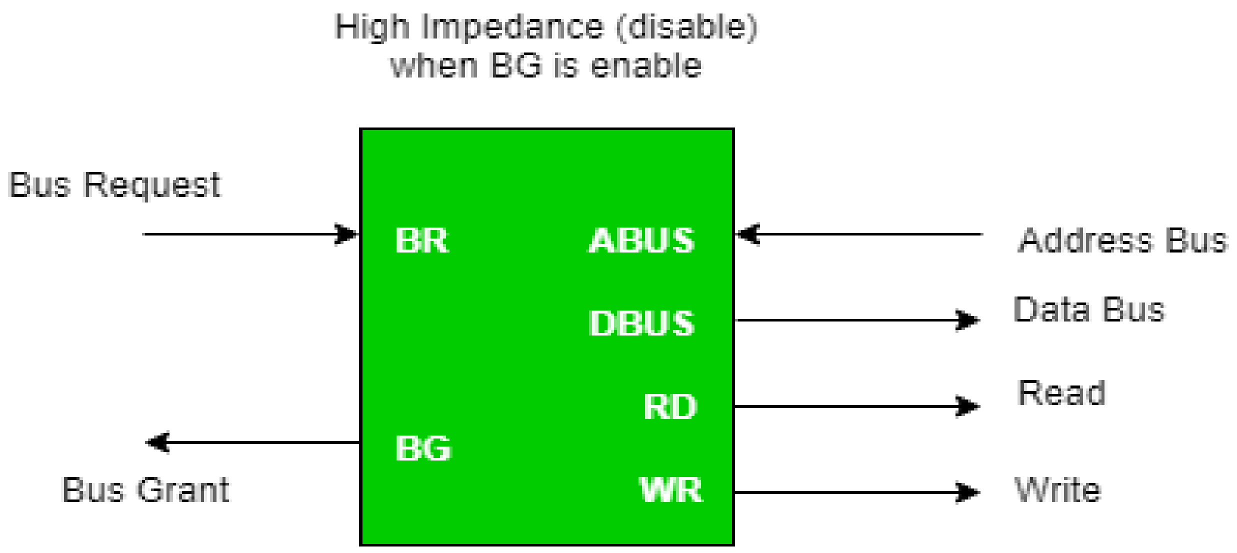


Figure - CPU Bus Signals for DMA Transfer

Bus Request:- It is used by the DMA controller to request the CPU to relinquish the control of the buses.

Bus Grant: - It is activated by the CPU to inform the external DMA controller that the buses are in high impedance state and the requesting DMA can take control of the buses. Once the DMA has taken the control of the buses it transfers the data. This transfer can take place in many ways.

Thank You