At the beginning of each instruction cycle, the processor fetches an instruction from memory. In a typical processor, a register called the program counter (PC) holds the address of the instruction to be fetched next. Unless told otherwise, the processor always increments the PC after each instruction fetch so that it will fetch the next instruction in sequence. The fetched instruction is loaded into a register in the processor known as the instruction register (IR). The instruction contains bits that specify the action the processor is to take. The processor interprets the instruction and performs the required action.

Execute Cycle:

In general, the required actions fall into four categories:

- Processor-memory: Data may be transferred from processor to memory or from memory to processor.
- Processor-I/O: Data may be transferred to or from a peripheral device by transferring between the processor and an I/O module.
- Data processing: The processor may perform some arithmetic or logic operation on data.
- Control: An instruction may specify that the sequence of execution be altered.
- An instruction’s execution may involve a combination of these actions.

Instruction Cycle State Diagram:

Figure 4 provides a more detailed look at the basic instruction cycle. The figure is in the form of a state diagram. For any given instruction cycle, some stales may be null and others may be visited more than once. The states can be described as follows:

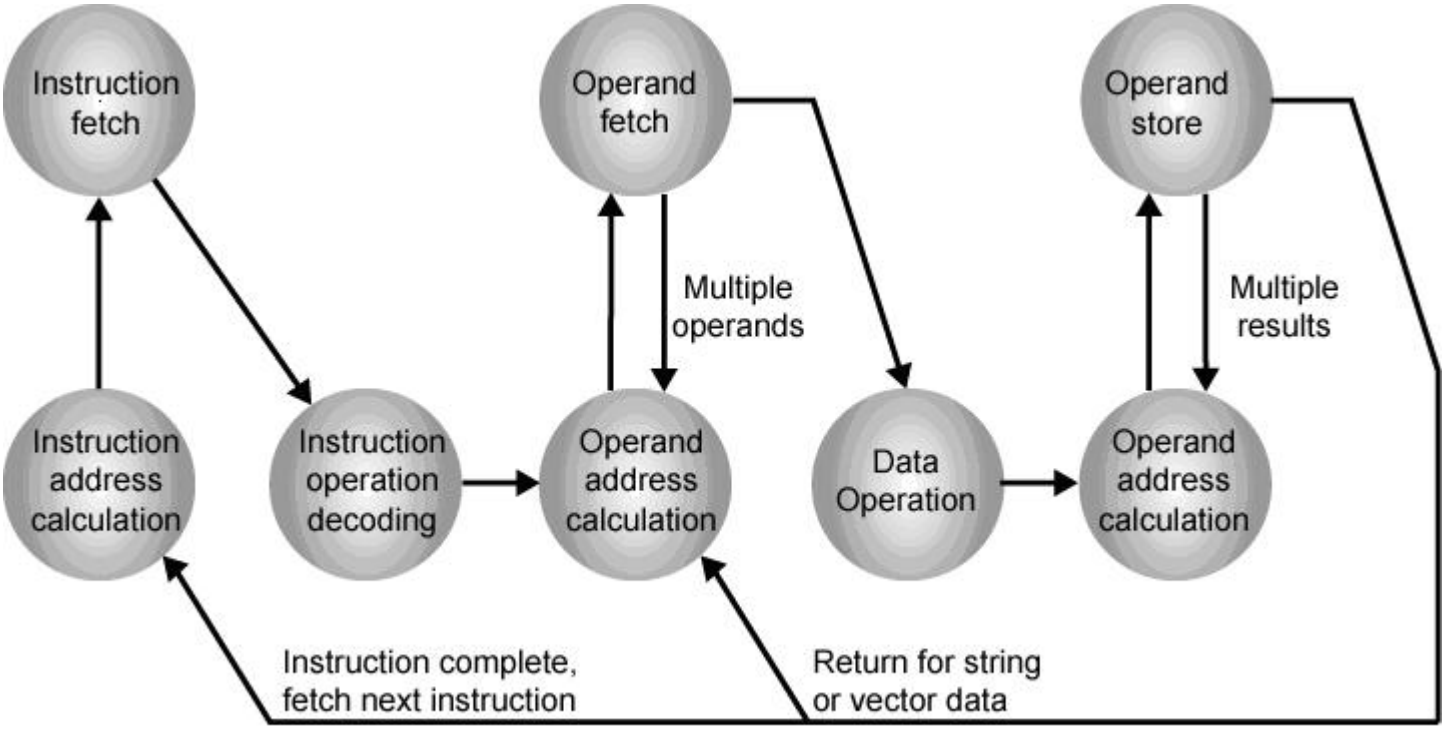

Figure 4: Instruction cycle state diagram

- Instruction address calculation (iac): Determine the address of the next instruction to be executed. Usually, this involves adding a fixed number to the address of the previous instruction. For example, if each instruction is 16 bits long and memory is organized into 16-bit words, then add 1 to the previous address. If, instead, memory is organized as individually addressable 8-bit bytes, then add 2 to the previous address.
- Instruction fetch (if): Read instruction from its memory location into the processor.
- Instruction operation decoding (iod): Analyze instruction to determine type of operation to he performed and operand(s) to be used.
- Operand address calculation (oac): If the operation involves reference to an operand in memory or available via I/O. then determine the address of the operand.
- Operand fetch (of): Fetch the operand from memory or read it in from I/O,
- Data operation (do): Perform the operation indicated in the instruction.
- Operand store (os): Write the result into memory or out to I/O

2.2 Interrupts

Virtually all computers provide a mechanism called Interrupt, by which other modules (I/O. memory) may interrupt the normal processing of the processor. Interrupts are provided primarily as a way to improve processing efficiency.

For example, most external devices are much slower than the processor. Suppose that the processor is transferring data to a printer using the instruction cycle scheme of Figure 3. After each write operation, the processor must pause and remain idle until the printer catches up. The length of this pause may be on the order of many hundreds or even

thousands of instruction cycles that do not involve memory. Clearly, this is a very wasteful use of the processor. The figure 5a illustrates this state of affairs.