

CIT309 - COMPUTER ARCHITECTURE SUMMARY

Q. Explain the operational units of a computer system.

Computer organization refers to the operational units and their interconnection that realize the architectural specification. Examples of architectural attributes include the instruction set, the number of bit used to represent various data types (e. g numbers, characters), I/O mechanism, and techniques for addressing memory. Organizational attributes include those hardware details transparent to the programmer, such as control signals; interfaces between the computer peripherals and memory technology used. Outline types of operands and operations specific by machine instruction.

Q. Explain opcodes, operands and addressing modes

Opcodes

Within the computer, each instruction is represented by a sequence of bits. The instruction is divided into fields, corresponding to the constituents elements of the instruction Opcodes are represented by abbreviation called mnemonics that indicate the operation. Within the computer, each instruction is represented by a sequence of bits. The instruction is divided into fields, corresponding to the constituents elements of the instruction Opcodes are represented by abbreviation called mnemonics that indicate the operation. Common examples include:

ADD	add
SUB	SUBTRACT
MUL	multiply
DIV	divide
LOAD	Load data form memory
STOR	Store data to memory

Operands

Operands are also represented symbolically. For example the instruction ADD, R, Y May mean add the value contained in data location Y to the contents of register R. In this example Y refers to the address of a location in memory, and R refers to a particular register. Note that the operation is performed on the contents of a location not on its address:

Thus, it is possible to write a machine language program in symbolic form.

X= 413

Y= 414

Addressing modes

Q. What are the four main functions of a computer

1. Data processing
2. Data storage
3. Data movement
4. Control

STRUCTURE AND FUNCTION

A computer is a computer system, contemporary computers contain millions of elementary electronic components.

- Structure: The way in which the components are interrelated.
- Function: The operation of each individual component as part of the structure.

In term of description, there are two choices: starting at the bottom and building up to a complete description, or beginning with a top view and decomposing the system into its subparts. Evidence from a number of fields suggests that the top down approach is the clearest and most effective. The approach taken is that the computer be described from the top down.

Q. What do you understand by "Computer function"?

Computers have four functions. They are input (Receiving information), storage (store data), processing (manipulate data) and output (provides the user with information).

Q. Explain what is meant by Micro-operation?

To summarize, the execution of a program consists of the sequential execution of instructions. Each instruction is executed during an instruction cycle made up of shorter subcycles (e.g., fetch, indirect, execute, interrupt). The execution of each subcycle involves one or more shorter operations, that is, micro-operations. Micro-operations are the functional, or atomic, operations of a processor. Each micro-operation involves the movement of data into or out of a register.

Q. Briefly analyse the functional components of a computer.

ALU

An arithmetic logic unit (ALU) is a digital circuit used to perform arithmetic and logic operations. It represents the fundamental building block of the central processing unit (CPU) of a computer. Modern CPUs contain very powerful and complex ALUs. In addition to ALUs, modern CPUs contain a control unit (CU). The control unit controls the operation of the ALU by a set of control signals. These signals activate various logic circuits and gates within the ALU.

The arithmetic and logic unit (ALU) is that part of the computer that actually performs arithmetic and logical operations on data. All of the other elements of the computer system- Control unit, registers memory, I/O- are there mainly to bring into the ALU for it to process and then take the result back out. An ALU and all electronic components in the computers are based on the use of simple digital logic devices that can store binary digits and perform simple Boolean logic operations.

Registers

Number of processor registers that can be referenced by instructions and their use. Data are presented to the ALU in registers and the results of an operation are stored in registers. These registers are temporary storage locations within the processor that are connected by signal paths to the ALU.

Internal data paths External data paths

A data path (also written as datapath) is a set of functional units that carry out data processing operations. Datapaths, along with a control unit, make up the CPU (central processing unit) of a computer system. A larger data path can also be created by joining more than one together

using multiplexers. Internal data paths are used to move data between registers and between register and ALU. External data paths link registers to memory and I/O modules, often by means of a system bus. The control unit causes operations to happen within the processor.

Control unit

The control unit causes operations to happen within the processor. The execution of a program consists of operations involving these processor elements. As we have seen, these operations consist of a sequence of micro-operations. microoperations fall into one of the following categories:

- Transfer data from one register to another.
- Transfer data from a register to an external interface (e.g., system bus).
- Transfer data from an external interface to a register.
- Perform an arithmetic or logic operation, using registers for input and output.

All of the micro-operations needed to perform one instruction cycle, including all of the micro-operations to execute every instruction in the instruction set, fall into one of these categories.

We can now be somewhat more explicit about the way in which the control unit functions. The control unit performs two basic tasks:

- Sequencing: The control unit causes the processor to step through a series of micro-operations in the proper sequence, based on the program being executed.
- Execution: The control unit causes each micro-operation to be performed.

Q. Explain the concept of the Von Neumann computer.

Von-Neumann proposed his computer architecture design in 1945 which was later known as Von-Neumann Architecture. It consisted of a Control Unit, Arithmetic, and Logical Memory Unit (ALU), Registers and Inputs/Outputs. This design is still used in most computers produced today. An input device will bring instructions and data in sequentially. But a program is not invariably executed sequentially; it may jump around (e.g., the IAS jump instruction). Similarly, operations on data may require access to more than just one element at a time in a

predetermined sequence. Thus, there must be a place to store temporarily both instructions and data. That module is called memory, or main memory to distinguish it from external storage of peripheral devices. Von Neumann pointed out that the same memory could be used to store both instructions and data.

Q. State the sequence of operations of the control unit in one clock pulse.

The control unit functions as follows:

1. To execute an instruction, the sequencing logic unit issues a READ command to the control memory.
2. The word whose address is specified in the control address register is read into the control buffer register.
3. The content of the control buffer register generates control signals and next address information for the sequencing logic unit.
4. The sequencing logic unit loads a new address into the control address register based on the next-address information from the control buffer register and the ALU flags.

Q. Explain the instruction format

An instruction format defines the layout of the bits of an instruction, in terms of its constituent's fields. An instruction format must include an opcode and implicitly or explicitly, zero or more operands. The format must implicitly or explicitly, indicate the addressing mode for each operand. For most instruction sets, more than one instruction format is used.

Q. What is processor's instruction set?

The operation of the processor is determined by the instructions it executes referred to as machine instructions or computer instruction. The collection of different instructions that the processor can execute is referred to as the processor's instruction set.

Q. Discuss the elements of a machine instruction

ELEMENTS OF A MACHINE INSTRUCTION

These elements are as follows:

Operation code: Specifies the operation to be performed (e.g, ADD, I/O). The operation is specified by a binary code, known as the operation code or opcode.

Source operand reference: This operation may involve one or more source operands, that is operands that are inputs for the operation

Results operands reference: The operation may produce a result

Next instruction reference: This tells the processor where to fetch the next instruction after the execution of this instruction is complete. The address of the next instruction to be fetched could be either a real address or a virtual address, depending on the architecture.

Main or virtual memory: As with next instruction references, the main or virtual memory address must be supplied.

Processor register: With rare exception a processor contains one or more registers that may be referenced by machine instructions. If only one registers exists reference to it may be implicit. If more than one register exists, then each register is assigned a unique name or number, and the instruction must contain the number of the designed register

Immediate: The value of the operand is contained in a field in the instruction being executed.

I/O device: The instruction must specify the I/O module and device for the operation. If memory-mapped I/O is used, this is just another main or virtual memory address

Q. List and briefly explain five important instruction set design issues

The instruction set is the programmer's means of controlling the processor. Thus, programmer requirements must be considered in designing the instruction set. The most important of these fundamental design issues include the following:

1. Operation repertoire: How many and which operations to provide and how complex operations should be.
2. Data types: The various types of data upon which operations are perform.
3. Instruction format: Instruction length (in nits) number of assesses size of various fields and so on.

4. Registers: Number of processor registers that can be referenced by instructions and their use.
5. Addressing: The mode or modes by which the address of an operand is specified.

These issues are highly interrelated and must be considered together in designing an instruction set.

Q. What in general terms is the distinction between computer organization and computer architecture?

Computer organization refers to the operational units and their interconnections that realize the architectural specification. On the other hand, computer architecture refers to those attributes of a system visible to a programmer or those attributes that have a direct impact on the logical execution of a program.

Q. What is the arithmetic and logic unit (ALU)?

The arithmetic and logic unit (ALU) is that part of the computer that actually performs arithmetic and logical operations on data. All of the other elements of the computer system- Control unit, registers memory, I/O- are there mainly to bring into the ALU for it to process and then take the result back out. An ALU and all electronic components in the computers are based on the use of simple digital logic devices that can store binary digits and perform simple Boolean logic operations. Data are presented to the ALU in registers and the results of an operation are stored in registers. These registers are temporary storage locations within the processor that are connected by signal paths to the ALU. The ALU may also set flags as the result of an operation. For example, an overflow flag is set to 1 if the result of a computation exceeds the length of the register into which it is to be stored. The flag values are also stored in registers within the processor. The control unit provides signals that control the operation of the ALU and the movement of the data into and out of the ALU.

Q. Write short note on the four register involved in the Fetch Cycles.

THE FETCH CYCLE

We begin by looking at the fetch cycle, which occurs at the beginning of each instruction cycle and causes an instruction to be fetched from memory. Four registers are involved:

- Memory address register (MAR): Is connected to the address lines of the system bus. It specifies the address in memory for a read or write operation.
- Memory buffer register (MBR): Is connected to the data lines of the system bus. It contains the value to be stored in memory or the last value read from memory.
- Program counter (PC): Holds the address of the next instruction to be fetched.
- Instruction register (IR): Holds the last instruction fetched.

Q. What is symmetric multi processor?

The term SMP refers to a computer hardware architecture and also to the operating system behaviour that reflects that architecture. It can be defined as a stands alone computer system with the following characteristics. An SMP operating system manages processor and other computer resources so that the user perceives a single operating system controlling system resources. In fact such a configuration should appear as a single processor multi programming system. In both the SMP and uniprocessor cases, multiple jobs or processes may be active at one time and it is the responsibility of the operating system to schedule their execution and to allocate resources. A user may construct application that use multiple jobs or process may be active at one time, and it is the responsibility of the operating system to schedule their execution and to allocate resources. A user may construct application that use multiple processes or multiple threads within processes without regard to whether a simple processor or multiple processor will be available. Thus a multi processor operating system must provide all the functionality of a multi programming system plus additional features to accord ate multiple processor. The operating system of an SMP schedules processes or threads across all of the processors. An SMP organization has a number of potential advantages over a uniprocessor organization.

Q. Identify key characteristics of a symmetric multiprocessor (SMP) system.

1. Addressing: It must be possible to distinguish modules on the bus to determine the source and destination of data.

2. Arbitration: Any I/Q module can temporarily function as “master” A mechanism is provided to arbitrate competing requests for bus control, using some sort of priority scheme.

3. Time – sharing: When one module is controlling the bus other modules are locked out and must if necessary, suspend operation until bus access is achieved. These uniprocessor features are directly usable in an SMP organization. In this latter case, there are now multiple processors as well as multiple I/O processors all attempting to gain access to one or more memory modules via the bus. The bus organization has several attractive features:

- Simplicity: This is the simplest approach to multi processor organization. The physical feature interface and the addressing, arbitration and time sharing logic of each processor remain the same as in a single processor system.
- Flexibility: It is generally easy to expand the system by attracting more processors to the bus.
- Reliability: The bus is essentially a passive medium, and the failure of any attached device should not cause failure of the whole system.

Q. What are some of key operating system design issues for an SMP?

Among the key design issues:

1. Simultaneous concurrent processes: Operating system routines need to be reentrant to allow several processor to execute the case is code simultaneously. With multiple processors executing the same or different parts of the operating system, operating system tables and management structures must be managed properly to avoid dead lock or invalid operations.

2. Scheduling: Any processor may perform scheduling, so conflict must be avoided. The scheduler must assign ready processes to available processors.

3. Synchronization: With multiple active processes having potential accesses to shared address spaces or shared I/O resources, care must be taken to provide effective. Synchronization is a facility that enforces mutual exclusion and event ordering.

4. Memory management: Memory management on a multi processor must deal with all of the issues found on unit processor machines. In addition the operating system needs to exploit the available hardware parallelism, such as multi ported memories, to achieve the best performance. The paging mechanism on different processors shares a page replacement.

5. Reliability and fault tolerance: The operating system should provide graceful degradation influence of processor failure. The scheduler and other portions of the operating system must recognize the loss of a processor and restructure management tables accordingly.

Q. What are the chief characteristics of an SMP?

1. There are two or more similar processors of comparable capability.
2. These processors share the same main memory and I/O facilities and are interconnected by a bus or other internal connection scheme such that memory access time is approximately the same for each process.
3. All processors share access to I/O devices either through the same channels or through different channels that provide paths to the same device.
4. All process can perform the same function (hence the term symmetric).
5. The system is controlled by an integrated operating system that provide interaction between processors and their programs at the job task file and data element levels.

Q. What are some of the potential advantages of an SMP compared with a uniprocessor?

1. Availability: In a symmetric multiprocessor, because all processors can perform the same functions, the failure of a single processor does not halt the machine. Instead, the system can continue to function at reduced performance.
2. Incremental growth: A user can enhance the performance of a system by adding an additional processor.
3. Scaling: Vendors can offer a range of products with different price and performance characteristics based on the number of processors configured in the system.

Q. Explain multi threading

In computer architecture, multithreading is the ability of a central processing unit (CPU) (or a single core in a multi-core processor) to provide multiple threads of execution concurrently, supported by the operating system. Where multiprocessing systems include multiple complete processing units in one or more cores, multithreading aims to increase utilization of a single core by using thread-level parallelism, as well as instruction-level parallelism. As the two techniques are complementary, they are sometimes combined in systems with multiple multithreading CPUs and with CPUs with multiple multithreading cores.

The concept of thread used in discussing multi-thread processors may not be the same as the concept of software threads in a multi-programmed operating system. At minimum, a multithreaded processor must provide a separate program counter for each thread of execution to be executed concurrently. The designs differ in the amount and type of additional hardware used to support concurrent thread execution. In general, instruction fetching takes place on a thread basis. The processor treats each thread separately and may use a number of techniques for optimizing single-thread execution, including branch prediction, register renaming, and superscalar techniques. What is achieved is thread-level parallelism, which may provide for greatly improved performance when married to instruction-level parallelism.

Q. List and briefly explain four principal approaches to multi threading

1. Interleaved multithreading: This is also known as fine-grained multithreading. The processor deals with two or more thread contexts at a time, switching from one thread to another at each clock cycle. If a thread is blocked because of data dependencies or memory latencies, that thread is skipped and a ready thread is executed.
2. Blocked multithreading: This is also known as coarse-grained multithreading. The instructions of a thread are executed successively until an event occurs that may cause delay, such as a cache miss. This event induces a switch to another thread. This approach is effective on an in-order processor that would stall the pipeline for a delay event such as a cache miss.
3. Simultaneous multithreading (SMT): Instructions are simultaneously issued from multiple threads to the execution units of a superscalar

processor. This combines the wide superscalar instruction issue capability with the use of multiple thread contexts.

4. Chip multiprocessing: In this case, the entire processor is replicated on a single chip and each processor handles separate threads. The advantage of this approach is that the available logic area on a chip is used effectively without depending on ever increasing complexity in pipeline design. This is referred to as multicore. Chip multiprocessing also enables simultaneous execution of instructions from different threads.

- Single-threaded scalar: This is the simple pipeline found in traditional RISC and CISC machines, with no multithreading.
- Interleaved multithreaded scalar: This is the easiest multithreading approach to implement. By switching from one thread to another at each clock cycle. The pipeline stages can be kept fully occupied, or close to fully occupied. The hardware must be capable of switching from one thread context to another between cycles.

Q. Explain multi threading and chip multi processor

A multi core computer also known as a chip multi processor, combines two or more processor (called cores) on a single piece of silicon (called die). Another organizational design decision in a multi core will be superscalar or will implement simultaneous multi threading (SMT). In chip multiprocessing, the entire processor is replicated on a single chip and each processor handles separate threads. The advantage of this approach is that the available logic area on a chip is used effectively without depending on ever increasing complexity in pipeline design.

Q. What do you understand by the word "Process"?

A process is an instance of a program running in a computer. It is close in meaning to task, a term used in some operating systems. In UNIX and some other operating systems, a process is started when a program is initiated (either by a user entering a shell command or by another program). Like a task, a process is a running program with which a particular set of data is associated so that the process can be kept track of. An application that is being shared by multiple users will generally have one process at some stage of execution for each user.

Q. What do you understand by "Process switch"?

Process switch is an operation that switches the processor from one process to another, by saving all the process control data, registers, and other information for the first and replacing them with the process information for the second.

Q. Briefly discuss the inputs and outputs of the general model of the control unit.

Clock: This is how the control unit "keeps time." The control unit causes one micro-operation (or a set of simultaneous micro-operations) to be performed for each clock pulse. This is sometimes referred to as the processor cycle time, or the clock cycle time.

Instruction register: The opcode and addressing mode of the current instruction are used to determine which micro-operations to perform during the execute cycle.

Flags: These are needed by the control unit to determine the status of the processor and the outcome of previous ALU operations. For example, for the increment-and-skip-if-zero (ISZ) instruction, the control unit will increment the PC if the zero flag is set.

Control signals from control bus: The control bus portion of the system bus provides signals to the control unit.

The outputs are as follows:

Control signals within the processor: These are two types: those that cause data to be moved from one register to another, and those that activate specific ALU functions.

Control signals to control bus: These are also of two types: control signals to memory, and control signals to the I/O modules.

Three types of control signals are used:

1. Those that activate an ALU function
2. Those that activate a data path
3. Those that are signals on the external system bus or other external interface.

Q. What is a "Thread"?

A dispatchable unit of work within a process. It includes a processor context (which includes the program counter and stack pointer) and its own data area for a stack (to enable subroutine branching). A thread executes sequentially and is interruptible so that the processor can turn to another thread. The concept of thread used in discussing multi-thread processors may not be the same as the concept of software threads in a multi-programmed operating system.

Q. Write briefly on the following

Resource ownership

A process includes a virtual address space to hold the process image; the process image is the collection of program, data, and attributes that define the process. From time to time, a process may allocate control or ownership of resources, such as main memory, I/O channels, I/O devices, and files.

Scheduling/execution

The execution of a process follows an execution (trace) through one or more programs. This execution may be interleaved with that of other processes. Thus, a process has an execution state (Running, Ready, etc.) and a dispatching priority and is the entity that is scheduled and dispatched by the operating system.

Process switch

An operation that switches the processor from one process to another, by saving all the process control data, registers, and other information for the first and replacing them with the process information for the second.

Q. Briefly describe the taxonomy of parallel processing systems.

A taxonomy first introduced by Flynn is still the most common way of categorizing systems with parallel processing capability. He proposed the following categories of computer systems:

Single instruction, single data (SISD) stream: A single processor executes a single instruction stream to operate on data stored in a single memory. Uniprocessors fall into this category.

Single instruction, multiple data (SIMD) stream: A single machine instruction controls the simultaneous execution of a number of processing elements on a lock-step basis. Each processing element has

associated data memory so that each instruction is executed on a different set of data by the different processors. Vector and array processors fall into this category.

Multiple instruction, single data (MISD) stream: A sequence of data is transmitted to a set of processors, each of which executes a different instruction sequence. This structure is not commercially implemented.

Multiple instructions, multiple data (MIMD) stream: A sequence of data transmitted to a set of processors, each of which executes a different instruction sequence. This structure is none commercially implemented.

Multiple instructions, multiple data (MIMD) stream: A set of processors simultaneously execute different instruction sequences on different data sets. SMPs, clusters and NUMA systems fit into this category.

Q. What is process scheduling?

Scheduling is defined as a program in execution. It is the key to multi programming. The process scheduling is the activity of the process manager that handles the removal of the running process from the CPU and the selection of another process on the basis of a particular strategy. Process scheduling is an essential part of Multiprogramming operating systems. Operating system performs a number of functions namely; process scheduling and memory management can only be performs efficiently and rapidly if the processor hardware includes capabilities to support the operating system. In computing, scheduling is the method by which work is assigned to resources that complete the work. The work may be virtual computation elements such as threads, processes or data flows, which are in turn scheduled onto hardware resources such as processors, network links or expansion cards.

Q. List and briefly describe the features provided to facilitate DMA transfers from I/O processors in a multiprocessor system.

To facilitate DMA transfers from I/O processors, the following features are provided:

1. Addressing: It must be possible to distinguish modules on the bus to determine the source and destination of data.

2. Arbitration: Any I/Q module can temporarily function as “master” A mechanism is provided to arbitrate competing requests for bus control, using some sort of priority scheme.

3. Time – sharing: When one module is controlling the bus other modules are locked out and must if necessary, suspend operation until bus access is achieved. These uniprocessor features are directly usable in an SMP organization. In this latter case, there are now multiple processors as well as multiple I/O processors all attempting to gain access to one or more memory modules via the bus.

4. The bus organization has several attractive features:

5. Simplicity: This is the simplest approach to multi processor organization. The physical feature interface and the addressing, arbitration and time sharing logic of each processor remain the same as in a single processor system.

6. Flexibility: It is generally easy to expand the system by attracting more processors to the bus.

7. Reliability: The bus is essentially a passive medium, and the failure of any attached device should not cause failure of the whole system.

Q. List the characteristics of a RISC machine.

1. One instruction per cycle
2. Register-to-register operations
3. Simple addressing modes
4. Simple instruction formats

Q. What is a delayed branch?

To compensate for these dependencies, code reorganization techniques have been developed. First, let us consider branching instructions. ,belayed branch, a way of increasing the efficiency of the pipeline, makes use of a branch that does not take effect until after execution of the following instruction (hence the term delayed). The instruction location immediately following the branch is referred to the delay slot. A RISC instruction set architecture also lends itself to the delayed branch technique, in which branch instructions are rearranged with other instruction to improve pipeline efficiency.

Q. List the key characteristics of a symmetric multiprocessor (SMP) system.

1. There are two or more similar processors of comparable capability.
2. These processors share the same main memory and I/O facilities and are interconnected by a bus or other internal connection scheme such that memory access time is approximately the same for each process.
3. All processors share access to I/O devices either through the same channels or through different channels that provide paths to the same device.
4. All process can perform the same function (hence the term symmetric).
5. The system is controlled by an integrated operating system that provide interaction between processors and their programs at the job task file and data element levels.

Q. State the common types of scheduling.

1. **The Long Term Scheduling**

The long-term scheduler determines which programs are admitted to the system for processing. Thus, it controls the degree of multiprogramming (number of processes in memory). Once admitted, a job or user program becomes a process and is added to the queue for the short-term scheduler. In some systems, a newly created process begins in a swapped out condition, in which case it is added to a queue for the medium-term scheduler.

In a batch system, or for the batch portion of a general-purpose OS, newly submitted jobs are routed to disk and held in a batch queue. The long-term scheduler creates processes from the queue when it can. There are two decisions involved here. First, the scheduler must decide that the OS can take on one or more additional processes. Second, the scheduler must decide which job or jobs to accept and turn into processes. The criteria used may include priority, expected execution time, and I/O requirements.

For interactive programs in a time-sharing system, a process request is generated when a user attempts to connect to the system. Time-sharing users are not simply queued up and kept waiting until the system can accept them. Rather, the OS will accept all authorized comers until the

system is saturated, using some predefined measure of saturation. At that point, a connection request is met with a message indicating that the system is full and the user should try again later.

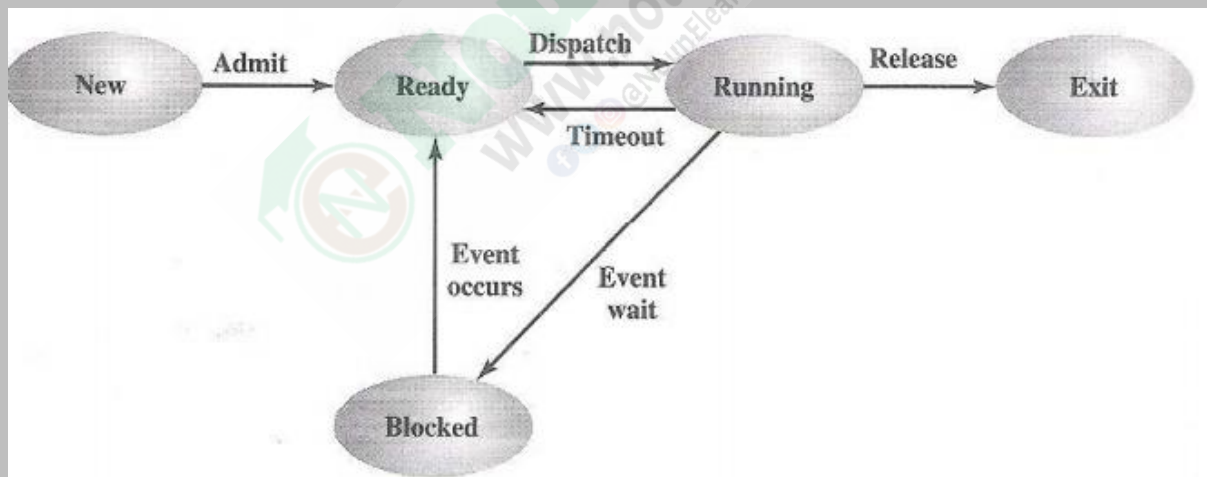
2. Medium-Term Scheduling

Medium-term scheduling is part of the swapping function, described in Section 8.3. Typically, the swapping-in decision is based on the need to manage the degree of multiprogramming. On a system that does not use virtual memory, memory management is also an issue. Thus, the swapping-in decision will consider the memory requirements of the swapped-out processes.

3. Short-Term Scheduling

The long-term scheduler executes relatively infrequently and makes the coarse-grained decision of whether or not to take on a new process, and which one to take. The short-term scheduler, also known as the dispatcher, executes frequently and makes the fine-grained decision of which job to execute next.

Q. With the aid of an illustrative diagram, explain the five-state process model.



To understand the operation of the short-term scheduler, we need to consider the concept of a process state. During the lifetime of a process, its status will change a number of times. Its status at any point in time is referred to as a state. The term state is used because it connotes that certain information exists that defines the status at that point. At minimum, there are five defined states for a process (Figure 3.1.1)

Q. Explain what is meant by a hardwired control unit.

In a hardwired implementation, the control unit is essentially a state machine circuit. Its input logic signals are transformed into a set of output logic signals, which are the control signals. An alternative to a hardwired control unit is a micro programmed control unit in which the logic of the control unit is specified by a micro program. A micro program consist of a sequence of instructions in a microprogramming language. These are very simple instruction that specify micro operations.

Q. Briefly describe the following:

i. Dual core processor chip.

A multi-core processor is a computer processor integrated circuit with two or more separate processing units, called cores, each of which reads and executes program instructions, as if the computer had several processors. Each processor chip includes two identical central processor, in which most of the instructions are hard wired and the rest are executed by vertical micro code. Each CP includes 9256- KBL 1 instruction cache and a 256- KB data cache.

ii. L2 cache

Each L2 caches are arranged in clusters of five, with each cluster supporting eight processor chips and providing access to the entire main memory space.

iii. System control element (SCE).

The SCE arbitrates system communication and has a central role in maintaining cache coherence.

iv. Main store control (MSC).

The MSCs interconnect the L2 caches and the main memory.

v. Memory card.

Each card holds 32 GB of memory. The maximum configurable memory consists of 8 memory cards for a total of 256 GB. Memory cards interconnect to the MSC via synchronous memory interfaces (SMIs)

vi. Memory has adapter (MBA)

The MBA provides an interface to various types of I/O channels. Traffic to/from the channels goes directly to the L2 cache.

Q. Explain what is meant by Micro-operation.

Micro-operations are the functional, or atomic, operations of a processor. In computer central processing units, micro-operations (also known as a micro-ops or μ ops) are detailed low-level instructions used in some designs to implement complex machine instructions (sometimes termed macro-instructions in this context)

Q. List the functional elements of the processor?

The basic functional elements of the processor are the following:

1. ALU
2. Registers
3. Internal data paths External data paths
4. Control unit

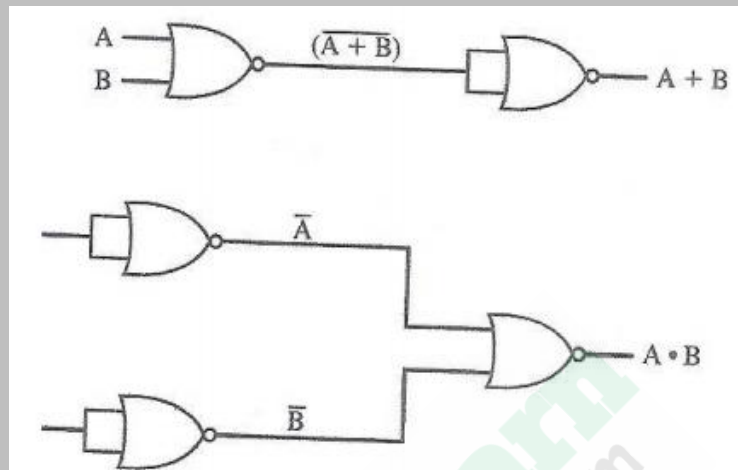
Q. What is a gate?

A gate is an electronic circuit that produces an output signal that is a simple Boolean operation on its input signals. The basic gates used in digital logic are AND, OR, NOT, NAND, NOR, and XOR. Figure 20.1 depicts these six gates. Each gate is defined in three ways: graphic symbol, algebraic notation, and truth table. The symbology used here and throughout the appendix is the IEEE standard, IEEE Std 91. Note that the inversion (NOT) operation is indicated by a circle. Gates and control signals are provided for movement of data onto and off the bus from each register. Additional control signals control data transfer to and from the system (external) bus and the operation of the ALU.

Q. Explain the basic logic gates

Typically, not all gate types are used in implementation. Design and fabrication are simpler if only one or two types of gates are used. Thus, it is important to identify functionally complete sets of gates. This means that any Boolean function can be implemented using only the gates in the set. The following are functionally complete sets:

- AND, OR, NOT
- AND, NOT
- OR, NOT
- NAND
- NOR



Briefly explain the following concepts:

i) Delayed load

A similar sort of tactic, called the delayed load, can be used on LOAD instructions. On LOAD instructions, the register that is to be the target of the load is locked by the processor. The processor then continues execution of the instruction stream until it reaches an instruction requiring that register, at which point it idles until the load is complete. If the compiler can rearrange instructions so that useful work can be done while the load is in the pipeline, efficiency is increased.

ii) Loop unrolling

Another compiler technique to improve instruction parallelism is loop unrolling [BAC094]. Unrolling replicates the body of a loop some number of times called the unrolling factor (u) and iterates by step a instead of step 1. Unrolling can improve the performance by

- reducing loop overhead
- increasing instruction parallelism by improving pipeline performance
- improving register, data cache, or TLB locality

iii) Delayed Branch

To compensate for these dependencies, code reorganization techniques have been developed. First, let us consider branching instructions. ,belayed branch, a way of increasing the efficiency of the pipeline, makes use of a branch that does not take effect until after execution of the following instruction (hence the term delayed). The instruction location immediately following the branch is referred to the delay slot.

Q. State and explain the three performance parameters of the memory.

1. Access time (latency)

For random-access memory, this is the time it takes to perform a read or write operation, that is, the time from the instant that an address is presented to the memory to the instant that data have been stored or made available for use. For non-random-access memory, access time is the time it takes to position the read-write mechanism at the desired location.

2. Memory cycle time

This concept is primarily applied to random-access memory and consists of the access time plus any additional time required before a second access can commence. This additional time may be required for transients to die out on signal lines or to regenerate data if they are read destructively. Note that memory cycle time is concerned with the system bus, not the processor.

3. Transfer rate

This is the rate at which data can be transferred into or out of a memory unit. For random-access memory, it is equal to $1/(\text{cycle time})$.

Q. Briefly explain the following terms:

i) Process switch

An operation that switches the processor from one process to another, by saving all the process control data, registers, and other information for the first and replacing them with the process information for the second.

ii) Delayed branch

To compensate for these dependencies, code reorganization techniques have been developed. First, let us consider branching instructions.

,delayed branch, a way of increasing the efficiency of the pipeline, makes use of a branch that does not take effect until after execution of the following instruction (hence the term delayed). The instruction location immediately following the branch is referred to the delay slot.

iii) Thread switch

The act of switching processor control from one thread to another within the same process. Typically, this type of switch is much less costly than a process switch.

Q. Enumerate five ways by which Vector facility is integrated into the System J370 Architecture.

The vector facility is integrated into the System J370 architecture in the following ways:

1. Existing System/370 instructions are used for all scalar operations.
2. Arithmetic operations on individual vector elements produce exactly the same result as do corresponding System/370 scalar instructions.
3. Vector instructions are interruptible, and their execution can be resumed for the point of interruption after appropriate action has been taken, in a manner compatible with the System/370 program-interruption scheme.
4. Vector data reside in virtual storage, with page faults being handled in a standard manner.
5. Arithmetic exceptions are the same as, or extensions of, exceptions for the scalar arithmetic instructions of the System/370, and similar fix-up routines can be used.

Q. State the three benefits provided by the level of integration of vector facility of the J370 architecture.

1. Existing operating systems can support the vector facility with minor extensions.
2. Existing application programs, language compilers, and other software can be run unchanged.
3. Software that could take advantage of the vector facility can be modified as desired.

Q. State any two of the benefits provided by the level of integration of vector facility of the J370 architecture.

1. Existing operating systems can support the vector facility with minor extensions.
2. Existing application programs, language compilers, and other software can be run unchanged.

Q. Enumerate the two ways by which a disk cache can improve performance

Disk cache improves performance in two ways:

1. Disk writes are clustered. Instead of many small transfers of data, we have a few large transfers of data. This improves disk performance and minimize processor involvement.
2. Some data destined for write-out may be referenced by a program before the next dump to disk. In that case, the data are retrieved rapidly from the software cache rather than slowly from the disk.

Q. Define the following:

i. processors instruction

The operation of the processor is determined by the instructions it executes referred to as machine instructions or computer instruction. The collection of different instructions that the processor can execute is referred to as the processors instruction set.

ii. Instruction

An instruction format defines the layout of the bits of an instruction, in terms of its constituent's fields. An instruction format must include an opcode and implicitly or explicitly, zero or more operands. The format must implicitly explicitly, indicate the addressing mode for each operands. For most instruction sets, more than one instruction format is used.

iii. Data types

The various types of data upon which operations are perform.

iv. Registers

Number of processor registers that can be referenced by instructions and their use.

v. Addressing

The mode or modes by which the address of an operand is specified

Q. Define the following acronym:

i. MAR.

Memory address register (MAR): Is connected to the address lines of the system bus. It specifies the address in memory for a read or write operation.

ii. MBR.

Memory buffer register (MBR): Is connected to the data lines of the system. It contains the value to be stored in memory or the last value read from memory.

iii. IR.

Instruction register (IR): Holds the last instruction fetched.

iv. PC.

Program counter (PC): Holds the address of the next instruction to be fetched.

v. Interrupt.

The indirect cycle is always followed by the execute cycle. The interrupt cycle is always followed by the fetch cycle. For both the fetch and execute cycles, the next cycle depends on the state of the system.

vi. Timer.

A timer is used to prevent a single job from monopolizing the system. The timer is set at the beginning of each job. If the timer expires an interrupt occurs and control returns to the monitor.

Q. Explain the following:

i. Clock

This is how the control unit "keeps time." The control unit causes one micro-operation (or a set of simultaneous micro-operations) to be performed for each clock pulse. This is sometimes referred to as the processor cycle time, or the clock cycle time.

ii. Flags

These are needed by the control unit to determine the status of the processor and the outcome of previous ALU operations. For example, for the increment-and skip-if-zero (ISZ) instruction, the control unit will increment the PC if the zero flag is set.

iii. Instruction register

The opcode and addressing mode of the current instruction are used to determine which micro-operations to perform during the execute cycle.

Q. What is Interrupt control?

An interrupt control register, or ICR, is a hardware register in a computer chip used to configure the chip to generate interrupts—to raise a signal on an interrupt line—in response to some event occurring within the chip or a circuit connected to the chip. This module handles multiple levels of interrupt signals.

Q. State the four Characteristics of Reduced Instruction Set Architectures.

Characteristics of Reduced Instruction Set Architectures

1. One instruction per cycle

The first characteristic listed is that there is one machine instruction per machine cycle. A machine cycle is defined to be the time it takes to fetch two operands from registers, perform an ALU operation, and store the result in a register. Thus, RISC machine instructions should be no more complicated than, and execute about as fast as, microinstructions on CISC machines (discussed in Part Four). With simple, one-cycle instructions, there is little or no need for microcode; the machine instructions can be hardwired. Such instructions should execute faster than comparable machine instructions on other machines, because it is not necessary to access a microprogram control store during instruction execution.

2. Register-to-register operations

A second characteristic is that most operations should be register to register. with only simple LOAD and STORE operations accessing

memory. This design feature simplifies the instruction set and therefore the control unit. For example, a RISC instruction set may include only one or two ADD instructions (e.g., integer add, add with carry); the VAX has 25 different ADD instructions. Another benefit is that such an architecture encourages the optimization of register use, so that frequently accessed operands remain in high-speed storage.

3. Simple addressing modes

A third characteristic is the use of simple addressing modes. Almost all RISC instructions use simple register addressing. Several additional modes, such as displacement and PC-relative, may be included. Other, more complex modes can be synthesized in software from the simple ones. Again, this design feature simplifies the instruction set and the control unit.

4. Simple instruction formats

A final common characteristic is the use of simple instruction formats. Generally, only one or a few formats are used. Instruction length is fixed and aligned on word boundaries. Field locations, especially the opcode, are fixed. This design feature has a number of benefits. With fixed fields, opcode decoding and register operand accessing can occur simultaneously. Simplified formats simplify the control unit. Instruction fetching is optimized because word-length units are fetched. Alignment on a word boundary also means that a single instruction does not cross page boundaries.

Q. What is Instruction address calculation (ac)?

Instruction address calculation (ac) determines 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.

Q. Describe instruction representation in the computer.

Within the computer, each instruction is represented by a sequence of bits. The instruction is divided into fields, corresponding to the constituent elements of the instruction. Opcodes are represented by abbreviation called mnemonics that indicate the operation. Common examples include:

ADD	add
SUB	SUBTRACT
MUL	multiply
DIV	divide
LOAD	Load data form memory
STOR	Store data to memory

Q. Briefly describe what you understand by instruction format

An instruction format defines the layout of the bits of an instruction, in terms of its constituent's fields. An instruction format must include an opcode and implicitly or explicitly, zero or more operands. The format must implicitly explicitly, indicate the addressing mode for each operands. For most instruction sets, more than one instruction format is used. The instruction format provide 4 bits for the opcode, so that there can be as many as $2^4 = 16$ different opcodes, aup to $2^{12} = 4096$ (4K) words of memory can be directly addressed.

Q. Source and result operands can be in one of four areas. Enumerate these areas and state when and how each are is in use.

1. Main or virtual memory: As with next instruction references, the main or virtual memory address must be supplied.

2. Processor register: With rare exception a processor contains one or more registers that may be referenced by machine instructions. If only one registers exits reference to it may be implicit. If more than one register exists, then each register is assigned a unique name or number, and the instruction must contain the number of the designed register

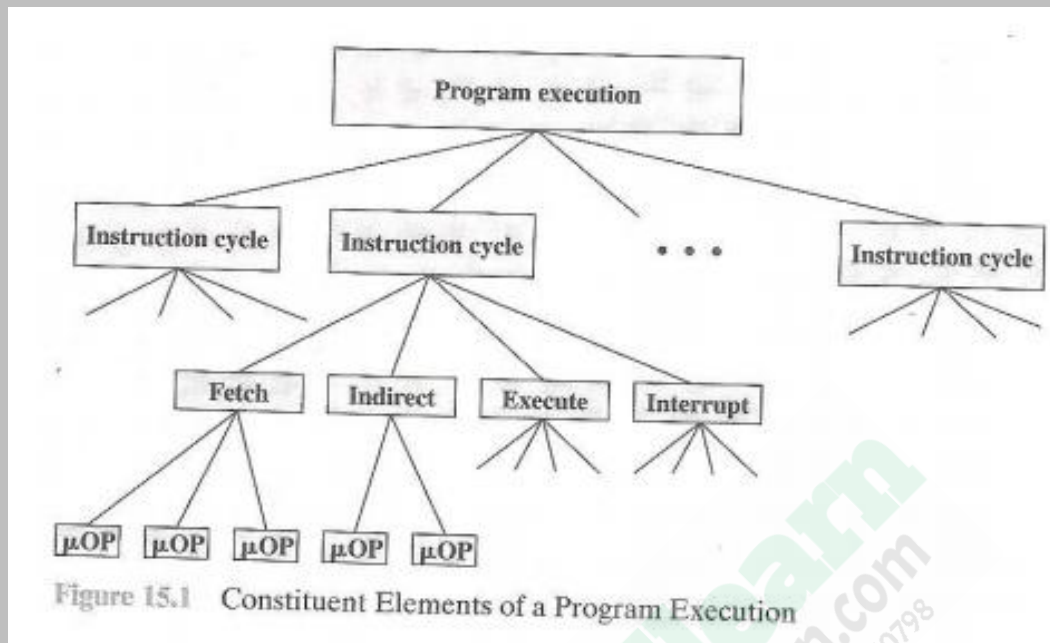
3. Immediate: The value of the operand is contained in a field in the instruction being executed.

4. I/O device: The instruction must specify the I/O module and device for the operation. If memory-mapped I/O is used, this is just another main or virtual memory address

Q. Write short note on the simultaneous execution of the action: Read word from memory and Increment PC.

To summarize, the execution of a program consists of the sequential execution of instructions. Each instruction is executed during an

instruction cycle made up of shorter subcycles (e.g., fetch, indirect, execute, interrupt). The execution of each subcycle involves one or more shorter operations, that is, micro-operations.



Micro-operations are the functional, or atomic, operations of a processor bus, the control unit issues a READ command on the control bus, and the result appears on the data bus and is copied into the memory buffer register (MBR). We also need to increment the PC by the instruction length to get ready for the next instruction. Because these two actions (read word from memory, increment PC) do not interfere with each other, we can do them simultaneously to save time. The third step is to move the contents of the MBR to the instruction register (IR). This frees up the MBR for use during a possible indirect cycle.

Thus, the simple fetch cycle actually consists of three steps and four microoperations. Each micro-operation involves the movement of data into or out of a register. So long as these movements do not interfere with one another, several of them can take place during one step, saving time.

Q. Draw a logic gate to generate the result for the following:

i. $\text{Sum} = A BC + ABC + ABC + ABC$

ii. $\text{Carry} = AB + AC + BCY$

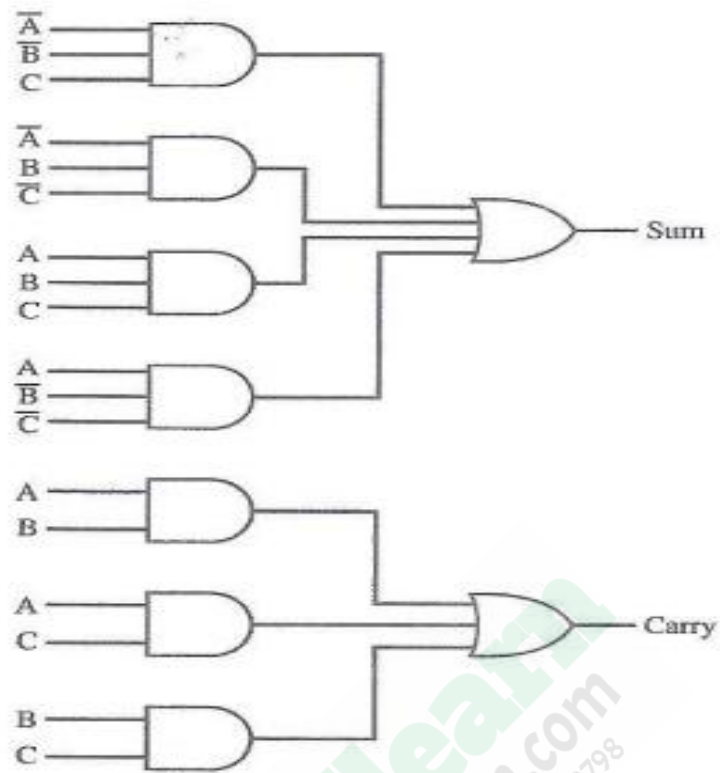


Figure 20.20 Implementation of an Adder