MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

Estd. 2000

L - 01

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 16ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit I : I – Introduction to Computer Architecture & Organization

Date of Lecture:

Topic of Lecture: Computing and computers

Introduction:

A **computer** is an electronic device that receives input, stores or processes the input as per user instructions and provides output in desired format

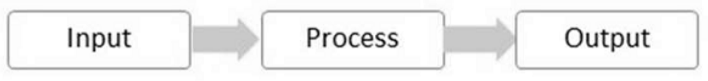
Prerequisite knowledge for Complete understanding and learning of Topic:

Fundamentals of computing, Microprocessor and microcontrollers

Computer is an electronic device that receives input, stores or processes the input as per user instructions and provides output in desired format.

Input-Process-Output Model

Computer input is called **data** and the output obtained after processing it, based on user's instructions is called **information**. Raw facts and figures which can be processed using arithmetic and logical operations to obtain information are called **data**.

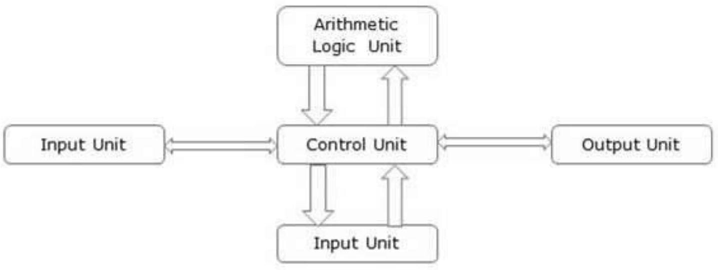


Simplest Workflow of Computer

The processes that can be applied to data are of two types −

- **Arithmetic operations** − Examples include calculations like addition, subtraction, differentials, square root, etc.
- **Logical operations** − Examples include comparison operations like greater than, less than, equal to, opposite, etc.

The corresponding figure for an actual computer looks something like this −



The basic parts of a computer are as follows −

- **Input Unit** − Devices like keyboard and mouse that are used to input data and instructions to the computer are called input unit.

- **Output Unit** − Devices like printer and visual display unit that are used to provide information to the user in desired format are called output unit.
- **Control Unit** − As the name suggests, this unit controls all the functions of the computer. All devices or parts of computer interact through the control unit.
- **Arithmetic Logic Unit** − This is the brain of the computer where all arithmetic operations and logical operations take place.
- **Memory** − All input data, instructions and data interim to the processes are stored in the memory. Memory is of two types − **primary memory** and **secondary memory**. Primary memory resides within the CPU whereas secondary memory is external to it.

Control unit, arithmetic logic unit and memory are together called the **central processing unit** or **CPU**. Computer devices like keyboard, mouse, printer, etc. that we can see and touch are the **hardware** components of a computer. The set of instructions or programs that make the computer function using these hardware parts are called **software**. We cannot see or touch software. Both hardware and software are necessary for working of a computer.

Advantages of Using Computer

Now that we know the characteristics of computers, we can see the advantages that computers offer−

- Computers can do the same task repetitively with same accuracy.
- Computers do not get tired or bored.
- Computers can take up routine tasks while releasing human resource for more intelligent functions.

Disadvantages of Using Computer

Despite so many advantages, computers have some disadvantages of their own −

- Computers have no intelligence; they follow the instructions blindly without considering the outcome.
- Regular electric supply is necessary to make computers work, which could prove difficult everywhere especially in developing nations.

Booting

Starting a computer or a computer-embedded device is called **booting**. Booting takes place in two steps −

- Switching on power supply
- Loading operating system into computer's main memory
- Keeping all applications in a state of readiness in case needed by the user

The first program or set of instructions that run when the computer is switched on is called **BIOS** or **Basic Input Output System**. BIOS is a **firmware**, i.e. a piece of software permanently programmed into the hardware.

If a system is already running but needs to be restarted, it is called **rebooting**. Rebooting may be required if a software or hardware has been installed or system is unusually slow.

There are two types of booting −

- **Cold Booting** − When the system is started by switching on the power supply it is called cold booting. The next step in cold booting is loading of BIOS.
- **Warm Booting** − When the system is already running and needs to be restarted or rebooted, it is called warm booting. Warm booting is faster than cold booting because BIOS is not reloaded.

Video Content / Details of website for further learning (if any):

https://www.youtube.com/watch?v=7cXEOWAStq4

Important Books/Journals for further learning including the page nos.:

John P. Hayes, “Computer Architecture and Organization”, Tata McGraw-Hill, 3rd edition, 1998, page no. :1 to 12

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

DESIGNING YOUR FUTURE
Estd. 2000

IQAC

L - 02

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit I : I - Introduction to Computer Architecture & Organization

Date of Lecture:

Topic of Lecture: Evolution of computers, VLSI Era

Introduction:

Computers work through an interaction of hardware and software. The whole picture of the computer goes back to decades. However, there are five apparent generations of computers. Each generation is defined by a paramount technological development that changes necessarily how computers operate – leading to more compressed, inexpensive, but more dynamic, efficient and booming machines.

Prerequisite knowledge for Complete understanding and learning of Topic:

Microprocessor and microcontrollers

Evolution of computers:

First Generation – Vacuum Tubes (1940 – 1956)

These ancient computers utilized vacuum tubes as circuitry and magnetic drums for recollection. As a result, they were huge, actually taking up entire rooms and costing resources to run.

Second Generation – Transistors (1956 – 1963)

The supersession of vacuum tubes by transistors, visualized the onset of the second generation of computing. The language emerged from strange binary language to symbolic ('assembly') languages. This meant programmers could discover instructions in words. Meanwhile during the same time high caliber programming languages were being developed (early versions of COBOL and FORTRAN).

Third Generation – Integrated Circuits (1964 – 1971)

By this phase, transistors were now being miniaturised and put on silicon chips. This led to a huge improvement in speed and effectiveness of these machines. These were the first computers where users interacted utilizing keyboards and monitors which interfaced with an operating system, a consequential leap up from the punch cards and printouts.

Fourth Generation – Microprocessors (1972 – 2010)

This innovation can be defined in one word: Intel. The chip-maker accomplished the Intel 4004 chip in 1971, which located all components of computer such as CPU, recollection, input/output controls onto a single chip. What overcrowded a room in the 1940s now gets fit in the palm of the hand.The Intel chip contained thousands of unified circuits.

Fifth Generation – Artificial Intelligence (2010 Onwards)

Computer devices with artificial potentiality are still in development, but some of these technologies are commencing to emerge and be used such as voice recognition. AI is an authenticity, made possible by adopting parallel processing and superconductors. Inclining to the future, computers will be thoroughly revolutionized again by quantum computation, molecular and nano technology.

VLSI Era:

Moore's prediction, which is more commonly known as Moore's law nowadays, has been widely used in the semiconductor and microelectronic industries as a tool to predict the increase of components in a chip for the coming generations [4]. To date, Moore's law has been proven to have held valid for more than half a century. Table 1 depicts the progressive trend of the integration level for the semiconductor industry. It can be observed from the table that the number of transistors that can be fabricated in a chip has been growing continuously over the years. In fact, this growth has complied closely with Moore's law. To distinguish the increase of transistors in every 10 years, each era is designated a name, that is, the SSI, MSI, LSI, VLSI, ULSI and SLSI eras. During the VLSI era, a microprocessor was fabricated for the first time into a single integrated circuit chip. Although this era has now long passed, the VLSI term is still being widely used today. This is partly due to the absence of an obvious qualitative leap between VLSI and its subsequent ULSI and SLSI eras, and partly, it is also because IC engineers and experts working in this field have been so used to this term that they decided to continue adopting it.

Integration level	**Year**	**Number of transistors in a chip**
Small-scale integration (SSI)	1950	Less than 100
Medium-scale integration (MSI)	1960	Between 100 and 1000
Large-scale integration (LSI)	1970	Between 1000 and 10,000
Very large-scale integration (VLSI)	1980	Between 10,000 and 100,000
Ultra large-scale Integration (ULSI)	1990	Between 100,000 and 10,000,000
Super large-scale integration (SLSI)	2000	More than 10,000,000

Video Content / Details of website for further learning (if any):
https://www.youtube.com/watch?v=UFbuRFY6Ps8

Important Books/Journals for further learning including the page nos.:
John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no. 12 to 56

Course Faculty

Verified by HOD

L - 03

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit I : Introduction to Computer Architecture & Organization

Date of Lecture:

Topic of Lecture: System Design

Introduction:

The basic function performed by a computer is execution of a program, which consists of a set of instructions stored in memory. Instruction processing consists of two steps: The processor reads (fetches) instructions from memory one at a time and executes each instruction. Program execution consists of repeating the process of instruction fetch and instruction execution.

Prerequisite knowledge for Complete understanding and learning of Topic:

Microprocessor and microcontrollers

System Design:

Virtually all contemporary computer designs are based on concepts developed by John von Neumann at the Institute for Advanced Studies, Princeton. Such a design is referred to as the von Neumann architecture and is based on three key concepts:

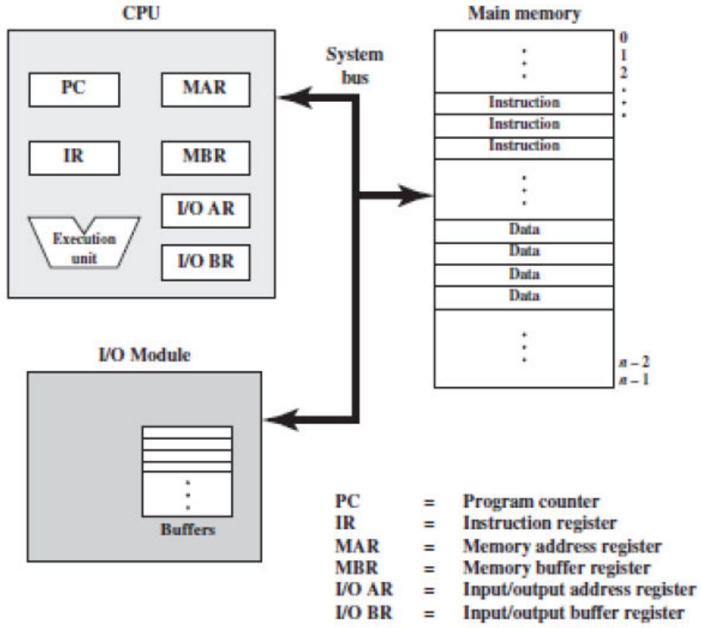
- Data and instructions are stored in a single read–write memory

The contents of this memory are addressable by location, without regard to the type of data contained there.

- Execution occurs in a sequential fashion (unless explicitly modified) from one instruction to the next.

If there is a particular computation to be performed, a configuration of logic components designed specifically for that computation could be constructed. The resulting “program” is in the form of hardware and is termed a hardwired program.

The CPU exchanges data with memory. For this purpose, it typically makes use of two internal (to the CPU) registers: a memory address register (MAR), which specifies the address in memory for the next read or write, and a memory buffer register (MBR), which contains the data to be written
into memory or receives the data read from memory. Similarly, an I/O address register (I/OAR) specifies a particular I/O device. An I/O buffer (I/OBR) register is used for the exchange of data between an I/O module and the CPU.



Video Content / Details of website for further learning (if any):
https://www.youtube.com/watch?v=quLrc3PbuIw&list=PLMCXHnjXnTnvo6alSjVkgxV-VH6EPyvoX

Important Books/Journals for further learning including the page nos.:
John P. Hayes, “Computer Architecture and Organization”, Tata McGraw-Hill, 3rd edition, 1998, page no. :64

Course Faculty

Verified by HOD

L - 04

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit I : Introduction to Computer Architecture & Organization

Date of Lecture:

Topic of Lecture: Register level, Processor level

Introduction:
A digital system can be treated at different level of abstraction or complexity. At a higher level than the gate level, we have the register level. The central component at this level is an n-bit storage device, or register. The processor is the element of a system that involves the actual transformation of input into output. It is the operational component of a system. Processors may modify the input either totally or partially, depending on the output specification.

Prerequisite knowledge for Complete understanding and learning of Topic:
Microprocessor and microcontrollers

Register level
A computer system employs a memory hierarchy. At higher levels of the hierarchy, memory is faster, smaller, and more expensive (per bit). Within the processor, there is a set of registers that function
as a level of memory above main memory and cache in the hierarchy. The registers in the processor perform two roles:

- **User-visible registers:** Enable the machine- or assembly language programmer to minimize main memory references by optimizing use of registers.
- **Control and status registers:** Used by the control unit to control the operation of the processor and by privileged, operating system programs to control the execution of programs.

User-Visible Registers
A user-visible register is one that may be referenced by means of the machine language that the processor
executes. We can characterize these in the following categories:

- General purpose
- Data
- Address
- Condition codes

General-purpose registers can be assigned to a variety of functions by the programmer. Sometimes their use within the instruction set is orthogonal to the operation. That is, any general-purpose
register can contain the operand for any opcode. This provides true general-purpose register use. There may be dedicated registers for floating-point and stack operations.
In some cases, general-purpose registers can be used for addressing functions (e.g., register indirect, displacement).
Data registers may be used only to hold data and cannot be employed in the calculation of an operand address.

Address registers may themselves be somewhat general purpose, or they may be devoted to a particular addressing mode. Examples include the following:

- **Segment pointers:** In a machine with segmented addressing, a segment register holds the address of the base of the segment.
- **Index registers:** These are used for indexed addressing and may be auto indexed.
- **Stack pointer:** If there is user-visible stack addressing, then typically there is a dedicated register that points to the top of the stack.

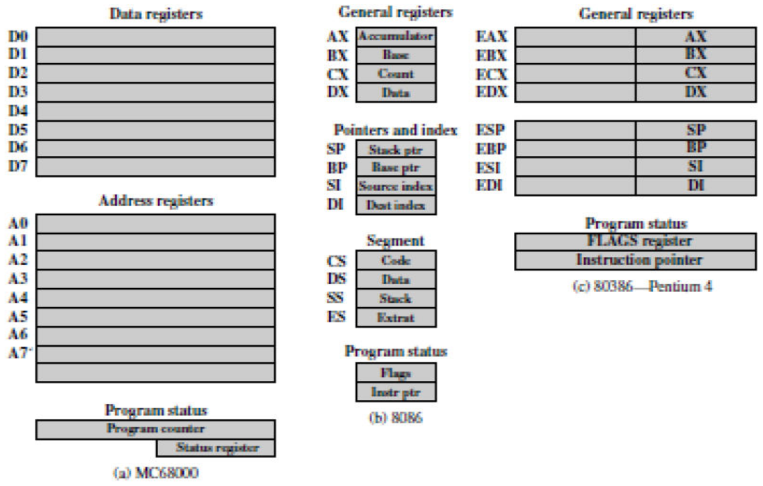
There are a variety of processor registers that are employed to control the operation of the processor. Most of these, on most machines, are not visible to the user. Some of them may be visible to machine instructions executed in a control or operating system mode.

Four Registers are essential for instruction Execution

- **Program counter (PC):** Contains the address of an instruction to be fetched
- **Instruction register (IR):** Contains the instruction most recently fetched
- **Memory address register (MAR):** Contains the address of a location in memory
- **Memory buffer register (MBR):** Contains a word of data to be written to memory or the word most recently read

Many processor designs include a register or set of registers, often known as the *program status word* (PSW), that contain status information. The PSW typically contains condition codes plus other status information.

.



(a) MC68000 (b) 8086 (c) 80386—Pentium 4

Example Microprocessor Register Organisation

Processor level:

The primary function of a processor unit is to execute sequence of instructions stored in a memory, which is external to the processor unit.

The sequence of operations involved in processing an instruction constitutes an instruction cycle,which can be subdivided into 3 major phases:

1. Fetch cycle 2. Decode cycle 3. Execute cycle

Basic instruction cycle

To perform fetch, decode and execute cycles the processor unit has to perform set of operations called micro-operations.

Single bus organization of processor unit shows how the building blocks of processor unit are organised and how they are interconnected.

They can be organised in a variety of ways, in which the arithmetic and logic unit and all processor registers are connected through a single common bus.

It also shows the external memory bus connected to memory address(MAR) and data register(MDR).

Single Bus Organisation of processor

➢ The registers Y,Z and Temp are used only by the processor unit for temporary storage during the execution of some instructions.

➢ These registers are never used for storing data generated by one instruction for later use by another instruction.

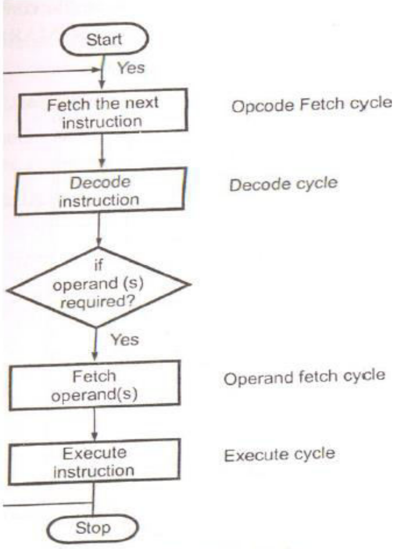
➢ The programmer cannot access these registers.

➢ The IR and the instruction decoder are integral parts of the control circuitry in the processing unit.

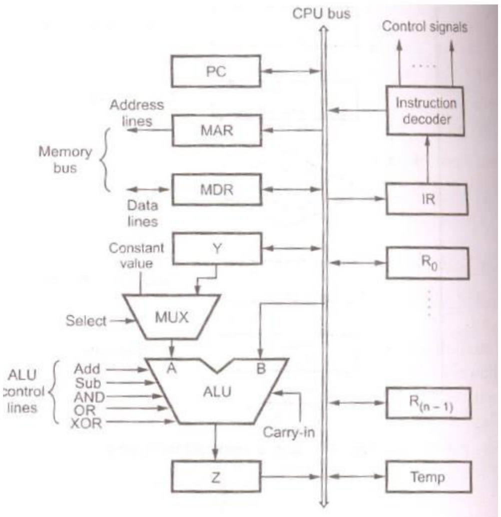
➢ All other registers and the ALU are used for storing and manipulating data.

➢ The data registers, ALU and the interconnecting bus is referred to as data path.

- Register R_0 through $R_{(n-1)}$ are the processor registers.
- The number and use of these register vary considerably from processor to processor.



Basic instruction cycle



Single bus organisation of processor

- These registers include general purpose registers and special purpose registers such as stack pointer, index registers and pointers.
- These are 2 options provided for A input of the ALU.
- The multiplexer(MUX) is used to select one of the two inputs.
- It selects either output of Y register or a constant number as an A input for the ALU according to the status of the select input.
- It selects output of Y when select input is 1 (select Y) and it selects a constant number when select input is 0(select C) as an input A for the multiplier.
- The constant number is used to increment the contents of program counter.

Video Content / Details of website for further learning (if any):
https://www.youtube.com/watch?v=Vt3lFnBwgpo
https://www.youtube.com/watch?v=Ngu1UbRAeqQ

Important Books/Journals for further learning including the page nos.:
John P. Hayes, “Computer Architecture and Organization”, Tata McGraw-Hill, 3rd edition, 1998, page no.:83 to 126

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

Estd. 2000

L - 05

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit I : Introduction to Computer Architecture & Organization

Date of Lecture:

Topic of Lecture: CPU organization

Introduction:

CPU acts as a brain of computer. It performs all calculations and controls all components. It carries out the instructions of a computer program, performs the basic arithmetical, logical and input and output operations of the system.

Prerequisite knowledge for Complete understanding and learning of Topic:

Microprocessor and microcontrollers

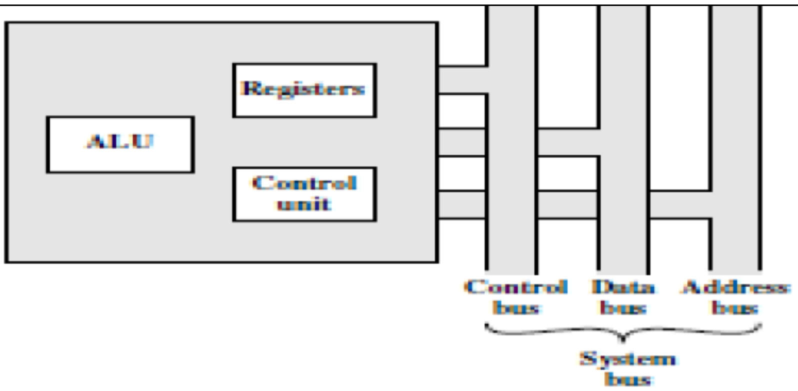
CPU organization:

To understand the organization of the processor, let us consider the requirements placed on the processor, the things that it must do:

- **Fetch instruction:** The processor reads an instruction from memory (register, cache, main memory).
- **Interpret instruction:** The instruction is decoded to determine what action is required.
- **Fetch data:** The execution of an instruction may require reading data from memory or an I/O module.
- **Process data:** The execution of an instruction may require performing some arithmetic or logical operation on data.
- **Write data:** The results of an execution may require writing data to memory or an I/O module.

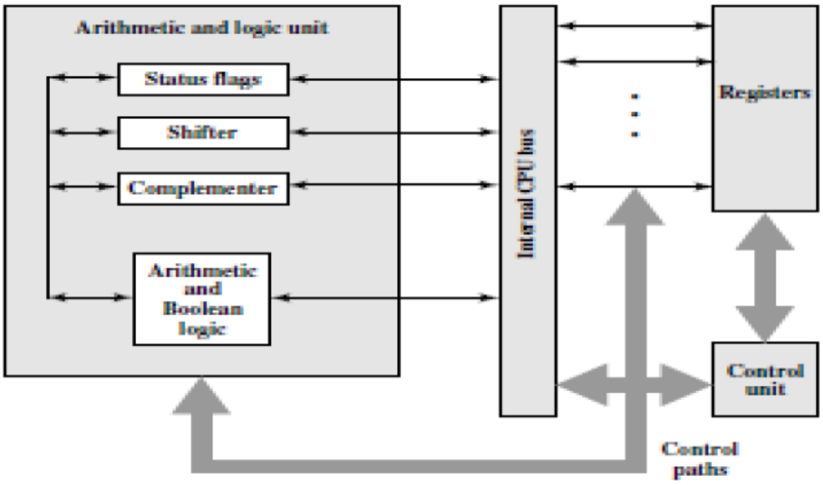
To do these things, it should be clear that the processor needs to store some data temporarily. In other words, the processor needs a small internal memory.

Figure is a simplified view of a processor, indicating its connection to the rest of the system via the system bus. The major components of the processor are an *arithmetic and logic unit* (ALU) and a *control unit* (CU). The ALU does the actual computation or processing of data. The control unit controls the movement of data and instructions into and out of the processor and controls the operation of the ALU. In addition, the figure shows a minimal internal memory, consisting of a set of storage locations, called *registers.*



The CPU With System Bus

Figure is a slightly more detailed view of the processor. The data transfer and logic control paths are indicated, including *internal processor bus which* is needed to transfer data between the various registers and the ALU because the ALU in fact operates only on data in the internal processor memory.



'Internal Structure of the CPU

Video Content / Details of website for further learning (if any):
https://www.youtube.com/watch?v=Za7ozdjE8VI

Important Books/Journals for further learning including the page nos.:
John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no. :137 to 160

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

L - 06

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit I : Introduction to Computer Architecture & Organization

Date of Lecture:

Topic of Lecture: Data representation- Fixed point numbers, Floating point numbers

Introduction:

Digital Computers use Binary number system to represent all types of information inside the computers. Alphanumeric characters are represented using binary bits (i.e., 0 and 1). Digital representations are easier to design, storage is easy, accuracy and precision are greater.

Prerequisite knowledge for Complete understanding and learning of Topic:

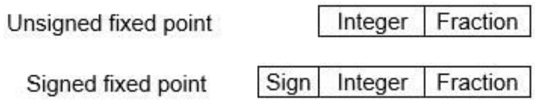
Microprocessor and microcontrollers

Data representation:

There are two major approaches to store real numbers (i.e., numbers with fractional component) in modern computing. These are (i) Fixed Point Notation and (ii) Floating Point Notation. In fixed point notation, there are a fixed number of digits after the decimal point, whereas floating point number allows for a varying number of digits after the decimal point.

Fixed-Point Representation:

This representation has fixed number of bits for integer part and for fractional part. For example, if given fixed-point representation is IIII.FFFF, then you can store minimum value is 0000.0001 and maximum value is 9999.9999. There are three parts of a fixed-point number representation: the sign field, integer field, and fractional field.

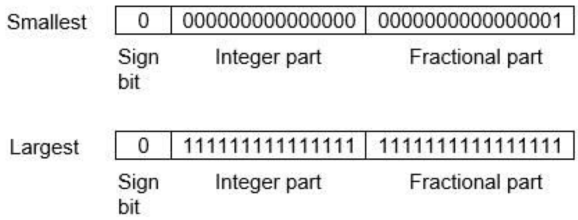

Unsigned fixed point | Integer | Fraction
Signed fixed point | Sign | Integer | Fraction

We can represent these numbers using:

- Signed representation: range from $-(2^{(k-1)}-1)$ to $(2^{(k-1)}-1)$, for k bits.
- 1's complement representation: range from $-(2^{(k-1)}-1)$ to $(2^{(k-1)}-1)$, for k bits.
- 2's complementation representation: range from $-(2^{(k-1)})$ to $(2^{(k-1)}-1)$, for k bits.

2's complementation representation is preferred in computer system because of unambiguous property and easier for arithmetic operations.

The advantage of using a fixed-point representation is performance and disadvantage is relatively limited range of values that they can represent. So, it is usually inadequate for numerical analysis as it does not allow enough numbers and accuracy. A number whose representation exceeds 32 bits would have to be stored inexactly.



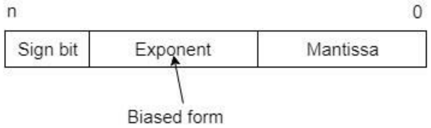
These are above smallest positive number and largest positive number which can be store in 32-bit representation as given above format. Therefore, the smallest positive number is $2^{-16} \approx$ 0.000015 approximate and the largest positive number is $(2^{15}-1)+(1-2^{-16})=2^{15}(1-2^{-16})$ =32768, and gap between these numbers is 2^{-16}.

Floating-Point Representation:

This representation does not reserve a specific number of bits for the integer part or the fractional part. Instead it reserves a certain number of bits for the number (called the mantissa or significand) and a certain number of bits to say where within that number the decimal place sits (called the exponent).

The floating number representation of a number has two part: the first part represents a signed fixed point number called mantissa. The second part of designates the position of the decimal (or binary) point and is called the exponent. The fixed point mantissa may be fraction or an integer. Floating -point is always interpreted to represent a number in the following form: Mxr^e.

Only the mantissa m and the exponent e are physically represented in the register (including their sign). A floating-point binary number is represented in a similar manner except that is uses base 2 for the exponent. A floating-point number is said to be normalized if the most significant digit of the mantissa is 1.



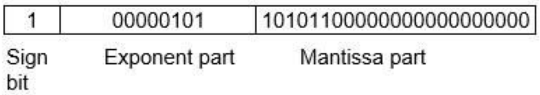
So, actual number is $(-1)^s(1+m)x2^{(e\text{-}Bias)}$, where *s* is the sign bit, *m* is the mantissa, *e* is the exponent value, and *Bias* is the bias number.

Note that signed integers and exponent are represented by either sign representation, or one's complement representation, or two's complement representation.

The floating point representation is more flexible. Any non-zero number can be represented in the normalized form of $\pm(1.b_1b_2b_3\ ...)_2x2^n$ This is normalized form of a number x.

Example: Suppose number is using 32-bit format: the 1 bit sign bit, 8 bits for signed exponent, and 23 bits for the fractional part. The leading bit 1 is not stored (as it is always 1 for a normalized number) and is referred to as a *"hidden bit*".

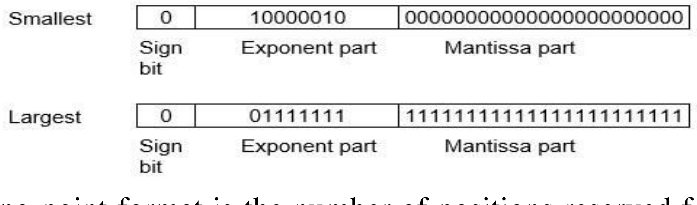
Then −53.5 is normalized as $-53.5=(-110101.1)_2=(-1.101011)x2^5$, which is represented as following below,



Where 00000101 is the 8-bit binary value of exponent value +5.

Note that 8-bit exponent field is used to store integer exponents $-126 \leq\ n \leq 127$.

The smallest normalized positive number that fits into 32 bits is $(1.00000000000000000000000)_2x2^{-126}=2^{-126}\approx1.18x10^{-38}$, and largest normalized positive number that fits into 32 bits is $(1.11111111111111111111111)_2x2^{127}=(2^{24}-1)x2^{104} \approx$ $3.40x10^{38}$. These numbers are represented as following below,



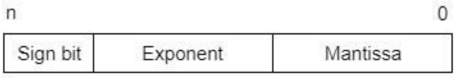
The precision of a floating-point format is the number of positions reserved for binary digits plus one (for the hidden bit). In the examples considered here the precision is 23+1=24.

The gap between 1 and the next normalized floating-point number is known as machine epsilon. the gap is $(1+2^{-23})-1=2^{-23}$for above example, but this is same as the smallest positive floating-point number because of non-uniform spacing unlike in the fixed-point scenario.

Note that non-terminating binary numbers can be represented in floating point representation, e.g., 1/3 = $(0.010101 \ldots)_2$ cannot be a floating-point number as its binary representation is non-terminating.

IEEE Floating point Number Representation:

IEEE (Institute of Electrical and Electronics Engineers) has standardized Floating-Point Representation as following diagram.



So, actual number is $(-1)^s(1+m)x2^{(e-Bias)}$, where *s* is the sign bit, *m* is the mantissa, *e* is the exponent value, and *Bias* is the bias number. The sign bit is 0 for positive number and 1 for negative number. Exponents are represented by or two's complement representation.

According to IEEE 754 standard, the floating-point number is represented in following ways:

- Half Precision (16 bit): 1 sign bit, 5 bit exponent, and 10 bit mantissa
- Single Precision (32 bit): 1 sign bit, 8 bit exponent, and 23 bit mantissa
- Double Precision (64 bit): 1 sign bit, 11 bit exponent, and 52 bit mantissa
- Quadruple Precision (128 bit): 1 sign bit, 15 bit exponent, and 112 bit mantissa

Special Value Representation:

There are some special values depended upon different values of the exponent and mantissa in the IEEE 754 standard.

- All the exponent bits 0 with all mantissa bits 0 represents 0. If sign bit is 0, then +0, else -0.
- All the exponent bits 1 with all mantissa bits 0 represents infinity. If sign bit is 0, then +∞, else -∞.
- All the exponent bits 0 and mantissa bits non-zero represents denormalized number.
- All the exponent bits 1 and mantissa bits non-zero represents error.

Video Content / Details of website for further learning (if any):

https://www.youtube.com/watch?v=nZBHKa5YLR4

https://www.youtube.com/watch?v=x59jnFr3198

Important Books/Journals for further learning including the page nos.:

John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no. :160 to 178

Course Faculty

Verified by HOD

L - 07

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit I : Introduction to Computer Architecture & Organization

Date of Lecture:

Topic of Lecture: Instruction Formats

Introduction:

An instruction format defines the layout of the bits of an instruction. An instruction format must include an opcode and, implicitly or explicitly, zero or more operands. Each explicit operand is referenced using one of the addressing modes.

Prerequisite knowledge for Complete understanding and learning of Topic:

Microprocessor and microcontrollers

An instruction format defines the layout of the bits of an instruction. An instruction format must include an opcode and, implicitly or explicitly, zero or more operands. Each explicit operand is referenced using one of the addressing modes. Key design issues in X86 instruction formats are:

Instruction Length

The most basic design issue to be faced is the instruction format length which is affected by, memory size, memory organization, bus structure, processor complexity, and processor speed. Beyond this basic trade-off, there are other considerations.

- Either the instruction length should be equal to the memory-transfer length or one should be a multiple of the other.
- Memory transfer rate has not kept up with increases in processor speed.
- Memory can become a bottleneck if the processor can execute instructions faster than it can fetch them. One solution to this problem is to use cache memory and another is to use shorter instructions.
- Instruction length should be a multiple of the character length, which is usually 8 bits, and of the length of fixed-point numbers.

Allocation of Bits

An equally difficult issue is how to allocate the bits in that format. For a given instruction length, more opcodes obviously mean more bits in the opcode field. For an instruction format of a given length, this reduces the number of bits available for addressing. There is one interesting refinement to this tradeoff, and that is the use of variable-length opcodes.

In this approach, there is a minimum opcode length but, for some opcodes, additional operations may be specified by using additional bits in the instruction. For a fixed-length instruction, this leaves fewer bits for addressing. Thus, this feature is used for those instructions that require fewer operands and/or less powerful addressing.

The following interrelated factors go into determining the use of the addressing bits.

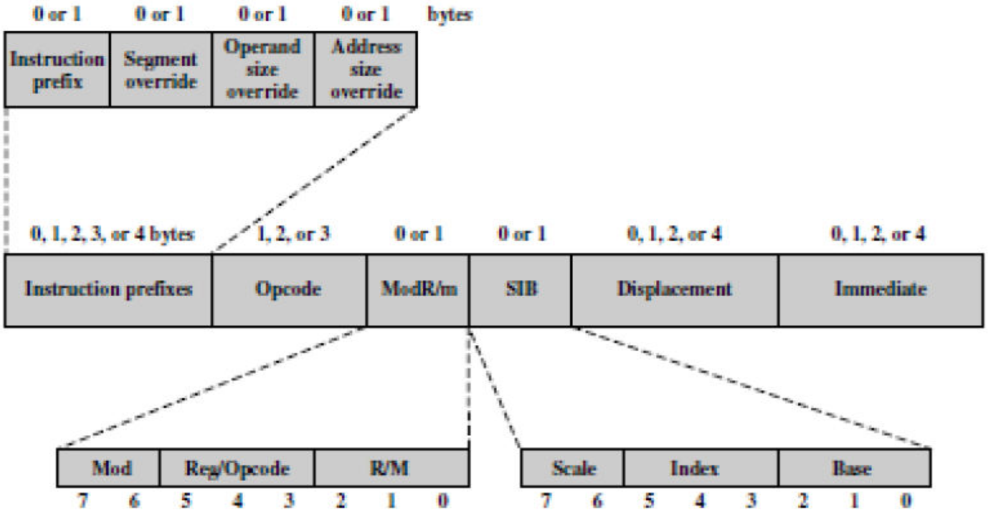
- **Number of addressing modes:** Sometimes an addressing mode can be indicated implicitly. In other cases, the addressing modes must be explicit, and one or more mode bits will be needed.
- **Number of operands:** Typical instructions on today’s machines provide for two operands. Each operand address in the instruction might require its own mode indicator, or the use of a mode indicator

could be limited to just one of the address fields.

• **Register versus memory:** A machine must have registers so that data can be brought into the processor for processing. With a single user-visible register (usually called the accumulator), one operand address is implicit and consumes no instruction bits. However, single-register programming is awkward and requires many instructions. Even with multiple registers, only a few bits are needed to specify the register. The more that registers can be used for operand references, the fewer bits are needed • **Number of register sets:** Most contemporary machines have one set of general-purpose registers, with typically 32 or more registers in the set. These registers can be used to store data and can be used to store addresses for displacement addressing

• **Address range:** For addresses that reference memory, the range of addresses that can be referenced is related to the number of address bits. Because this imposes a severe limitation, direct addressing is rarely used. With displacement addressing, the range is opened up to the length of the address register

• **Address granularity:** For addresses that reference memory rather than registers, another factor is the granularity of addressing. In a system with 16- or 32-bit words, an address can reference a word or a byte at the designer's choice. Byte addressing is convenient for character manipulation but requires, for a fixed-size memory, more address bits. Thus, the designer is faced with a host of factors to consider and balance.



X86 Instruction Format

Video Content / Details of website for further learning (if any):
https://www.youtube.com/watch?v=BrBZHteA3-0

Important Books/Journals for further learning including the page nos.:
John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no.:178 to 191

Course Faculty

Verified by HOD

L - 08

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit I : Introduction to Computer Architecture & Organization

Date of Lecture:

Topic of Lecture: Instruction types

Introduction:

An **instruction** is a binary pattern designed inside a microprocessor to perform a specific function. The entire group of instructions, called the **instruction set**, determines what functions the microprocessor can perform. These instructions can be classified into the following five functional categories: data transfer (copy) operations, arithmetic operations, logical operations, branching operations, and machine-control operations.

Prerequisite knowledge for Complete understanding and learning of Topic:

Microprocessor and microcontrollers

Depending on operation they perform, all instructions are divided in several groups:

Arithmetic Instructions
Branch Instructions
Data Transfer Instructions
Logic Instructions
Bit-oriented Instructions

The first part of each instruction, called MNEMONIC refers to the operation an instruction performs (copy, addition, logic operation etc.). Mnemonics are abbreviations of the name of operation being executed. For example:

INC R1 - Means: Increment register R1 (increment register R1);
LJMP LAB5 - Means: Long Jump LAB5 (long jump to the address marked as LAB5);
JNZ LOOP - Means: Jump if Not Zero LOOP (if the number in the accumulator is not 0, jump to the address marked as LOOP);

The other part of instruction, called OPERAND is separated from mnemonic by at least one whitespace and defines data being processed by instructions. Some of the instructions have no operand, while some of them have one, two or three. If there is more than one operand in an instruction, they are separated by a comma. For example:

RET - return from a subroutine;
JZ TEMP - if the number in the accumulator is not 0, jump to the address marked as TEMP;
ADD A,R3 - add R3 and accumulator;
CJNE A,#20,LOOP - compare accumulator with 20. If they are not equal, jump to the address marked as LOOP;

Arithmetic instructions

Arithmetic instructions perform several basic operations such as addition, subtraction, division, multiplication etc. After execution, the result is stored in the first operand. For example: ADD A,R1 - The result of addition (A+R1) will be stored in the accumulator.

Branch Instructions

There are two kinds of branch instructions: Unconditional jump instructions: upon their execution a jump to a new location from where the program continues execution is executed. Conditional jump instructions: a jump to a new program location is executed only if a specified condition is met. Otherwise, the program normally proceeds with the next instruction.

Data Transfer Instructions

Data transfer instructions move the content of one register to another. The register the content of which is moved remains unchanged. If they have the suffix “X” (MOVX), the data is exchanged with external memory.

Logic Instructions

Logic instructions perform logic operations upon corresponding bits of two registers. After execution, the result is stored in the first operand.

Bit-oriented Instructions

Similar to logic instructions, bit-oriented instructions perform logic operations. The difference is that these are performed upon single bits.

Video Content / Details of website for further learning (if any):
youtube.com/watch?v=BKUkIGc9JsE

Important Books/Journals for further learning including the page nos.:
John P. Hayes, “Computer Architecture and Organization”, Tata McGraw-Hill, 3rd edition, 1998, page no. : 191 to 211

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

Estd. 2000

L - 09

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit I : Introduction to Computer Architecture & Organization

Date of Lecture:

Topic of Lecture: Addressing modes

Introduction:

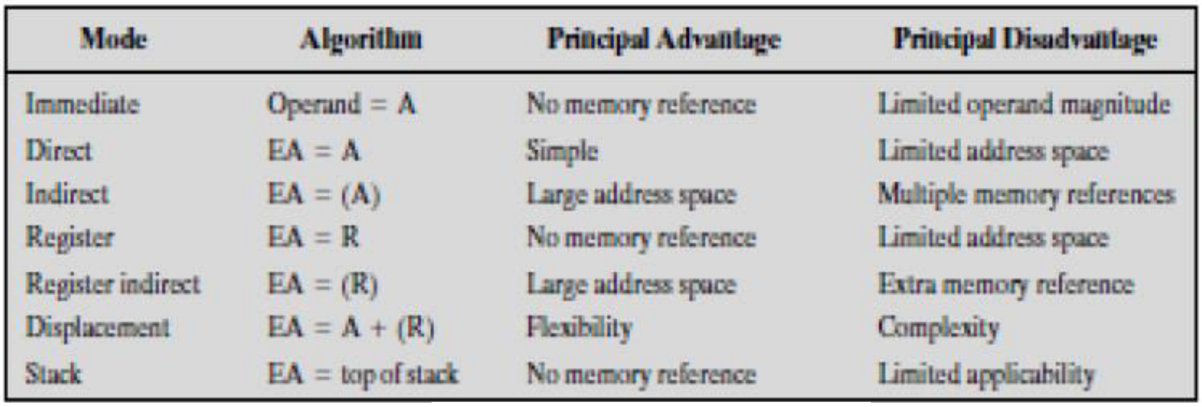
Addressing modes are the ways how architectures specify the address of an object they want to access. In GPR machines, an addressing mode can specify a constant, a register or a location in memory.

Prerequisite knowledge for Complete understanding and learning of Topic:

Microprocessor and microcontrollers

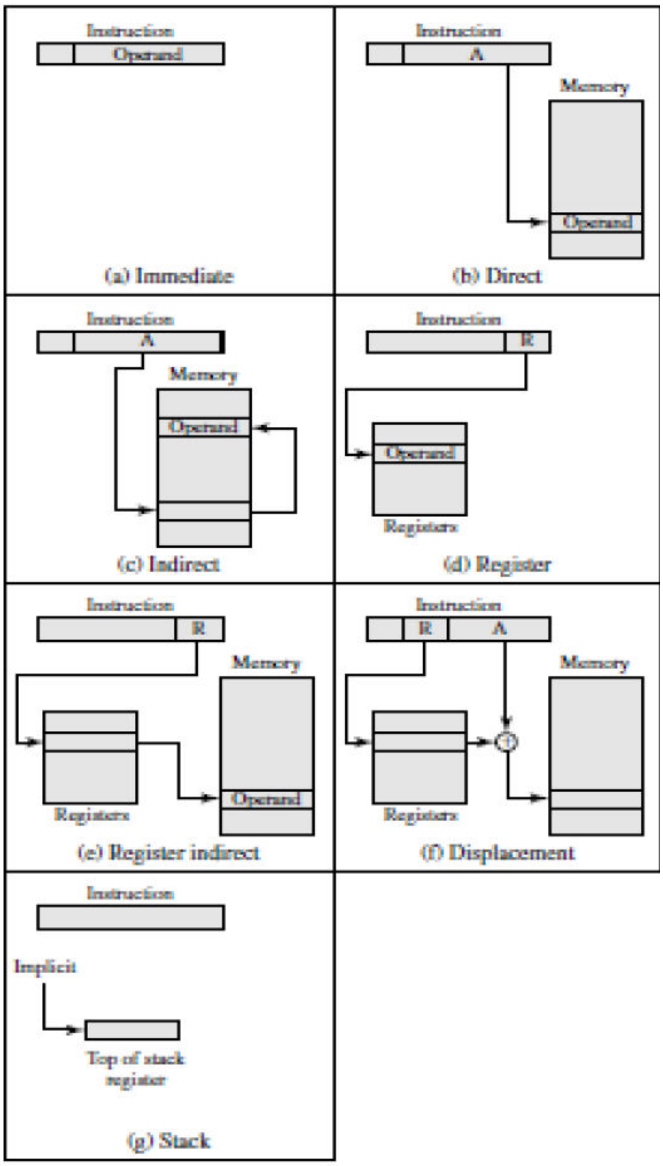
The address field or fields in a typical instruction format are relatively small. We should be able to reference a large range of locations in main memory. For this, a variety of addressing techniques has been
employed. The most common addressing techniques are:

- Immediate
- Direct
- Indirect
- Register
- Register indirect
- Displacement
- Stack



Mode	Algorithm	Principal Advantage	Principal Disadvantage
Immediate	Operand = A	No memory reference	Limited operand magnitude
Direct	EA = A	Simple	Limited address space
Indirect	EA = (A)	Large address space	Multiple memory references
Register	EA = R	No memory reference	Limited address space
Register indirect	EA = (R)	Large address space	Extra memory reference
Displacement	EA = A + (R)	Flexibility	Complexity
Stack	EA = top of stack	No memory reference	Limited applicability

Basic Addressing Modes



(a) Immediate

(b) Direct

(c) Indirect

(d) Register

(e) Register indirect

(f) Displacement

(g) Stack

A – contents of an address field in the instruction
R – contents of an address field in the instruction that refers to a register
EA – actual (effective) address of the location containing the referenced operand
(X) – contents of memory location X or register X

Video Content / Details of website for further learning (if any):
https://www.youtube.com/watch?v=_CH4cm5PhK8

Important Books/Journals for further learning including the page nos.:
John P. Hayes, “Computer Architecture and Organization”, Tata McGraw-Hill, 3rd edition, 1998, page no.:184 to 191

Course Faculty

Verified by HOD

L - 10

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 16ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit II : Arithmetic and Data Path Design **Date of Lecture:**

Topic of Lecture: Fixed point arithmetic, Addition, subtraction

Introduction:

Data path is the component of the processor that performs arithmetic operations. The collection of state elements, computation elements, and interconnections that together provide a conduit for the flow and transformation of data in the processor during execution.

Prerequisite knowledge for Complete understanding and learning of Topic:

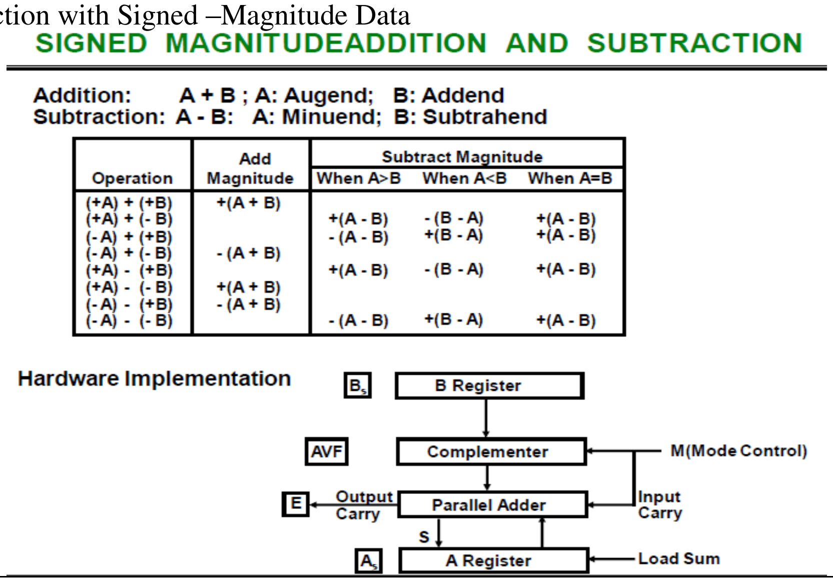
Microprocessors and Microcontrollers

Fixed point arithmetic, Addition, subtraction:

Data is manipulated by using the arithmetic instructions in digital computers. Data is manipulated to produce results necessary to give solution for the computation problems. The Addition, subtraction, multiplication and division are the four basic arithmetic operations. If we want then we can derive other operations by using these four operations. Fixed-point numbers are used to represent integers or fractions. We can have signed or unsigned negative numbers. Fixed-point addition is the simplest arithmetic operation.

There are three ways of representing negative fixed point -binary numbers signed magnitude, signed 1's complement or signed 2's complement. Most computers use the signed magnitude representation for the mantissa.

Addition and Subtraction with Signed –Magnitude Data



SIGNED MAGNITUDEADDITION AND SUBTRACTION

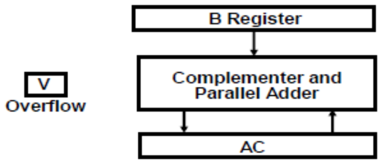
Addition: A + B ; A: Augend; B: Addend
Subtraction: A - B: A: Minuend; B: Subtrahend

Operation	Add Magnitude	Subtract Magnitude		
		When A>B	When A<B	When A=B
(+A) + (+B)	+(A + B)			
(+A) + (- B)		+(A - B)	- (B - A)	+(A - B)
(- A) + (+B)		- (A - B)	+(B - A)	+(A - B)
(- A) + (- B)	- (A + B)			
(+A) - (+B)		+(A - B)	- (B - A)	+(A - B)
(+A) - (- B)	+(A + B)			
(- A) - (+B)	- (A + B)			
(- A) - (- B)		- (A - B)	+(B - A)	+(A - B)

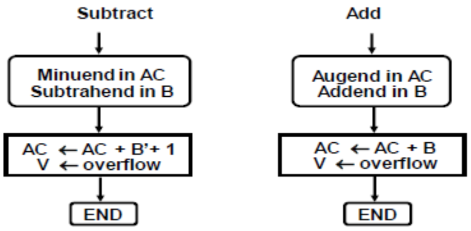
Hardware Implementation

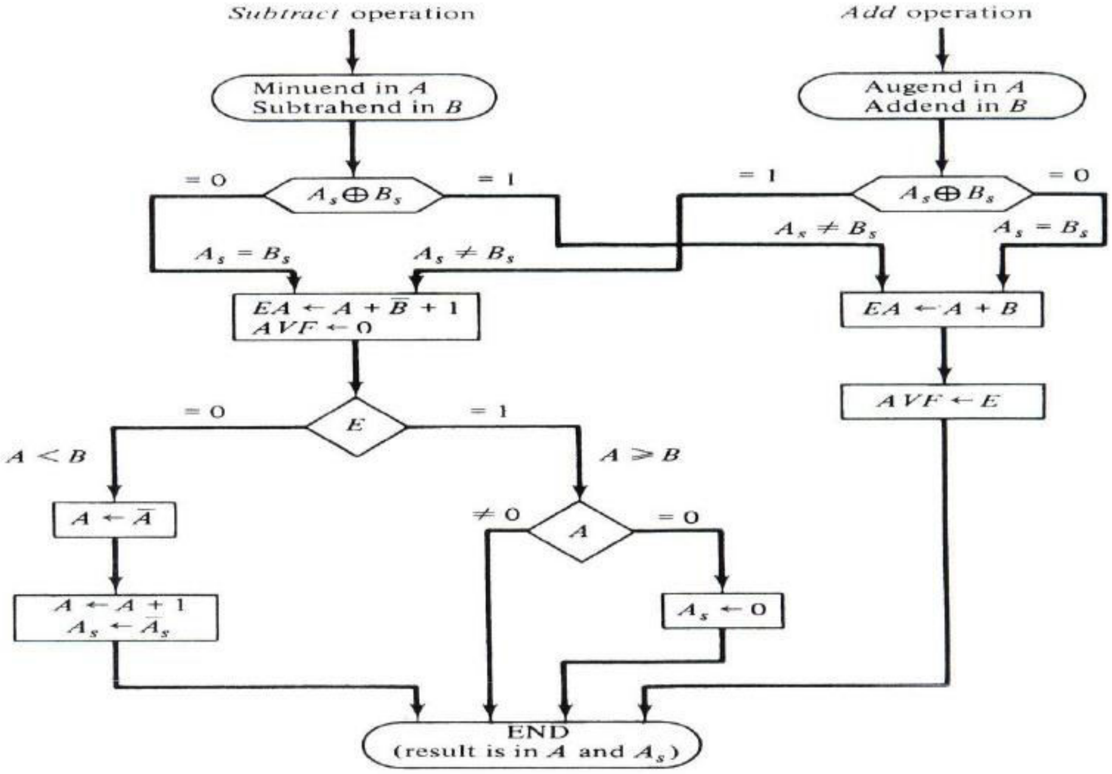
SIGNED 2'S COMPLEMENT ADDITION AND SUBTRACTION

Hardware



Algorithm





Flowchart for add and subtract operations.

Video Content / Details of website for further learning (if any):
https://www.youtube.com/watch?v=W9lUrAgxojo

Important Books/Journals for further learning including the page nos.:
John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no.:223 to 233

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

L - 11

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 16ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit II : Arithmetic and Data Path Design **Date of Lecture:**

Topic of Lecture: Multiplication and division

Introduction:

Arithmetic operations like addition, subtraction, multiplication, division are basic operations to be implemented in digital computers using basic gates likr AND, OR, NOR, NAND etc. Among all the arithmetic operations if we can implement addition then it is easy to perform multiplication (by repeated addition), subtraction (by negating one operand) or division (repeated subtraction).

Prerequisite knowledge for Complete understanding and learning of Topic:

Digital Electronics, Microprocessors and Microcontrollers

Multiplication Algorithm:

In the beginning, the multiplicand is in B and the multiplier in Q. Their corresponding signs are in Bs and Qs respectively. We compare the signs of both A and Q and set to corresponding sign of the product since a double-length product will be stored in registers A and Q. Registers A and E are cleared and the sequence counter SC is set to the number of bits of the multiplier. Since an operand must be stored with its sign, one bit of the word will be occupied by the sign and the magnitude will consist of n-1 bits.

Now, the low order bit of the multiplier in Qn is tested. If it is 1, the multiplicand (B) is added to present partial product (A), 0 otherwise. Register EAQ is then shifted once to the right to form the new partial product. The sequence counter is decremented by 1 and its new value checked. If it is not equal to zero, the process is repeated and a new partial product is formed. When SC = 0 we stops the process.

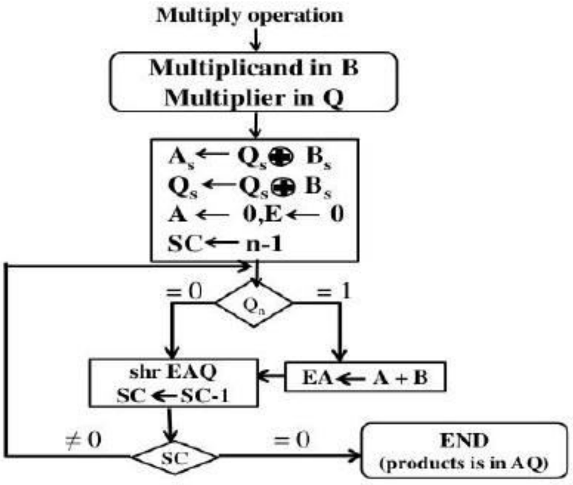
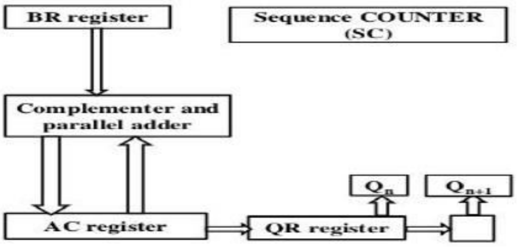


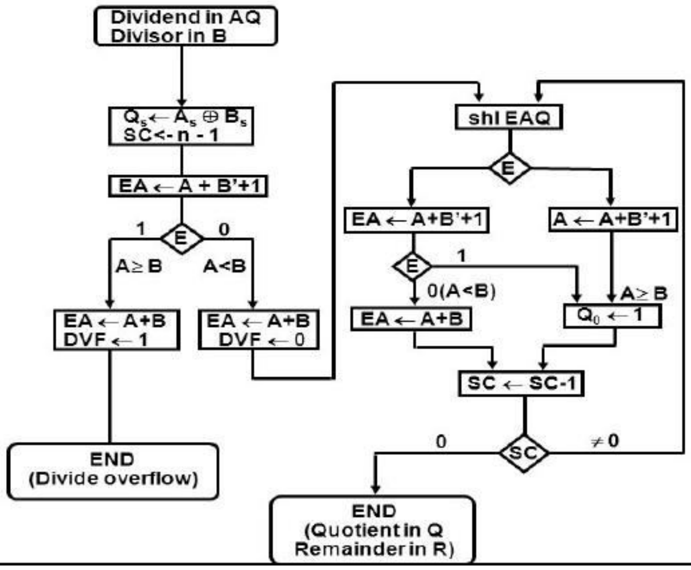
Figure: Flowchart for multiply operation.

Hardware for Booth Algorithm

- Sign bits are not separated from the rest of the registers
- rename registers A,B, and Q as AC,BR and QR respectively
- Q_n designates the least significant bit of the multiplier in register QR
- Flip-flop Qn+1 is appended to QR to facilitate a double bit inspection of the multiplier



FLOWCHART OF DIVIDE OPERATION



Video Content / Details of website for further learning (if any):
https://www.youtube.com/watch?v=prGwhrGkTMI&list=PL4p0tg5hj_ElgBaXw9YYbsdTDv62xgmOa

Important Books/Journals for further learning including the page nos.:
John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no. :233 to 251

Course Faculty

Verified by HOD

L - 12

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 16ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit II : Arithmetic and Data Path Design **Date of Lecture:**

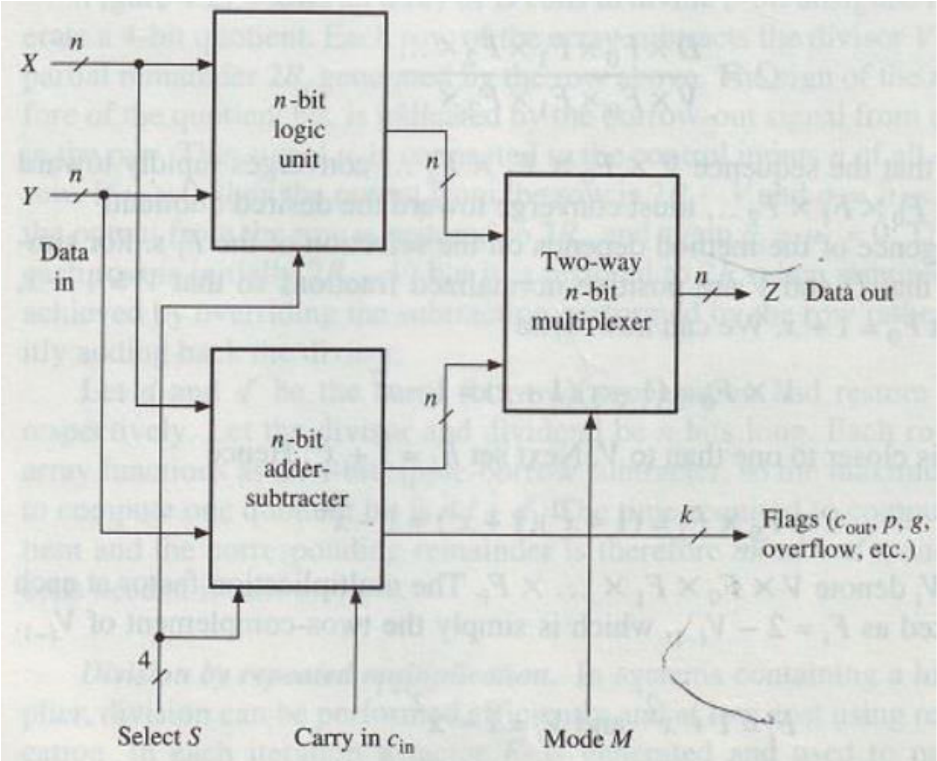
Topic of Lecture: Combinational and sequential ALUs

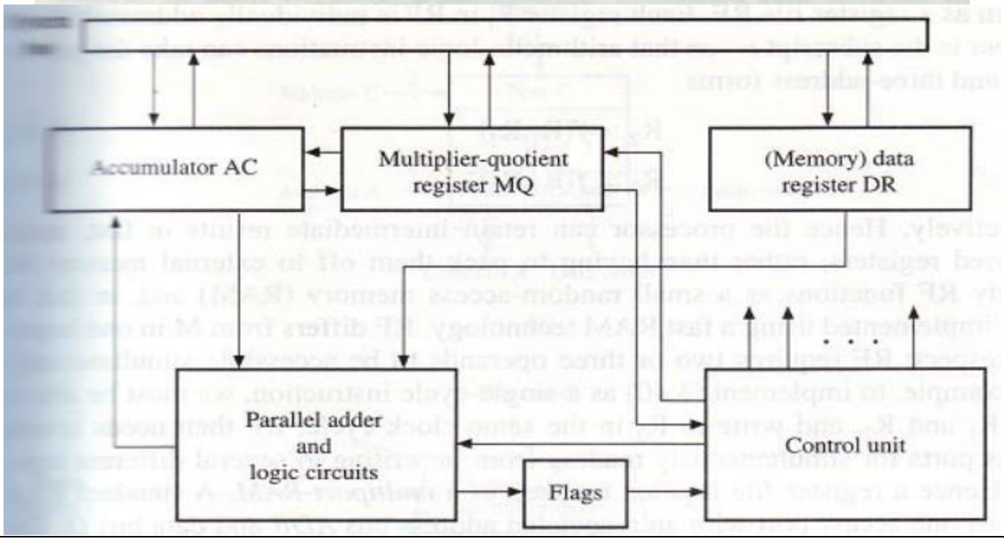
Introduction:
The various circuits used to execute data processing instructions are usually combined into a single circuit called an arithmetic logic unit (ALU).

Prerequisite knowledge for Complete understanding and learning of Topic:
Digital Electronics, Microprocessors and Microcontrollers

Combinational and sequential ALUs
The various circuits used to execute data processing instructions are usually combined into a single circuit called an arithmetic logic unit (ALU).
The complexity of ALU is determined by the way in which its arithmetic instructions are realized.
Two types of ALU:
1. Combinational ALU
2. Sequential ALU
It combines the functions of a two's-complement adder-subtracter with those of a circuit that generates word-based logic functions of the form *f(X,Y)*, for example, AND, XOR, NOT.
It implements most of a CPU's fixed point data-processing instructions.
Addition
AC:=AC+DR
Subtraction
AC:=AC-DR
Multiplication
AC.MQ:=DR X MQ
Division
AC.MQ:=MQ/DR
AND
AC:=AC and DR
OR
AC:=AC or DR
EX-OR
AC:=AC xor DR
NOT
AC:=not(AC)
Three one-word registers: AC, DR, MQ
AC and MQ are organized as a single register AC.MQ capable of left and right shifting.
DR can serve as a memory data register to store data addressed by an instruction address field ADR. Then DR can be replaced by M(ADR).

Register is selected by:
RA (number) selects the register to put on busA (data)
RB (number) selects the register to put on busB (data)
RW (number) selects the register to be written via busW (data) when Write Enable is
Clock input (CLK)
The CLK input is a factor ONLY during write operation.During read operation, behaves as a combinational logic block:RA or RB valid => bus A or busB valid after “access time.”




Accumulator AC
Multiplier-quotient register MQ
(Memory) data register DR
Parallel adder and logic circuits
Flags
Control unit
. . .

Video Content / Details of website for further learning (if any):
https://www.youtube.com/watch?v=33Do1NM5rnM

Important Books/Journals for further learning including the page nos.:
John P. Hayes, “Computer Architecture and Organization”, Tata McGraw-Hill, 3rd edition, 1998, page no. :252 to 260

Course Faculty

Verified by HOD

L - 13

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 16ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit II : Arithmetic and Data Path Design **Date of Lecture:**

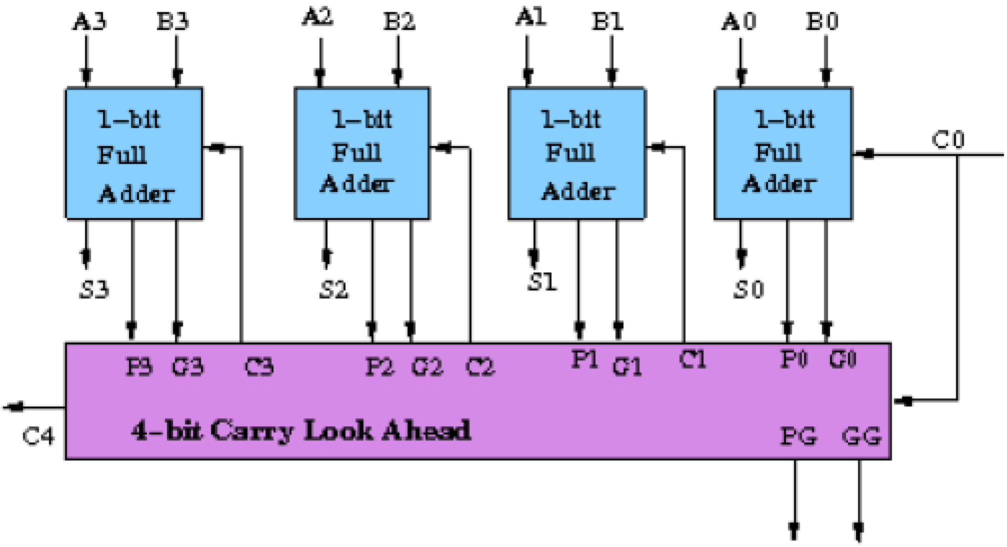
Topic of Lecture: Carry look ahead adder, Robertson Algorithm

Introduction:
A **carry-lookahead adder** (CLA) is a type of adder used in digital logic A carry-lookahead adder improves speed by reducing the amount of time required to determine carry bits. It can be contrasted with the simpler, but usually slower. The carry-lookahead adder calculates one or more carry bits before the sum, which reduces the wait time to calculate the result of the larger value bits.

James **Robertson** have designed a **multiply algorithm** for two's complement operands. This **algorithm** permits a unitary approach to the operands, irrespective of their signs. The sign bits are taken into consideration in the same measure as magnitude bits.

Prerequisite knowledge for Complete understanding and learning of Topic:
Digital Electronics, Microprocessors and Microcontrollers

Carry look ahead adder:
To reduce the computation time, there are faster ways to add two binary numbers by using carry lookahead adders. They work by creating two signals P and G known to be Carry Propagator and Carry Generator. The carry propagator is propagated to the next level whereas the carry generator is used to generate the output carry , regardless of input carry. The block diagram of a 4-bit Carry Lookahead Adder is shown here below –



$P_i = A_i \oplus B_i$

$G_i = A_i \cdot B_i$

The output sum and carry can be expressed as

$Sum_i = P_i \oplus C_i$

$C_i+1 = G_i + (P_i \cdot C_i)$

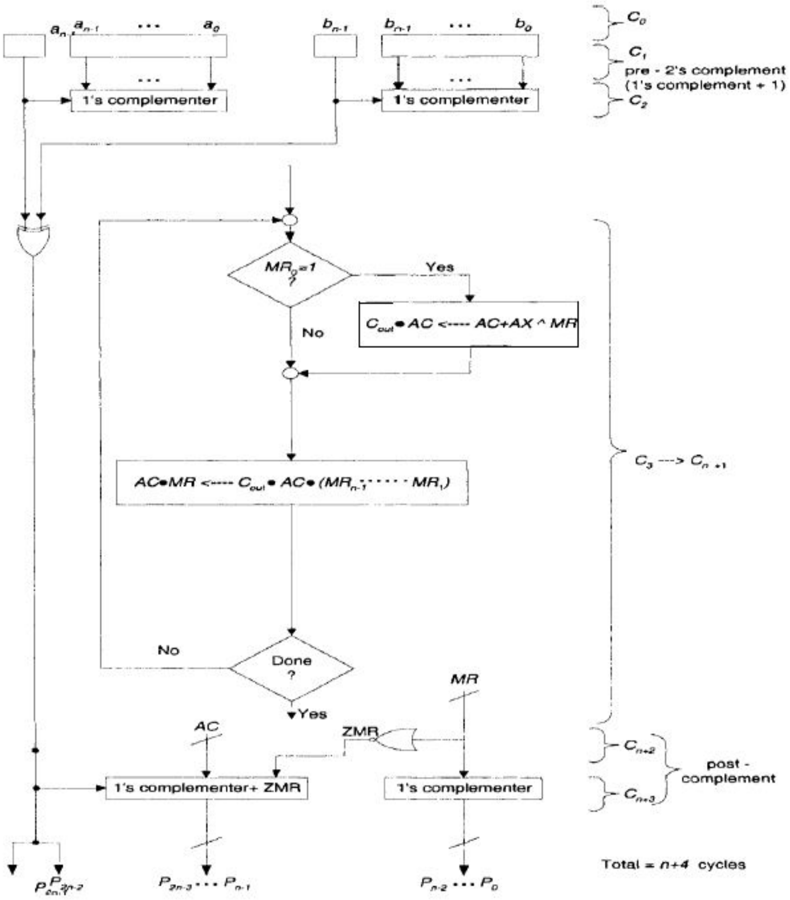
Robertson algorithm:

Consider two unsigned binary numbers X and Y . We want to multiply these numbers. The basic algorithm is similar to the one used in multiplying the numbers on pencil and paper. The main operations involved are shift and add.

There are two algorithms are used

1. Robertson algorithm

2 .Booth algorithm



Video Content / Details of website for further learning (if any):

https://www.youtube.com/watch?v=_bJ53XErKY8

https://www.youtube.com/watch?v=rhANNFMi9pY

Important Books/Journals for further learning including the page nos.:

John P. Hayes, “Computer Architecture and Organization”, Tata McGraw-Hill, 3rd edition, 1998, page no.:228 to 238

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

L - 14

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 16ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit II : Arithmetic and Data Path Design **Date of Lecture:**

Topic of Lecture: Booth's algorithm

Introduction:

Booth's algorithm results in reduction in the number of add/subtract steps needed (as compared to the Robertson's algorithm) if the multiplier contains runs (or sequences) of 1s or 0s.

Prerequisite knowledge for Complete understanding and learning of Topic:

Digital Electronics, Microprocessors and Microcontrollers

Booth's algorithm:

Booth algorithm gives a procedure for multiplying binary integers in signed- 2's complement representation.

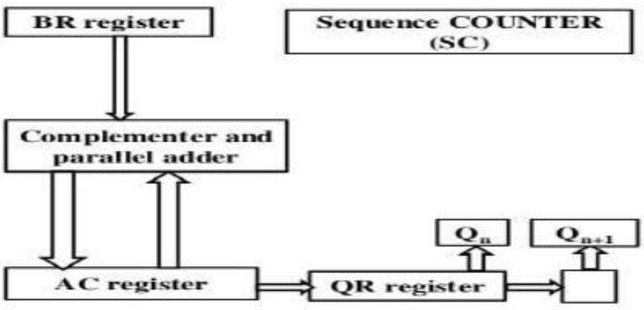
It operates on the fact that strings of 0's in the multiplier require no addition but just shifting, and a string of 1's in the multiplier from bit weight 2k to weight 2m can be treated as 2k+1 – 2m.

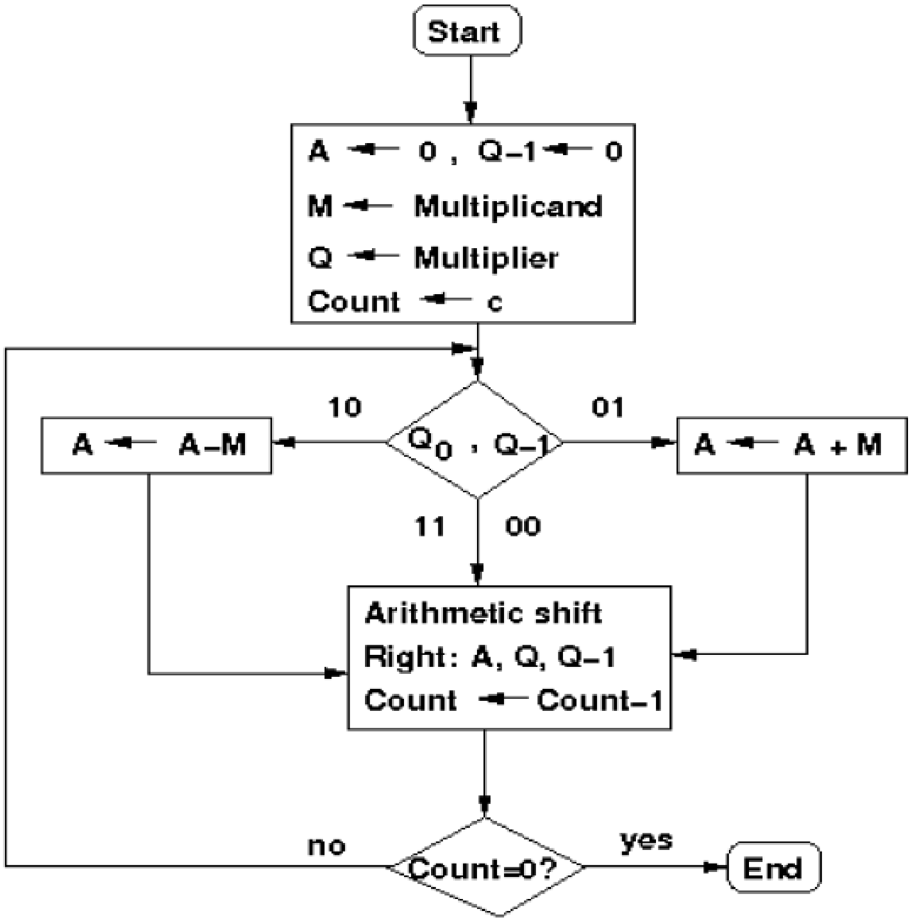
For example, the binary number 001110 (+14) has a string 1's from 23 to 21 (k=3, m=1). The number can be represented as 2k+1 – 2m. = 24 – 21 = 16 – 2 = 14. Therefore, the multiplication M X 14, where M is the multiplicand and 14 the multiplier, can be done as M X 24 – M X 21 .

Thus the product can be obtained by shifting the binary multiplicand M four times to the left and subtracting M shifted left once.

Hardware for Booth Algorithm

- Sign bits are not separated from the rest of the registers
- rename registers A,B, and Q as AC,BR and QR respectively
- Q_n designates the least significant bit of the multiplier in register QR
- Flip-flop Qn+1 is appended to QR to facilitate a double bit inspection of the multiplier





Video Content / Details of website for further learning (if any):
https://www.youtube.com/watch?v=QFXaddi-Ag8

Important Books/Journals for further learning including the page nos.:
John P. Hayes, “Computer Architecture and Organization”, Tata McGraw-Hill, 3rd edition, 1998, page no. :230 to 240

Course Faculty

Verified by HOD

L - 15

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 16ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit II : Arithmetic and Data Path Design **Date of Lecture:**

Topic of Lecture: Non-restoring division algorithm

Introduction:
A **division algorithm** is an algorithm which, given two integers N and D, computes their quotient and/or remainder, the result of Euclidean division. Division algorithms fall into two main categories: slow division and fast division. Slow division algorithms produce one digit of the final quotient per iteration. Examples of slow division include restoring, non-performing restoring, non-restoring, and SRT division.

Prerequisite knowledge for Complete understanding and learning of Topic:
Digital Electronics, Microprocessors and Microcontrollers

Now, here perform Non-Restoring division, it is less complex than the restoring one because simpler operation are involved i.e. addition and subtraction, also now restoring step is performed. In the method, rely on the sign bit of the register which initially contain zero named as A.

Here is the flow chart given below.

Let's pick the step involved:

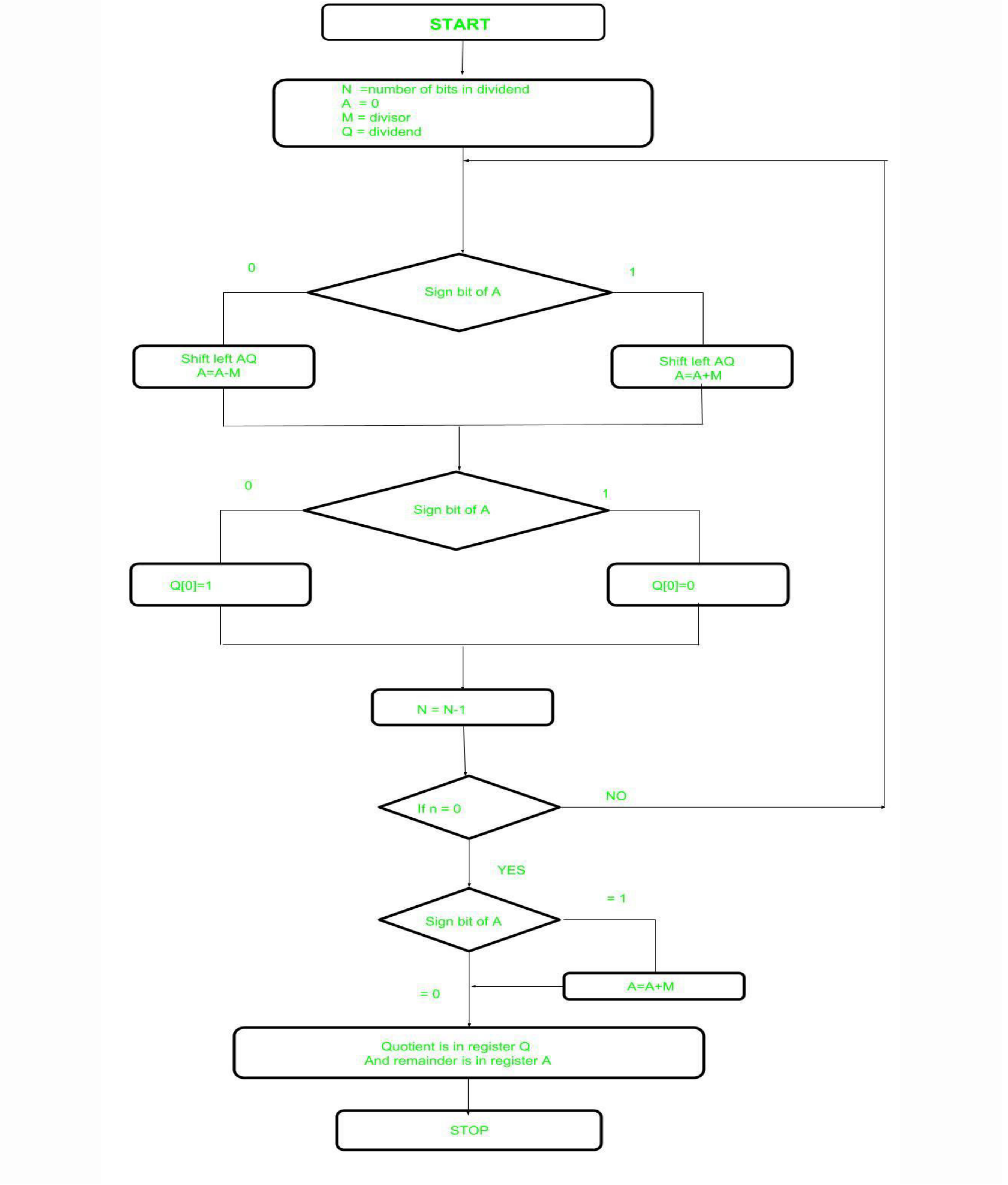
- **Step-1:** First the registers are initialized with corresponding values (Q = Dividend, M = Divisor, A = 0, n = number of bits in dividend)
- **Step-2:** Check the sign bit of register A
- **Step-3:** If it is 1 shift left content of AQ and perform A = A+M, otherwise shift left AQ and perform A = A-M (means add 2's complement of M to A and store it to A)
- **Step-4:** Again the sign bit of register A
- **Step-5:** If sign bit is 1 Q[0] become 0 otherwise Q[0] become 1 (Q[0] means least significant bit of register Q)
- **Step-6:** Decrements value of N by 1
- **Step-7:** If N is not equal to zero go to **Step 2** otherwise go to next step
- **Step-8:** If sign bit of A is 1 then perform A = A+M
- **Step-9:** Register Q contain quotient and A contain remainder

Examples: Perform Non_Restoring Division for Unsigned Integer

Dividend =11

Divisor =3

-M =11101



Video Content / Details of website for further learning (if any):
https://www.youtube.com/watch?v=f6A3ySUdT80

Important Books/Journals for further learning including the page nos.:
John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no. :248 to 249

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

Estd. 2000

L - 16

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 16ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit II : Arithmetic and Data Path Design **Date of Lecture:**

Topic of Lecture: Floating point arithmetic

Introduction:

In computing, **floating-point arithmetic** (**FP**) is arithmetic using formulaic representation of real numbers as an approximation to support a trade-off between range and precision. For this reason, floating-point computation is often found in systems which include very small and very large real numbers, which require fast processing times.

Prerequisite knowledge for Complete understanding and learning of Topic:

Digital Electronics, Microprocessors and Microcontrollers

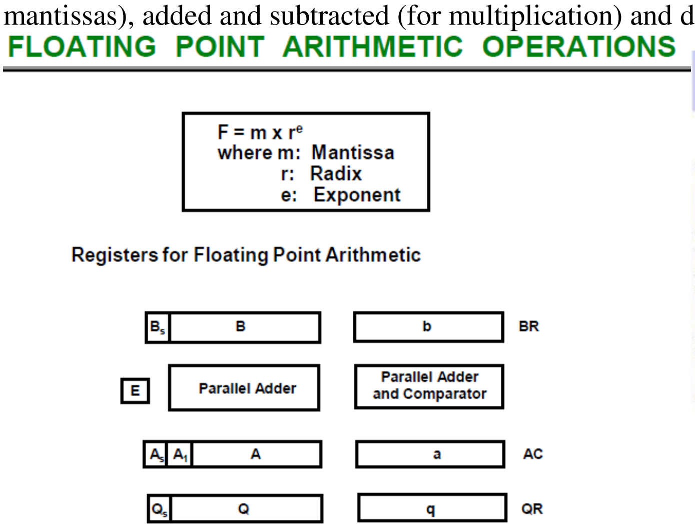
Floating point arithmetic:

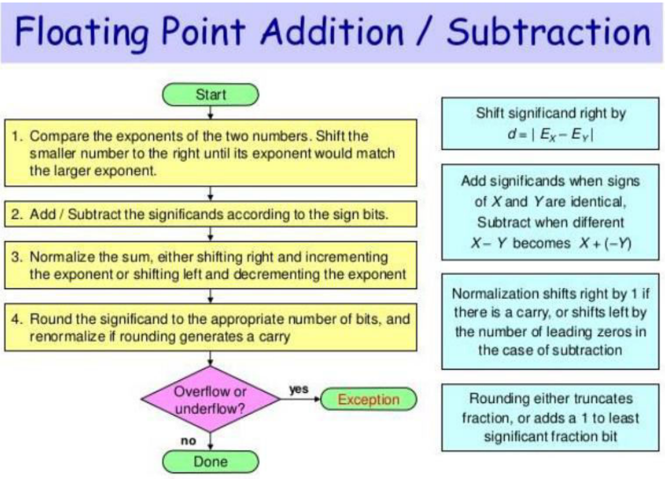
There are two part of a floating-point number in a computer - a mantissa m and an exponent e.

The two parts represent a number generated from multiplying m times a radix r raised to the value of e. Thus m x r_e

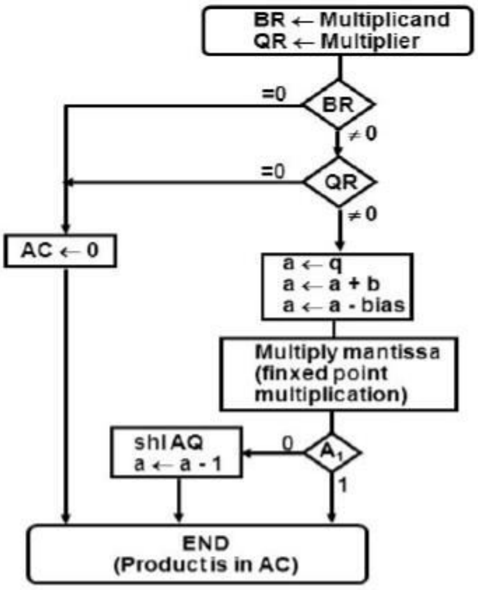
The mantissa may be a fraction or an integer. The position of the radix point and the value of the radix r are not included in the registers. Floating-point representation increases the range of numbers for a given register. Consider a computer with 48-bit words. Arithmetic operations with floating-point numbers are more complicated than with fixed-point numbers. Their execution also takes longer time and requires more complex hardware. Adding or subtracting two numbers requires first an alignment of the radix point since the exponent parts must be made equal before adding or subtracting the mantissas. Floating-point multiplication and division need not do an alignment of the mantissas. Multiplying the two mantissas and adding the exponents can form the product. Dividing the mantissas and subtracting the exponents perform division.

The operations done with the mantissas are the same as in fixed-point numbers, so the two can share the same registers and circuits. The operations performed with the exponents are compared and incremented (for aligning the mantissas), added and subtracted (for multiplication) and division), and decremented (to normalize the result).

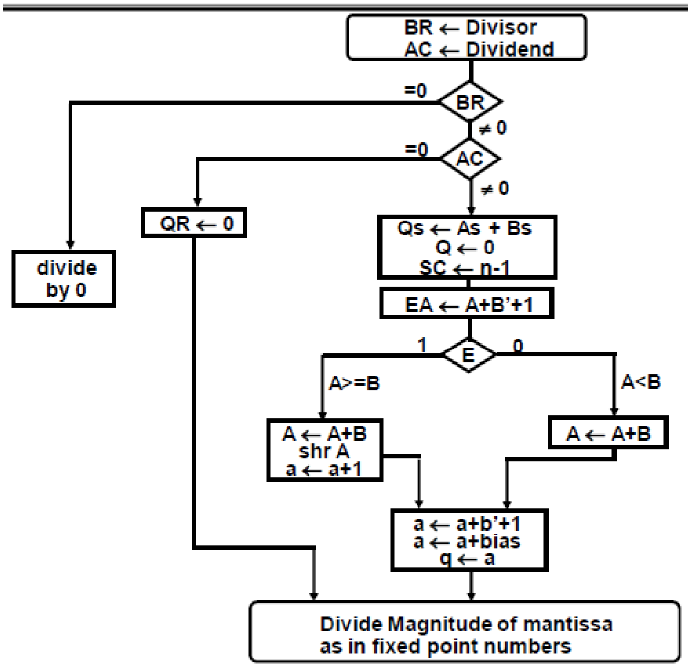

FLOATING POINT ARITHMETIC OPERATIONS
F = m x r^e
where m: Mantissa
r: Radix
e: Exponent
Registers for Floating Point Arithmetic
Bs
B
b
BR
E
Parallel Adder
Parallel Adder and Comparator
As
A1
A
a
AC
Qs
Q
q
QR


Floating Point Addition / Subtraction
Start
1. Compare the exponents of the two numbers. Shift the smaller number to the right until its exponent would match the larger exponent.
2. Add / Subtract the significands according to the sign bits.
3. Normalize the sum, either shifting right and incrementing the exponent or shifting left and decrementing the exponent
4. Round the significand to the appropriate number of bits, and renormalize if rounding generates a carry
Overflow or underflow?
yes
Exception
no
Done
Shift significand right by d = | Ex – Ey |
Add significands when signs of X and Y are identical, Subtract when different X – Y becomes X + (–Y)
Normalization shifts right by 1 if there is a carry, or shifts left by the number of leading zeros in the case of subtraction
Rounding either truncates fraction, or adds a 1 to least significant fraction bit

FLOATING POINT MULTIPLICATION



FLOATING POINT DIVISION



Video Content / Details of website for further learning (if any):
https://www.youtube.com/watch?v=XOMTNy2qiZ0

Important Books/Journals for further learning including the page nos.:
John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no. :266 to 248

Course Faculty

Verified by HOD

L - 17

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 16ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit II : Arithmetic and Data Path Design **Date of Lecture:**

Topic of Lecture: Coprocessor

Introduction:

Coprocessor is a separate instruction set processor means it has own instruction set supporting the special complex function. Coprocessor is closely coupled to the CPU and whose instruction and registers are direct extensions of the CPU**.**

Prerequisite knowledge for Complete understanding and learning of Topic:

Digital Electronics, Microprocessors and Microcontrollers

Coprocessor:

A **coprocessor** is a computer processor used to supplement the functions of the primary processor (the CPU). Operations performed by the coprocessor may be floating point arithmetic, graphics, signal processing, string processing, cryptography or I/O interfacing with peripheral devices. By offloading processor-intensive tasks from the main processor, coprocessors can accelerate system performance. Coprocessors allow a line of computers to be customized, so that customers who do not need the extra performance do not need to pay for it.

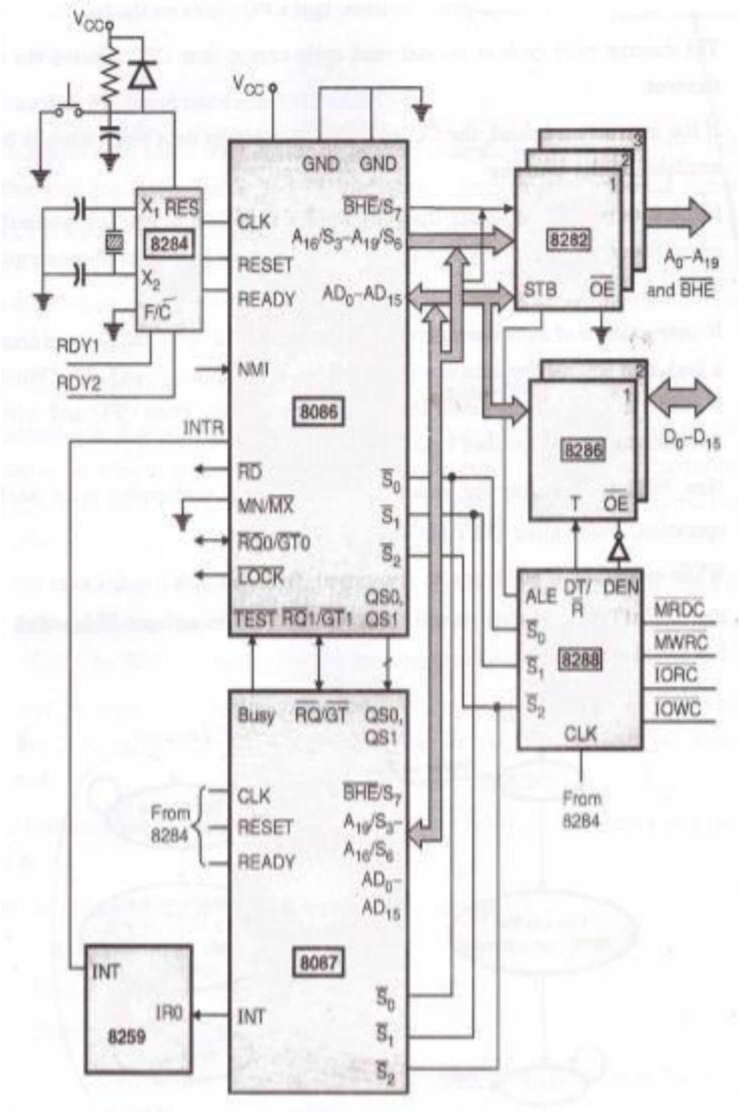
The 8087 was tightly integrated with the 8086/8088 and responded to floating-point machine code operation codes inserted in the 8088 instruction stream. An 8088 processor without an 8087 could not interpret these instructions, requiring separate versions of programs for FPU and non-FPU systems, or at least a test at run time to detect the FPU and select appropriate mathematical library functions.

8087 Math Coprocessor:

1. Intel 8087 is a processor with architecture and instruction set optimized for performing complicated arithmetic operations.
2. An 8087 is along with the host microprocessor 8086 rather than serving as the main processor itself. Therefore, it is referred to as the coprocessor.
3. An 8087 instruction may perform a given mathematical computation 100 times faster than the equivalent sequence of 8086 instructions.
4. 8087 is an actual processor with its own specialized instruction set. Instructions for 8087 are written in the program as needed, interspersed with 8086 instructions.
5. As the 8086 fetches instruction bytes from the memory and puts them in its queue, the 8087 also reads these instruction bytes and puts them in its queue. The 8087 decodes each instruction that comes into its queue.
6. When 8087 decodes an instruction from its queue and finds that it is an 8086 instruction, the 8087 simply treats the instruction as NOP.
7. Likewise, when the 8086 decodes an instruction from its queue and finds that it is an 8087 instruction, the 8086 simply treats the instruction as NOP or in some cases reads a data word from the memory for the 8087.

8. Each processor decodes all instructions in the fetched instruction byte stream but executes only its own instructions.

This is the complete inter-processor communication between 8086 and 8087 to form a homogeneous system.



8086-8087 interface

Video Content / Details of website for further learning (if any):
https://www.youtube.com/watch?v=f0uY6Fmo5rU

Important Books/Journals for further learning including the page nos.:
John P. Hayes, “Computer Architecture and Organization”, Tata McGraw-Hill, 3rd edition, 1998, page no. :272 to 275

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

L - 18

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 16ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit II : Arithmetic and Data Path Design **Date of Lecture:**

Topic of Lecture: Modified Booth's algorithm

Introduction:

Modified Booth multiplication algorithm is designed using high speed adder. High speed adder is used to speed up the operation of **multiplication**.

Prerequisite knowledge for Complete understanding and learning of Topic:
Digital Electronics, Microprocessors and Microcontrollers

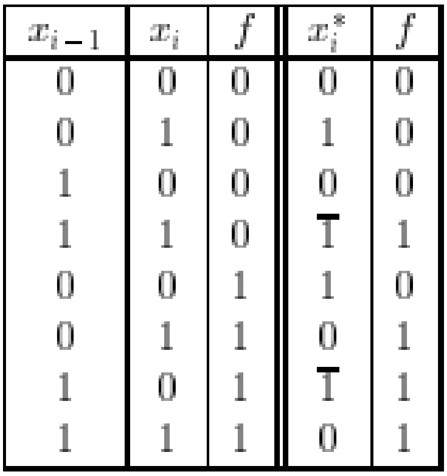
The basic booth's algorithm can be improved by detecting an isolated 1 in the multiplier and just performing addition at the corresponding point in the multiplication. Similarly, an isolated 0 corresponds to a subtraction. This is called as modied booth's algorithm, which always requires fewer addition/subtractions as compared to other multiplication algorithms.

Note that the basic booth's algorithm can be implemented by examining two adjacent bits xixi+1 of the multiplier. The modied booth's algorithm can be implemented by identifying isolated 1s and 0s. This is achieved by using a mode ip- op F , which is set to 1 when a run of two or more 1s is encountered, and is reset to 0 when the run of 1s end with two or more 0s.

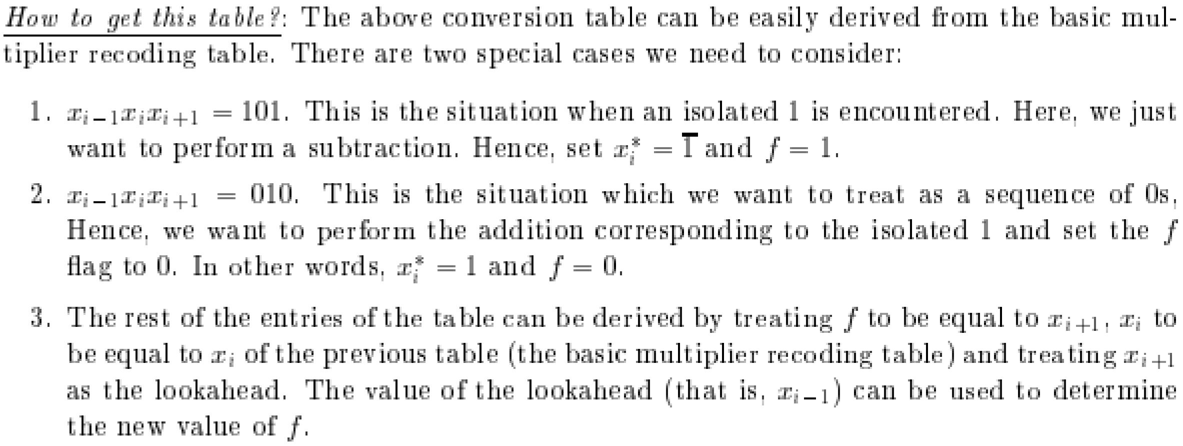
Analogous to the basic booth's algorithm recoding technique, the multiplier recoding scheme that takes isolated 0s and 1s into account is called a canonical signed digit recoding. The basic steps for canonical recoding are as follows:

First x1 = x0 is appended to the left end and a 0 is appended to the right end of the number x0x1x2:::xn1 to create X = x1x0x1:::xn10.

X is the scanned from right to left and the pair of bits xi1xi are used to determine bit x i of the number X using the following algorithm.



x_{i-1}	x_i	f	x_i^*	f
0	0	0	0	0
0	1	0	1	0
1	0	0	0	0
1	1	0	$\bar{1}$	1
0	0	1	1	0
0	1	1	0	1
1	0	1	$\bar{1}$	1
1	1	1	0	1



How to get this table?: The above conversion table can be easily derived from the basic multiplier recoding table. There are two special cases we need to consider:

1. $x_{i-1}x_ix_{i+1} = 101$. This is the situation when an isolated 1 is encountered. Here, we just want to perform a subtraction. Hence, set $x_i^* = \bar{1}$ and $f = 1$.
2. $x_{i-1}x_ix_{i+1} = 010$. This is the situation which we want to treat as a sequence of 0s, Hence, we want to perform the addition corresponding to the isolated 1 and set the f flag to 0. In other words, $x_i^* = 1$ and $f = 0$.
3. The rest of the entries of the table can be derived by treating f to be equal to x_{i+1}, x_i to be equal to x_i of the previous table (the basic multiplier recoding table) and treating x_{i+1} as the lookahead. The value of the lookahead (that is, x_{i-1}) can be used to determine the new value of f.

Video Content / Details of website for further learning (if any):
https://www.youtube.com/watch?v=db9-g_nnZSE

Important Books/Journals for further learning including the page nos.:
John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no. :446 to 390

Course Faculty

Verified by HOD

L - 19

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit III : Control Unit Design **Date of Lecture:**

Topic of Lecture: Introduction to control design, Hardwired control

Introduction:

Control unit generates timing and control signals for the operations of the computer. The control unit communicates with ALU and main memory. It also controls the transmission between processor, memory and the various peripherals. It also instructs the ALU which operation has to be performed on data.

Prerequisite knowledge for Complete understanding and learning of Topic:

Microprocessors and Microcontrollers

Design of Control Unit:

Control unit generates timing and control signals for the operations of the computer. The control unit communicates with ALU and main memory. It also controls the transmission between processor, memory and the various peripherals. It also instructs the ALU which operation has to be performed on data.

Control unit can be designed by two methods: Hardwired control and micro-programmed control.

Hardwired Control Unit:

The control hardware can be viewed as a state machine that changes from one state to another in every clock cycle, depending on the contents of the instruction register, the condition codes and the external inputs. The outputs of the state machine are the control signals. The sequence of the operation carried out by this machine is determined by the wiring of the logic elements and hence named as “hardwired”.

- Fixed logic circuits that correspond directly to the Boolean expressions are used to generate the control signals.
- Hardwired control is faster than micro-programmed control.
- A controller that uses this approach can operate at high speed.
- RISC architecture is based on hardwired control unit

Advantages of Hardwired Control unit:

1. It is faster than the microprogrammed control unit.
2. It can be optimized to produce the fast mode of operation.

Disadvantages of Hardwired control unit:

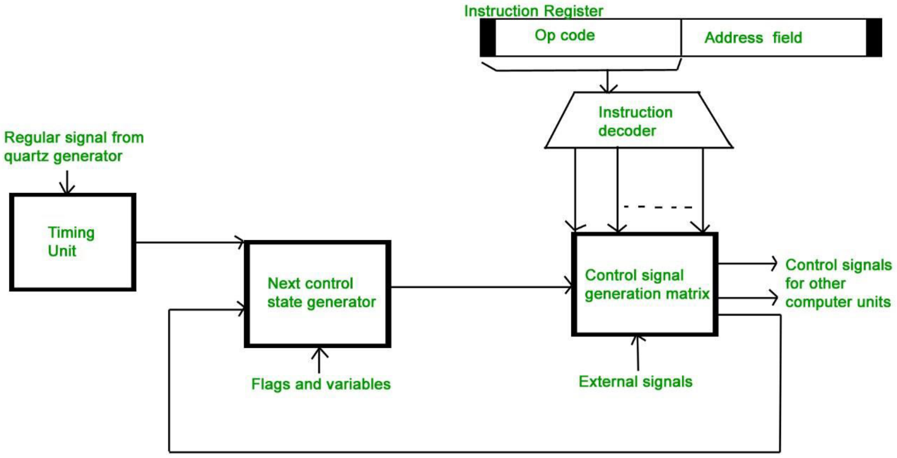
Instruction set, the **control** logic is directly implemented.

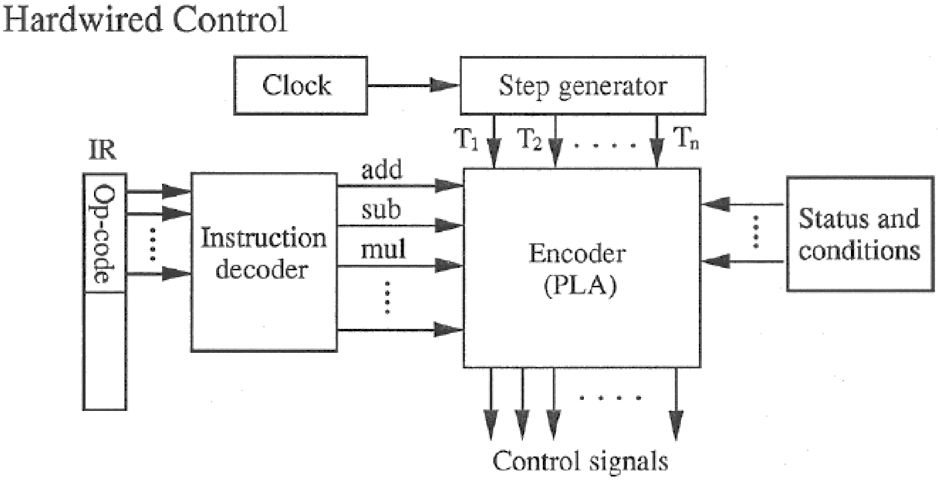
Requires change in wiring if the design has to be controlled.

An occurrence of an error is more.

Complex decoding and sequencing logic.

It requires a more chip area, therefore, it is a costlier **control unit**.


Instruction Register
Op code
Address field
Instruction decoder
Regular signal from quartz generator
Timing Unit
Next control state generator
Control signal generation matrix
Control signals for other computer units
Flags and variables
External signals



Video Content / Details of website for further learning (if any):
https://www.youtube.com/watch?v=1q2JKX3qg-4

Important Books/Journals for further learning including the page nos.:
John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no.:306 to 331

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

DESIGNING YOUR FUTURE
Estd. 2000

IQAC

L - 20

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit III : Control Unit Design

Date of Lecture:

Topic of Lecture: Control transfer, Micro programmed control

Introduction:

A sequence of micro operations is carried out by executing a program consisting of micro-instructions. In this organization any modifications or changes can be done by updating the micro program in the control memory by the programmer.

Prerequisite knowledge for Complete understanding and learning of Topic:

Microprocessors and Microcontrollers

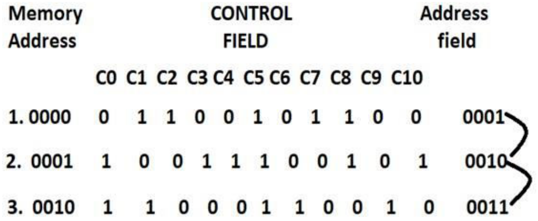
Control Transfer, Micro programmed control:

In the microprogrammed control unit, microinstructions are stored in a memory called control memory. In a response to a machine instruction, a set of a microinstruction is executed by which each microinstruction will generate a set of the control signals i.e. execution of a set of microinstructions will resemble the execution of a program.

A micro instruction consists of:

1. One or more microinstruction to be executed
2. Address of the next microinstruction to be executed

A microinstruction can cause the execution of one or more micro-operation and sequence of the microinstruction can cause execution of the instruction. The microprogram consists of the microinstruction which is generally nothing but a string of 0 and 1. In this if the content of the memory cell is 0, it will indicate, that the signal is not generated and is the content of the memory cell is 1 then it will indicate to generate the control signal at a time.



Memory Address	CONTROL FIELD											Address field
	C0	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	
1. 0000	0	1	1	0	0	1	0	1	1	0	0	0001
2. 0001	1	0	0	1	1	1	0	0	1	0	1	0010
3. 0010	1	1	0	0	0	1	1	0	0	1	0	0011

MICRO-PROGRAM

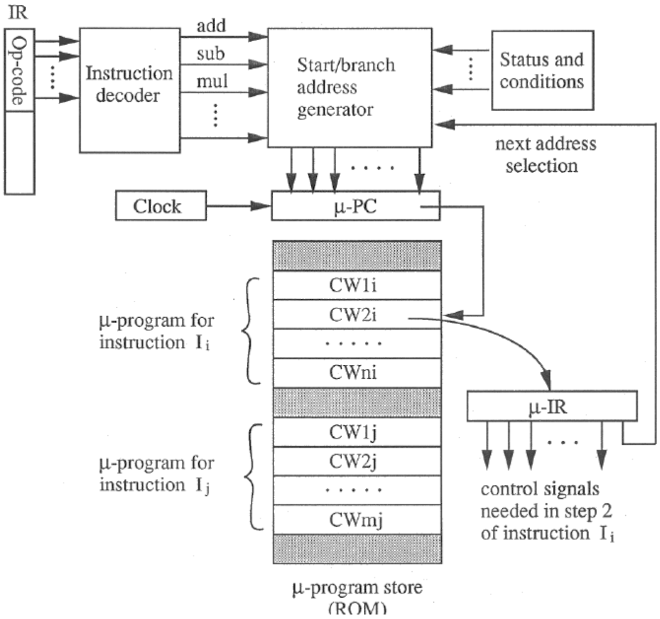
On execution of microinstructions at memory address 0000, C1, C2, C5, C7, C8 will be generated. Address of the next instruction is provided by the address field.

Functioning of micro programmed control unit:

1. The control unit can generate the control signal for any instruction by sequentially reading the control word of the corresponding microprogram from the memory.
2. To read the control word sequentially from the microprogram memory, a microprogram counter is needed.

3. The starting address block is responsible for the loading. The starting address of the microprogram into the PC provides a new instruction which is loaded into IR.
4. The PC is then automatically incremented by the clock, and it reads the successive microinstruction from the memory.
5. By this, each microinstruction provides the control signal and the microprogram counters ensure that the control signal will be delivered to the various parts of the CPU in the correct sequence.

μ-Programmed Control



Advantages of Micro Programmed Control unit:

1. It is both cheaper and the occurrence of an error is less.
2. More flexible to accommodate with new instructions.
3. Easier decoding and sequencing can be done.
4. Easier to handle complex instruction sets.
5. It requires a less chip area.

Disadvantages of Micro programmed control unit:

1. This is slower than the hardwired control unit because the microinstructions are to be fetched from the control memory which is time-consuming.

Video Content / Details of website for further learning (if any):
https://www.youtube.com/watch?v=1q2JKX3qg-4

Important Books/Journals for further learning including the page nos.:
John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no. :308 to 331

Course Faculty

Verified by HOD

L - 21

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit III : Control Unit Design **Date of Lecture:**

Topic of Lecture: Micro programmed control

Introduction:

A sequence of micro operations is carried out by executing a program consisting of micro-instructions. In this organization any modifications or changes can be done by updating the micro program in the control memory by the programmer.

Prerequisite knowledge for Complete understanding and learning of Topic:

Microprocessors and Microcontrollers

Micro-programmed Control Unit –

1. The control signals associated with operations are stored in special memory units inaccessible by the programmer as Control Words.
2. Control signals are generated by a program are similar to machine language programs.
3. Micro-programmed control unit is slower in speed because of the time it takes to fetch microinstructions from the control memory.

Some Important Terms –

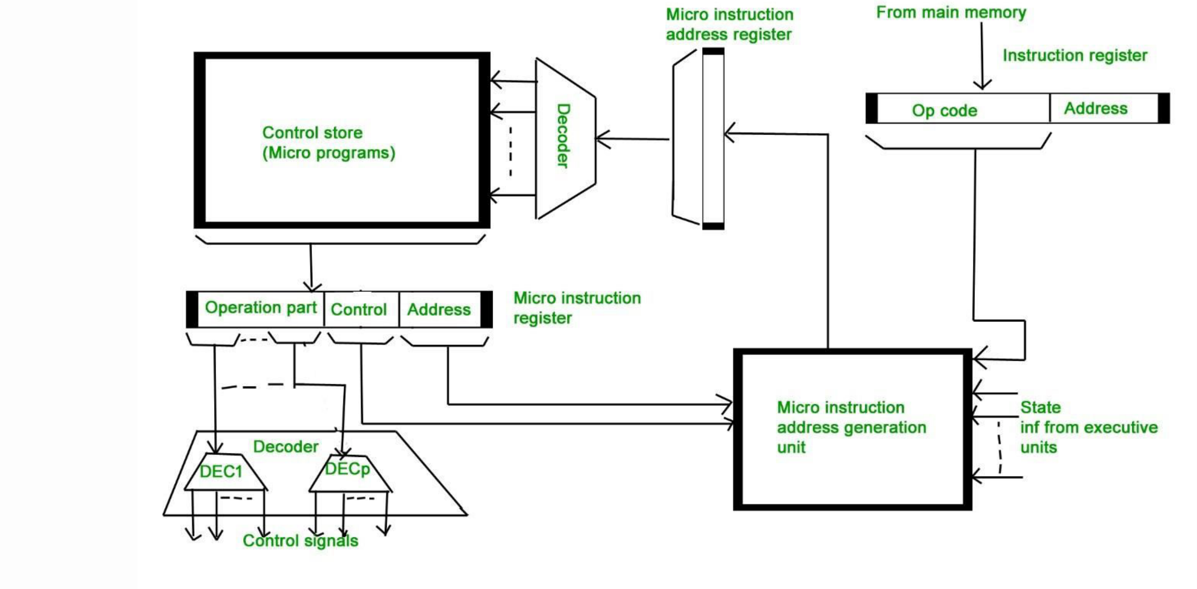
4. **Control Word :** A control word is a word whose individual bits represent various control signals.
5. **Micro-routine :** A sequence of control words corresponding to the control sequence of a machine instruction constitutes the micro-routine for that instruction.
6. **Micro-instruction :** Individual control words in this micro-routine are referred to as microinstructions.
7. **Micro-program :** A sequence of micro-instructions is called a micro-program, which is stored in a ROM or RAM called a Control Memory (CM).
8. **Control Store :** the micro-routines for all instructions in the instruction set of a computer are stored in a special memory called the Control Store.

Types of Micro-programmed Control Unit – Based on the type of Control Word stored in the Control Memory (CM), it is classified into two types :

1. Horizontal Micro-programmed control Unit :

The control signals are represented in the decoded binary format that is 1 bit/CS. Example: If 53 Control signals are present in the processor than 53 bits are required. More than 1 control signal can be enabled at a time.

1. It supports longer control word.
2. It is used in parallel processing applications.
3. It allows higher degree of parallelism. If degree is n, n CS are enabled at a time.
4. It requires no additional hardware(decoders). It means it is faster than Vertical Microprogrammed.
5. It is more flexible than vertical microprogrammed


Micro instruction address register
From main memory
Instruction register
Op code
Address
Control store (Micro programs)
Decoder
Operation part
Control
Address
Micro instruction register
Micro instruction address generation unit
State inf from executive units
Decoder
DEC1
DECp
Control signals

2. Vertical Micro-programmed control Unit :

The control signals re represented in the encoded binary format. For N control signals- Log2(N) bits are required.

6. It supports shorter control words.
7. It supports easy implementation of new conrol signals therefore it is more flexible.
8. It allows low degree of parallelism i.e., degree of parallelism is either 0 or 1.
9. Requires an additional hardware (decoders) to generate control signals, it implies it is slower than horizontal microprogrammed.

It is less flexible than horizontal but more flexible than that of hardwired control unit.

Difference between Hardwired Control and Microprogrammed Control

Hardwired Control	**Microprogrammed Control**
Technology is circuit based.	Technology is software based.
It is implemented through flip-flops, gates, decoders etc.	Microinstructions generate signals to control the execution of instructions.
Fixed instruction format.	Variable instruction format (16-64 bits per instruction).
Instructions are register based.	Instructions are not register based.
ROM is not used.	ROM is used.
It is used in RISC.	It is used in CISC.
Faster decoding.	Slower decoding.
Difficult to modify.	Easily modified.
Chip area is less.	Chip area is large.

Video Content / Details of website for further learning (if any):

https://www.youtube.com/watch?v=nj2u4rtnDMM

Important Books/Journals for further learning including the page nos.:

John P. Hayes, “Computer Architecture and Organization”, Tata McGraw-Hill, 3rd edition, 1998, page no. :332 to 344

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

Estd. 2000

L - 22

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit III : Control Unit Design

Date of Lecture:

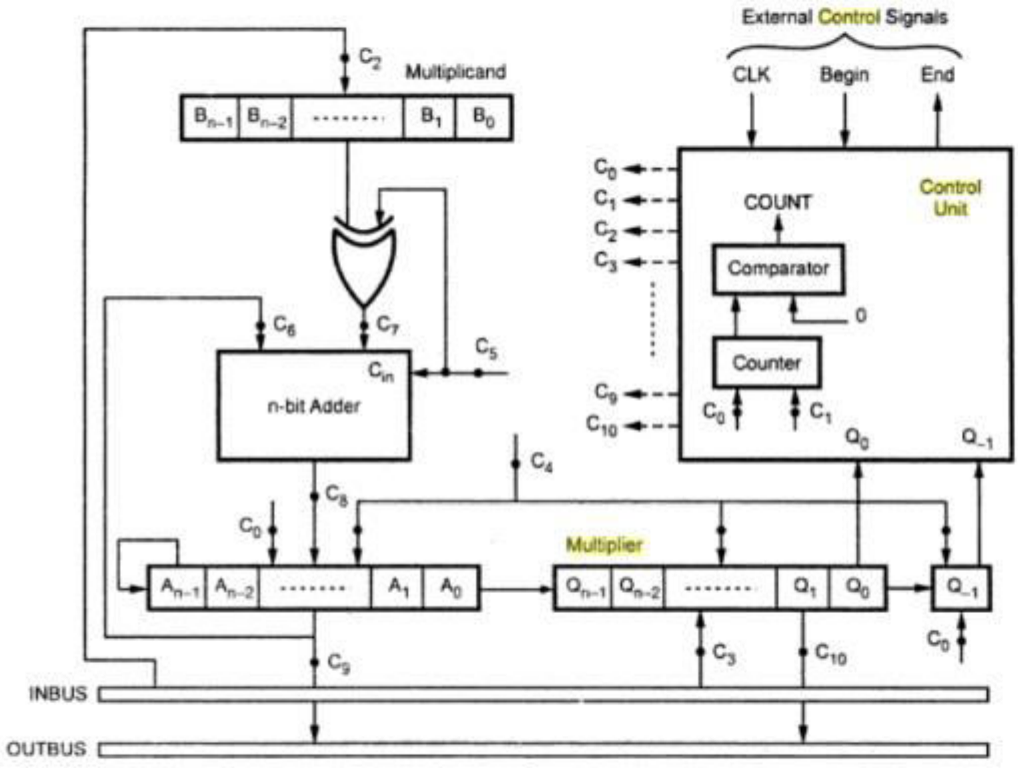
Topic of Lecture: Multiplier control unit

Introduction:

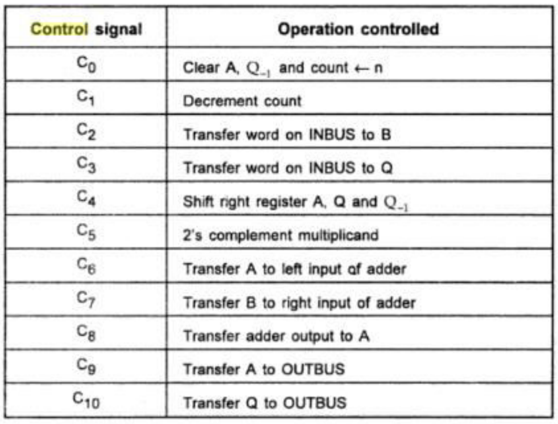
The **control unit** (CU) is a component of a computer's central processing **unit** (CPU) that directs the operation of the processor. It tells the computer's memory, arithmetic and logic **unit** and input and output devices how to respond to the instructions that have been sent to the processor.

Prerequisite knowledge for Complete understanding and learning of Topic:

Microprocessors and Microcontrollers

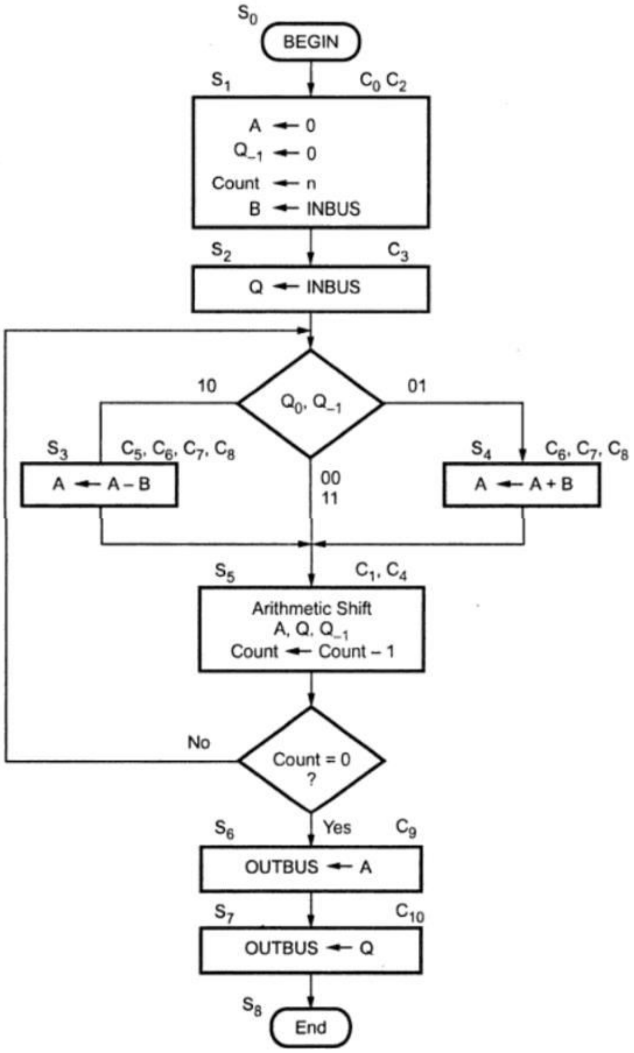


Multiplier control circuit with control signals



Control signal	Operation controlled
C_0	Clear A, Q_{-1} and count ← n
C_1	Decrement count
C_2	Transfer word on INBUS to B
C_3	Transfer word on INBUS to Q
C_4	Shift right register A, Q and Q_{-1}
C_5	2's complement multiplicand
C_6	Transfer A to left input of adder
C_7	Transfer B to right input of adder
C_8	Transfer adder output to A
C_9	Transfer A to OUTBUS
C_{10}	Transfer Q to OUTBUS

Control signals for the two's complement multiplier



Video Content / Details of website for further learning (if any):
https://www.youtube.com/watch?v=nj2u4rtnDMM

Important Books/Journals for further learning including the page nos.:
John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no. :344 to 371

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

Estd. 2000

L - 23

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit III : Control Unit Design **Date of Lecture:**

Topic of Lecture: CPU control unit

Introduction:

The **control unit** (CU) is a component of a computer's central processing **unit** (CPU) that directs the operation of the processor. It tells the computer's memory, arithmetic and logic **unit** and input and output devices how to respond to the instructions that have been sent to the processor.

Prerequisite knowledge for Complete understanding and learning of Topic:

Microprocessors and Microcontrollers

CPU control unit:

The operations of Input unit, output unit, ALU are co-ordinate by the control unit.

The control unit is the Nerve centre that sends control signals to other units and senses their states.

Data transfers between the processor and the memory are also controlled by the control unit through timing signals.

The operation of computers are,

- The computer accepts information in the form of programs and data through an input unit and stores it in the memory.

Information stored in the memory is fetched, under program control into an arithmetic and logic unit, where it is processed.

- Processed information leaves the computer through an output unit.
- All activities inside the machine are directed by the control unit.

BASIC OPERATIONAL CONCEPTS:

The data/operands are stored in memory.

The individual instruction are brought from the memory to the processor, which executes the specified operation.

Add LOC A ,R1

Instructions are fetched from memory and the operand at LOC A is fetched. It is then added to the contents of R0, the resulting sum is stored in Register R0.

Load LOC A, R1

Transfer the contents of memory location A to the register R1.

Add R1 ,R0

Add the contents of Register R1 & R0 and places the sum into R0.

Instruction Register(IR)

Program Counter(PC)

Memory Address Register(MAR)

Memory Data Register(MDR)

Instruction Register (IR):

- It holds the instruction that is currently being executed.
- It generates the timing signals.

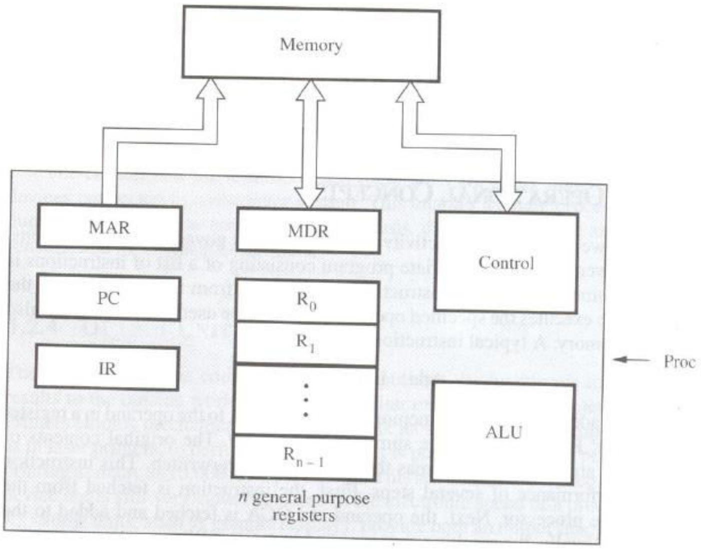


Fig:Connection between Processor and Main Memory

Program Counter (PC):

It contains the memory address of the next instruction to be fetched for execution.

Memory Address Register (MAR):

It holds the address of the location to be accessed.

Memory Data Register (MDR):

- It contains the data to written into or read out of the address location.
- MAR and MDR facilitates the communication with memory.

Operation Steps:

- The program resides in memory. The execution starts when PC is point to the first instruction of the program.
- MAR read the control signal.
- The Memory loads the address word into MDR. The contents are transferred to Instruction register. The instruction is ready to be decoded & executed.

Video Content / Details of website for further learning (if any):

https://www.youtube.com/watch?v=oO0dDHh7efQ

Important Books/Journals for further learning including the page nos.:

John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no. :354 to 364

Course Faculty

Verified by HOD

L - 24

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit III : Control Unit Design **Date of Lecture:**

Topic of Lecture: Pipeline Control

Introduction:

Pipelining is a process of arrangement of hardware elements of the CPU such that its overall performance is increased. Simultaneous execution of more than one instruction takes place in a pipelined processor.

Prerequisite knowledge for Complete understanding and learning of Topic:

Microprocessors and Microcontrollers

Pipeline control:

Pipelining is a process of arrangement of hardware elements of the CPU such that its overall performance is increased. Simultaneous execution of more than one instruction takes place in a pipelined processor.
pipelined operation increases the efficiency of a system.

Design of a basic pipeline

- In a pipelined processor, a pipeline has two ends, the input end and the output end. Between these ends, there are multiple stages/segments such that output of one stage is connected to input of next stage and each stage performs a specific operation.
- Interface registers are used to hold the intermediate output between two stages. These interface registers are also called latch or buffer.
- All the stages in the pipeline along with the interface registers are controlled by a common clock.

Execution in a pipelined processor

Execution sequence of instructions in a pipelined processor can be visualized using a space-time diagram. For example, consider a processor having 4 stages and let there be 2 instructions to be executed. We can visualize the execution sequence through the following space-time diagrams:

Pipeline Stages

RISC processor has 5 stage instruction pipeline to execute all the instructions in the RISC instruction set. Following are the 5 stages of RISC pipeline with their respective operations:

- **Stage 1 (Instruction Fetch)**
  In this stage the CPU reads instructions from the address in the memory whose value is present in the program counter.
- **Stage 2 (Instruction Decode)**
  In this stage, instruction is decoded and the register file is accessed to get the values from the registers used in the instruction.
- **Stage 3 (Instruction Execute)**
  In this stage, ALU operations are performed.
- **Stage 4 (Memory Access)**

In this stage, memory operands are read and written from/to the memory that is present in the instruction.

- **Stage 5 (Write Back)**

  In this stage, computed/fetched value is written back to the register present in the instructions.

Performance of a pipelined processor

Consider a 'k' segment pipeline with clock cycle time as 'Tp'. Let there be 'n' tasks to be completed in the pipelined processor. Now, the first instruction is going to take 'k' cycles to come out of the pipeline but the other 'n – 1' instructions will take only '1' cycle each, i.e, a total of 'n – 1' cycles. So, time taken to execute 'n' instructions in a pipelined processor:

$ET_{pipeline} = k + n - 1$ cycles

$= (k + n - 1)\ Tp$

In the same case, for a non-pipelined processor, execution time of 'n' instructions will be:

$ET_{non\text{-}pipeline} = n * k * Tp$

So, speedup (S) of the pipelined processor over non-pipelined processor, when 'n' tasks are executed on the same processor is:

S = Performance of pipelined processor /

Performance of Non-pipelined processor

As the performance of a processor is inversely proportional to the execution time, we have,

$S = ET_{non\text{-}pipeline} / ET_{pipeline}$

$\Rightarrow S = [n * k * Tp] / [(k + n - 1) * Tp]$

$S = [n * k] / [k + n - 1]$

When the number of tasks 'n' are significantly larger than k, that is, n >> k

$S = n * k / n$

$S = k$

where 'k' are the number of stages in the pipeline.

Also, **Efficiency** = Given speed up / Max speed up = S / S_{max}

We know that, Smax = k

So, **Efficiency** = S / k

Throughput = Number of instructions / Total time to complete the instructions

So, **Throughput** = n / (k + n – 1) * Tp

Video Content / Details of website for further learning (if any):

https://www.youtube.com/watch?v=q4fwx3h3mdg

Important Books/Journals for further learning including the page nos.:

John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no. :364 to 371

Course Faculty

Verified by HOD

LECTURE HANDOUTS

L - 25

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit III : Control Unit Design

Date of Lecture:

Topic of Lecture: Instruction Pipelines

Introduction:

Pipelining is a process of arrangement of hardware elements of the CPU such that its overall performance is increased. Simultaneous execution of more than one instruction takes place in a pipelined processor.

Prerequisite knowledge for Complete understanding and learning of Topic:

Microprocessors and Microcontrollers

Instruction Pipeline

Pipeline processing can occur not only in the data stream but in the instruction stream as well.

Most of the digital computers with complex instructions require instruction pipeline to carry out operations like fetch, decode and execute instructions.

In general, the computer needs to process each instruction with the following sequence of steps.

1. Fetch instruction from memory.
2. Decode the instruction.
3. Calculate the effective address.
4. Fetch the operands from memory.
5. Execute the instruction.
6. Store the result in the proper place.

Each step is executed in a particular segment, and there are times when different segments may take different times to operate on the incoming information. Moreover, there are times when two or more segments may require memory access at the same time, causing one segment to wait until another is finished with the memory.

The organization of an instruction pipeline will be more efficient if the instruction cycle is divided into segments of equal duration. One of the most common examples of this type of organization is a **Four-segment instruction pipeline.**

A **four-segment instruction** pipeline combines two or more different segments and makes it as a single one. For instance, the decoding of the instruction can be combined with the calculation of the effective address into one segment.

The instruction cycle is completed in four segments.

Segment 1: **The instruction fetch segment can be implemented using first in, first out (FIFO) buffer.**

Segment 2:

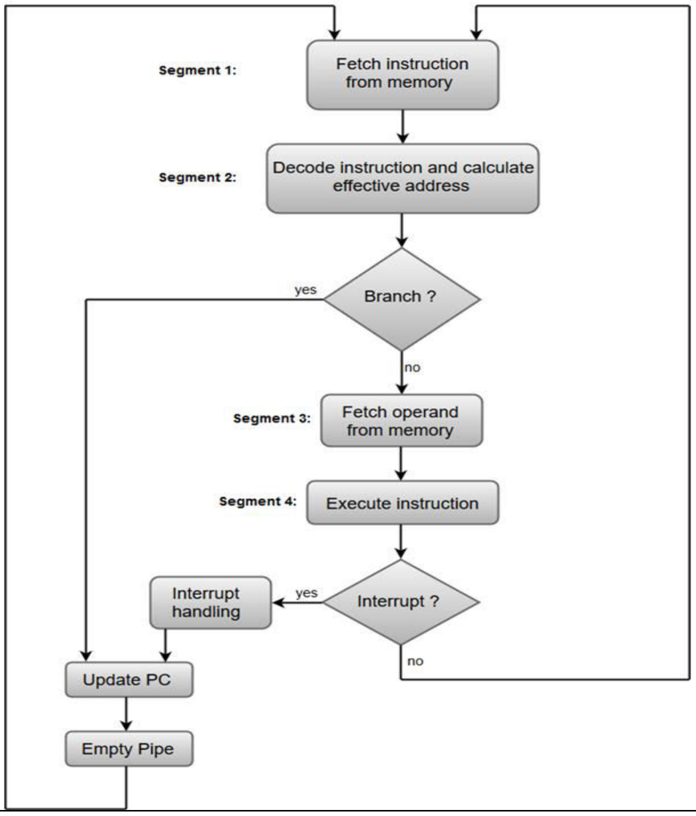
The instruction fetched from memory is decoded in the second segment, and eventually, the effective address is calculated in a separate arithmetic circuit.

Segment 3:

An operand from memory is fetched in the third segment.

Segment 4:

The instructions are finally executed in the last segment of the pipeline organization.



Video Content / Details of website for further learning (if any):

https://www.youtube.com/watch?v=YhGv5AOcz1s

Important Books/Journals for further learning including the page nos.:

John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no. :364 to 371

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

Estd. 2000

L - 26

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit III : Control Unit Design **Date of Lecture:**

Topic of Lecture: Pipeline Performance

Introduction:

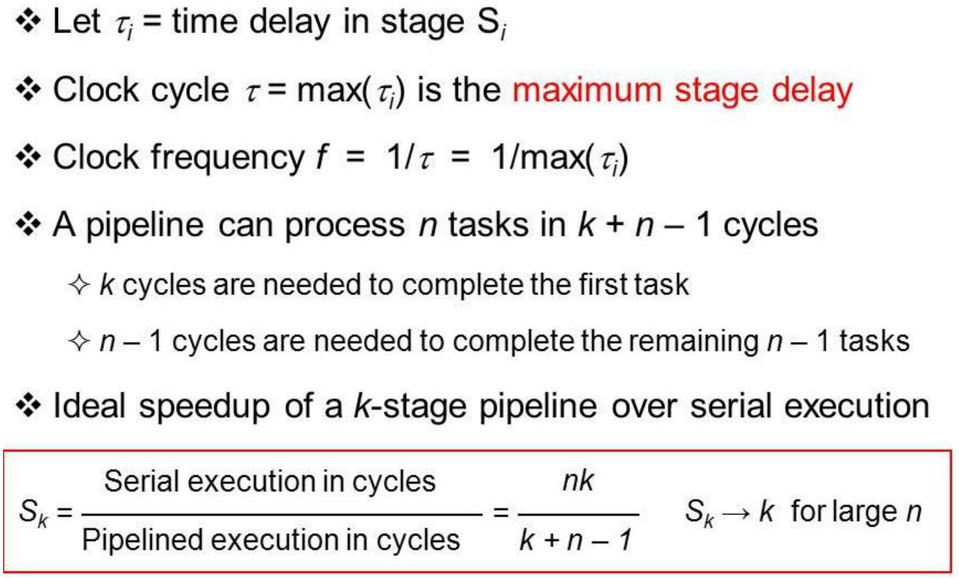
Pipelining is the process of accumulating instruction from the processor through a pipeline. It allows storing and executing instructions in an orderly process. It is also known as **pipeline processing**.

Prerequisite knowledge for Complete understanding and learning of Topic:

Microprocessors and Microcontrollers

Pipeline performance:

The ability to overlap stages of a sequential process for different input tasks (data or operations) results in an overall theoretical completion time of Tpipe = m*P + (n-1)*P,
where n is the number of input tasks, m is the number of stages in the pipeline, and P is the clock period. The term m*P is the time required for the first input task to get through the pipeline, and the term (n-1)*P is the time required for the remaining tasks.
For the ideal case when all stages have equal delay τi = τ for i = 1 to m, Tseq can be rewritten as Tseq = n*m* τ. If we ignore the small storing time tl that is required for latch storage (i.e., t1 = 0), then Tseq = n * m * P. (3.2) Now, speedup (S) may be represented as: S = Tseq / Tpipe = n*m / (m+n -1). The value S approaches m when n → ∞ . That is, the maximum speedup, also called ideal speedup, of a pipeline processor with m stages over an equivalent nonpipelined processor is m. In other words, the ideal speedup is equal to the number of pipeline stages. That is, when n is very large, a pipelined processor can produce output approximately m times faster than a nonpipelined processor. When n is small, the speedup decreases; in fact, for n=1 the pipeline has the minimum speedup of 1.



- Let τ_i = time delay in stage S_i
- Clock cycle $\tau = \max(\tau_i)$ is the maximum stage delay
- Clock frequency $f = 1/\tau = 1/\max(\tau_i)$
- A pipeline can process n tasks in $k + n - 1$ cycles
  - k cycles are needed to complete the first task
  - $n - 1$ cycles are needed to complete the remaining $n - 1$ tasks
- Ideal speedup of a k-stage pipeline over serial execution

$$S_k = \frac{\text{Serial execution in cycles}}{\text{Pipelined execution in cycles}} = \frac{nk}{k+n-1} \qquad S_k \rightarrow k \text{ for large } n$$

The ideal speedup is equal to the number of pipeline stages. In addition to speedup, two other factors are often used for determining the performance of a pipeline; they are efficiency and throughput.

The efficiency E of a pipeline with m stages is defined as: $E = S/m = [n*m / (m+n -1)] / m = n / (m+n - 1)$. The efficiency E, which represents the speedup per stage, approaches its maximum value of 1 when $n \to \infty$. When n=1, E will have the value 1/m, which is the lowest obtainable value. The throughput H, also called bandwidth, of a pipeline is defined as the number of input tasks it can process per unit of time. When the pipeline has m stages, H is defined as $H = n / Tpipe = n / [m*P + (n-1)*P] = E / P = S / (mP)$. When $n \to \infty$, the throughput H approaches the maximum value of one task per clock cycle. The number of stages in a pipeline often depends on the tradeoff between performance and cost. The optimal choice for such a number can be determined by obtaining the peak value of a performance/cost ratio (PCR) is the ratio of maximum throughput and pipeline cost.

Video Content / Details of website for further learning (if any):

https://www.youtube.com/watch?v=apz1qL7jDeQ

Important Books/Journals for further learning including the page nos.:

John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no. :371 to 383

Course Faculty

Verified by HOD

L - 27

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit III : Control Unit Design **Date of Lecture:**

Topic of Lecture: Superscalar Processing

Introduction:

A **superscalar processor** is a CPU that implements a form of parallelism called instruction-level parallelism within a single **processor**.

Prerequisite knowledge for Complete understanding and learning of Topic:

Microprocessors and Microcontrollers

Super scalar processing:

Superscalar architecture is a method of parallel computing used in many processors. In a superscalar computer, the central processing unit (CPU) manages multiple instruction pipelines to execute several instructions concurrently during a clock cycle. This is achieved by feeding the different pipelines through a number of execution units within the processor. To successfully implement a superscalar architecture, the CPU's instruction fetching mechanism must intelligently retrieve and delegate instructions. Otherwise, pipeline stalls may occur, resulting in execution units that are often idle.

It was first invented in 1987. It is a machine which is designed to improve the performance of the scalar processor. In most applications, most of the operations are on scalar quantities. Superscalar approach produces the high performance general purpose processors.

The main principle of superscalar approach is that it executes instructions independently in different pipelines. As we already know, that Instruction pipelining leads to parallel processing thereby speeding up the processing of instructions. In Superscalar processor, multiple such pipelines are introduced for different operations, which further improves parallel processing.

There are multiple functional units each of which is implemented as a pipeline. Each pipeline consists of multiple stages to handle multiple instructions at a time which support parallel execution of instructions.

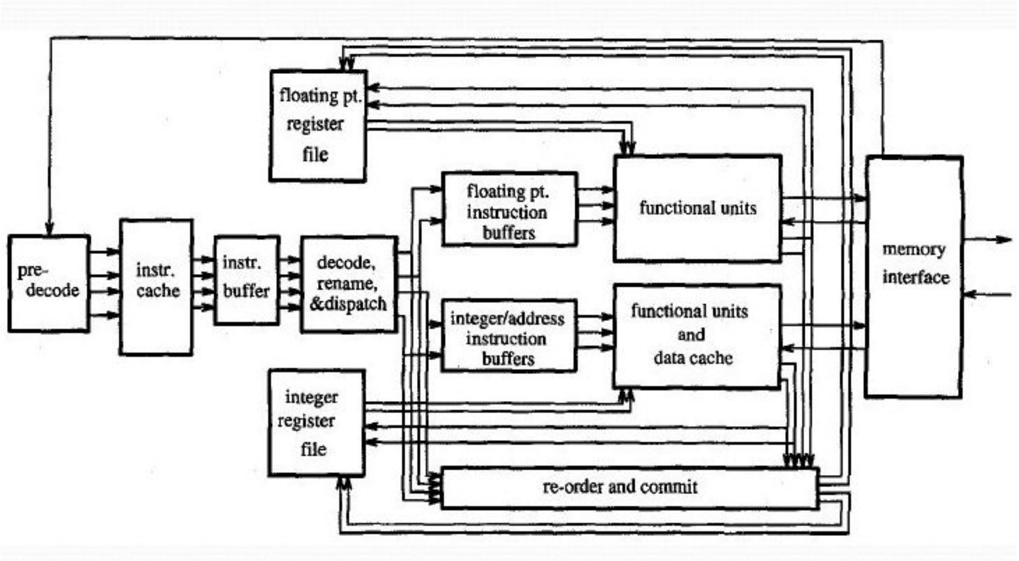
It increases the throughput because the CPU can execute multiple instructions per clock cycle. Thus, superscalar processors are much faster than scalar processors.

A **scalar processor** works on one or two data items, while the **vector processor** works with multiple data items. A **superscalar processor** is a combination of both. Each instruction processes one data item, but there are multiple execution units within each CPU thus multiple instructions can be processing separate data items concurrently.

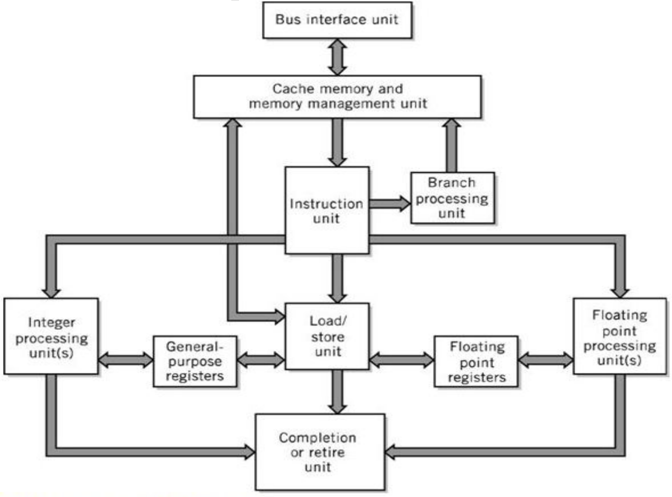
While a superscalar CPU is also pipelined, there are two different performance enhancement techniques. It is possible to have a non-pipelined superscalar CPU or pipelined non-superscalar CPU.

The superscalar technique is associated with some characteristics, these are:

1. Instructions are issued from a sequential instruction stream.
2. CPU must dynamically check for data dependencies.
3. Should accept multiple instructions per clock cycle.


floating pt. register file
pre-decode
instr. cache
instr. buffer
decode, rename, &dispatch
floating pt. instruction buffers
functional units
integer/address instruction buffers
functional units and data cache
memory interface
integer register file
re-order and commit

Super scalar architecture


Bus interface unit
Cache memory and memory management unit
Instruction unit
Branch processing unit
Integer processing unit(s)
General-purpose registers
Load/ store unit
Floating point registers
Floating point processing unit(s)
Completion or retire unit

Superscalar CPU Block diagram

Video Content / Details of website for further learning (if any):
https://www.youtube.com/watch?v=ZUhJu84LMQo

Important Books/Journals for further learning including the page nos.:
John P. Hayes, “Computer Architecture and Organization”, Tata McGraw-Hill, 3rd edition, 1998, page no.:384 to 390

Course Faculty

Verified by HOD

L - 28

LECTURE HANDOUTS

ECE **III/V**

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit IV : Memory Organization **Date of Lecture:**

Topic of Lecture: Random access memory, SRAM, DRAM

Introduction:

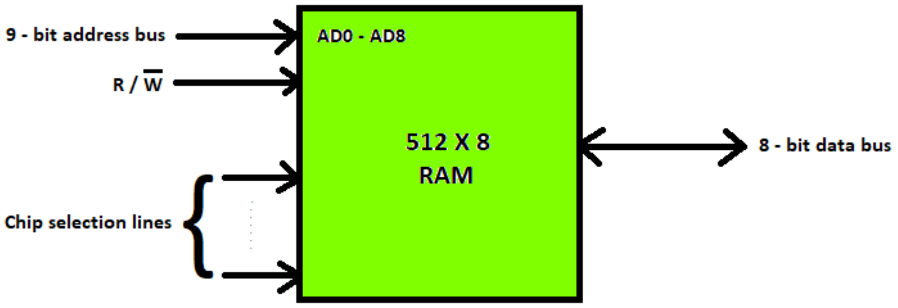
RAM (Random Access Memory) is the internal memory of the CPU for storing data, program, and program result. It is a read/write memory which stores data until the machine is working. As soon as the machine is switched off, data is erased.

Prerequisite knowledge for Complete understanding and learning of Topic:

Microprocessors and Microcontrollers

Random access memory, SRAM, DRAM:

RAM (Random Access Memory) is the internal memory of the CPU for storing data, program, and program result. It is a read/write memory which stores data until the machine is working. As soon as the machine is switched off, data is erased.



Access time in RAM is independent of the address, that is, each storage location inside the memory is as easy to reach as other locations and takes the same amount of time. Data in the RAM can be accessed randomly but it is very expensive.

RAM is volatile, i.e. data stored in it is lost when we switch off the computer or if there is a power failure. Hence, a backup Uninterruptible Power System (UPS) is often used with computers. RAM is small, both in terms of its physical size and in the amount of data it can hold.

RAM is of two types −

- Static RAM (SRAM)
- Dynamic RAM (DRAM)

Static RAM (SRAM)

The word **static** indicates that the memory retains its contents as long as power is being supplied. However, data is lost when the power gets down due to volatile nature. SRAM chips use a matrix of 6-transistors and no capacitors. Transistors do not require power to prevent leakage, so SRAM need not be refreshed on a regular basis.

There is extra space in the matrix, hence SRAM uses more chips than DRAM for the same amount of storage space, making the manufacturing costs higher. SRAM is thus used as cache memory and has very fast access.

Characteristic of Static RAM

- Long life
- No need to refresh
- Faster
- Used as cache memory
- Large size
- Expensive
- High power consumption

Dynamic RAM (DRAM)

DRAM, unlike SRAM, must be continually **refreshed** in order to maintain the data. This is done by placing the memory on a refresh circuit that rewrites the data several hundred times per second. DRAM is used for most system memory as it is cheap and small. All DRAMs are made up of memory cells, which are composed of one capacitor and one transistor.

Characteristics of Dynamic RAM

- Short data lifetime
- Needs to be refreshed continuously
- Slower as compared to SRAM
- Used as RAM
- Smaller in size
- Less expensive
- Less power consumption

Video Content / Details of website for further learning (if any):
https://www.youtube.com/watch?v=c3V8w2Wk-D0

Important Books/Journals for further learning including the page nos.:
John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no. :407 to 411

Course Faculty

Verified by HOD

L - 29

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit IV : Memory Organization **Date of Lecture:**

Topic of Lecture: Serial access memories

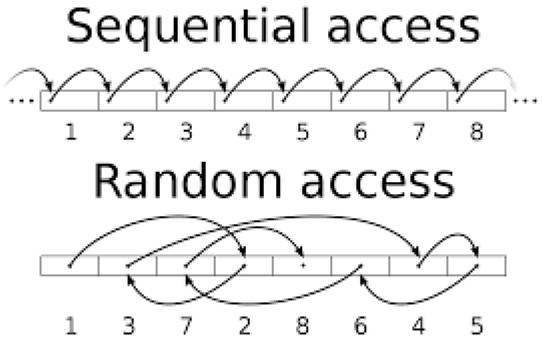
Introduction:

Sequential access means the system must search the storage device from the beginning of the memory address until it finds the required piece of data. Memory device which supports such access is called a Sequential Access Memory or Serial Access Memory. Magnetic tape is an example of serial access memory.

Prerequisite knowledge for Complete understanding and learning of Topic:
Microprocessors and Microcontrollers

Serial access memories:

In computing, **sequential access memory** (**SAM**) is a class of data storage devices that read stored data in a sequence. This is in contrast to random access memory (RAM) where data can be accessed in any order. Sequential access devices are usually a form of magnetic storage or optical storage.



While sequential access memory is read in sequence, arbitrary locations can still be accessed by "seeking" to the requested location. This operation, however, is often relatively inefficient (see seek time, rotational latency).

Magnetic sequential access memory is typically used for secondary storage in general-purpose computers due to their higher density at lower cost compared to RAM, as well as resistance to wear and non-volatility. Magnetic tape is a type of sequential access memory still in use; historically, drum memory has also been used.

In IBM mainframe operating systems, **Basic sequential access method** (**BSAM**) is an access method to read and write datasets sequentially. BSAM is available on OS/360, OS/VS2, MVS, z/OS, and related operating systems.

BSAM is used for devices that are naturally sequential, such as punched card readers, punches, line printers, and magnetic tape. It is also used for data on devices that could also be addressed directly, such as magnetic disks. BSAM offers device independence: to the extent possible, the same API calls are used for different devices.

BSAM allows programs to read and write physical blocks of data, as opposed to the more powerful but less flexible Queued Sequential Access Method (QSAM) which allows programs to access logical records within physical blocks of data. The BSAM user must be aware of the possibility of encountering short (truncated) blocks (blocks within a dataset which are shorter than the BLKSIZE of the dataset), particularly at the end of a dataset, but also in many cases within a dataset. QSAM has none of these limitations.

In IBM mainframe operating systems, **queued sequential access method** (**QSAM**) is an access method to read and write datasets sequentially. QSAM is available on OS/360, OS/VS2, MVS, z/OS, and related operating systems.

QSAM is used both for devices that are naturally sequential, such as punched card readers and punches and line printers, and for data on devices that could also be addressed directly, such as magnetic disks. QSAM offers device independence: to the extent possible, the same API calls are used for different devices.

QSAM is—as its name says—queued, in this specific context meaning buffered with deblocking of reads and blocking of writes. It allows programs to read and write logical records within physical blocks of data, as opposed to the less advanced basic sequential access method (BSAM) which allows programs to access physical blocks of data, but provides no support for accessing logical records within blocks.

Indeed, QSAM manages truncated final blocks and truncated embedded blocks completely transparently to the user.

The QSAM application program interface can be compared with the interface offered by *open*, *read*, *write* and *close* calls (using file handles) in other operating systems such as Unix and Windows.

Video Content / Details of website for further learning (if any):
https://www.youtube.com/watch?v=R8HKOdS8gqc

Important Books/Journals for further learning including the page nos.:
John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no. :420 to 424

Course Faculty

Verified by HOD

L - 30

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit IV : Memory Organization **Date of Lecture:**

Topic of Lecture: RAM interface

Introduction:
Memory interfacing is an essential topic for digital system design. In fact the among silicon area devoted to memory in a typical digital embedded system or a computer system is substantial.

Prerequisite knowledge for Complete understanding and learning of Topic:
Microprocessors and Microcontrollers

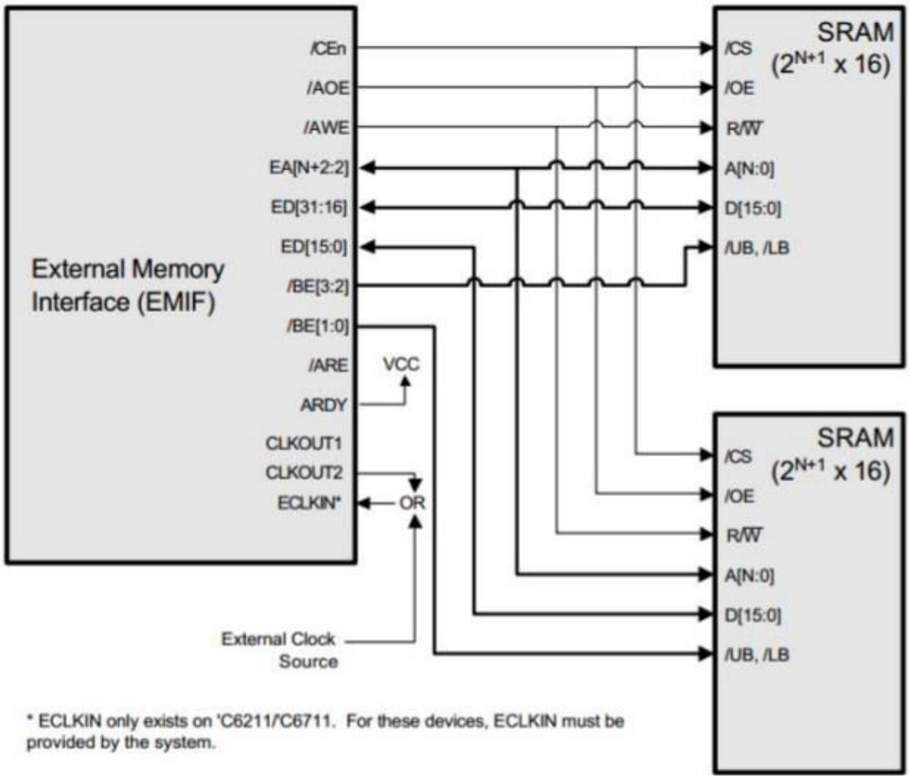
RAM Interfacing:
• The semiconductor RAM is broadly two types – Static RAM and Dynamic RAM.
• The semiconductor memories are organised as two dimensional arrays of memory locations. • For example, 4K * 8 or 4K byte memory contains 4096 locations, where each locations contains 8-bit data and only one of the 4096 locations can be selected at a time. Once a location is selected all the bits in it are accessible using a group of conductors called Data bus.
• For addressing the 4K bytes of memory, 12 address lines are required.
• In general, to address a memory location out of N memory locations, we will require at least n bits of address, i.e. n address lines where n = Log2 N.
• Thus, if the microprocessor has n address lines, then it is able to address at the most N locations of memory, where 2n =N. If out of N locations only P memory locations are to be interfaced, then the least significant p address lines out of the available n lines can be directly connected from the microprocessor to the memory chip while the remaining (n-p) higher order address lines may be used for address decoding as inputs to the chip selection logic.
• The memory address depends upon the hardware circuit used for decoding the chip select (CS). The output of the decoding circuit is connected with the CS pin of the memory chip.
• The general procedure of static memory interfacing with 8086 is briefly described as follows: 1. Arrange the available memory chip so as to obtain 16- bit data bus width. The upper 8-bit bank is called as odd address memory bank and the lower 8-bit bank is called as even address memory bank.
2. Connect available memory address lines of memory chip with those of the microprocessor and also connect the memory RD and WR inputs to the corresponding processor control signals. Connect the 16-bit data bus of the memory bank with that of the microprocessor 8086.
3. The remaining address lines of the microprocessor, BHE and A0 are used for decoding the required chip select signals for the odd and even memory banks. The CS of memory is derived from the o/p of the decoding circuit.

- As a good and efficient interfacing practice, the address map of the system should be continuous as far as possible, i.e. there should not be no windows in the map and no fold back space should be allowed.

A memory location should have a single address corresponding to it, i.e. absolute decoding should be preferred and minimum hardware should be used for decoding.

SRAM Memory Interface

Figure 1. EMIF-SRAM Interface (All C6000 Devices)



* ECLKIN only exists on 'C6211/C6711. For these devices, ECLKIN must be provided by the system.

Video Content / Details of website for further learning (if any):
https://www.youtube.com/watch?v=f6ERN_DsgWY

Important Books/Journals for further learning including the page nos.:
John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no. : 415 to 416

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

Estd. 2000

L - 31

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit IV : Memory Organization

Date of Lecture:

Topic of Lecture: Magnetic surface recording

Introduction:

Magnetic tape recording works by converting electrical audio signals into magnetic energy, which imprints a record of the signal onto a moving tape covered in magnetic particles. Playback is achieved by converting the recording on tape back into electrical energy to be amplified.

Prerequisite knowledge for Complete understanding and learning of Topic:

Microprocessors and Microcontrollers

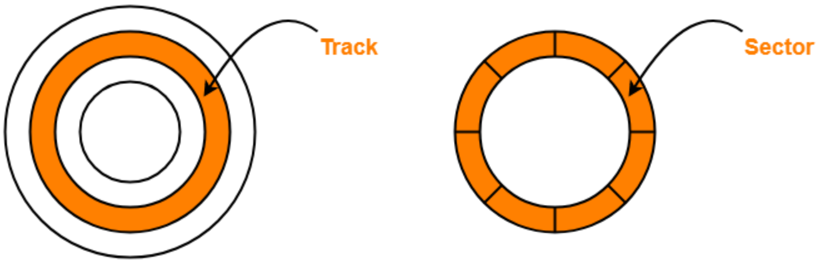
Magnetic surface recording:

In computer architecture,

- Magnetic disk is a storage device that is used to write, rewrite and access data.
- It uses a magnetization process.

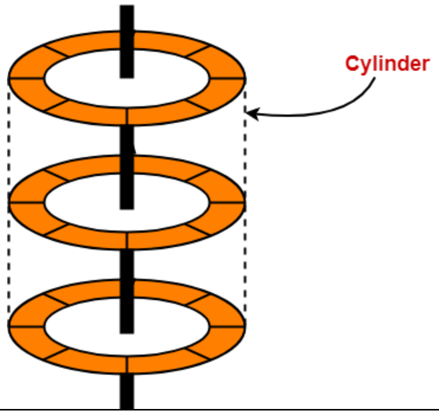
Architecture-

- The entire disk is divided into **platters**.
- Each platter consists of concentric circles called as **tracks**.
- These tracks are further divided into **sectors** which are the smallest divisions in the disk.

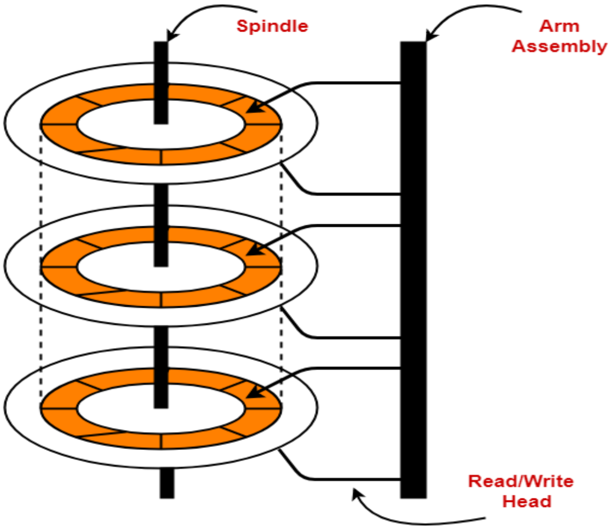


Disk divided into tracks   Track divided into sectors

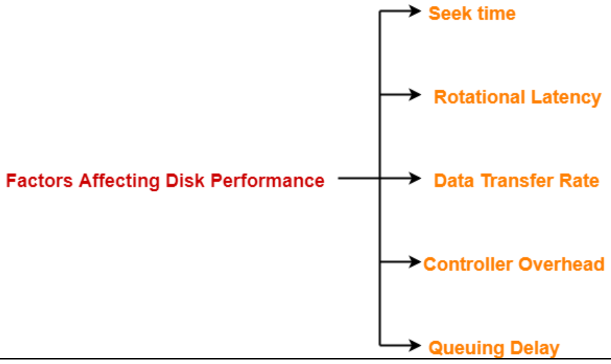
- A **cylinder** is formed by combining the tracks at a given radius of a disk pack.



- There exists a mechanical arm called as **Read / Write head**.
- It is used to read from and write to the disk.
- Head has to reach at a particular track and then wait for the rotation of the platter.
- The rotation causes the required sector of the track to come under the head.
- Each platter has 2 surfaces- top and bottom and both the surfaces are used to store the data.
- Each surface has its own read / write head.



Disk Performance Parameters-



Video Content / Details of website for further learning (if any):
https://www.youtube.com/watch?v=OjA4jSP_Na8

Important Books/Journals for further learning including the page nos.:
John P. Hayes, “Computer Architecture and Organization”, Tata McGraw-Hill, 3rd edition, 1998, page no. :420 to 424

Course Faculty

Verified by HOD

L - 32

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit IV : Memory Organization **Date of Lecture:**

Topic of Lecture: Optical memories

Introduction:
Compact Disks are optical storage media on which data is stored a read using laser light. CDs are available in the following types: CD-R and CD-RW

Prerequisite knowledge for Complete understanding and learning of Topic:
Microprocessors and Microcontrollers

Optical memories:
Compact Disks are optical storage media on which data is stored an read using laser light. CDs are available in the following types CD-R and CD-RW

CD-R: We can write data only once. We cannot rewrite the data. CD-RW We can rewrite data.

CDs can store data, audio or video files. Now CDs are coming with 700 MB storage capacity.

CD drives are also two types CD-ROM drive can only read from CDs. CD writer can read as well as write on to the CDs.

Advantages:

- Provides good storage capacity
- Can be replicated (duplicated) in expensively Removable.
- So they are very suitable for storage as well as for distribution of software or data.

Disadvantages:

- Accessing time is much more than magnetic disks (hard disks).

A CD is simple circular piece of plastic 12 cm in diameter and about 1.2 mm thick. Following figure shows the arrangement of tracks in CDs.

CD ROM drives use Constant Linear Velocity. A CD-ROM drive reads a disk starting at the innermost track. Inner tracks are shorter than outer tracks. So, drive travels faster on inner track than on outer track.

Data is written by creating pits and lands on the CD's surface. A pit is a depression on the surface, and a land is the height of the original surface. The transition from a land to a pit, or a pit to a land, represents a binary character of 1. Lands and pits represent binary 0. The reading of data is based on timing—the speed at which the CD is rotating — and the reflection of light. If no data is on the disk, the reflectivity will not change and the CD will read a series of binary Os. There are approximately 4 to 5 million pits per CD. They are arranged in a single outward-running spiral (track) approximately 3.75 miles (6 kilometers) long. The distance between each element is 1.6 thousandths of a millimeter.

During read operation, a read head containing a laser beam is passed over the spinning CD-ROM via a system of prisms and mirrors. The beam passes through the plastic coating to the aluminum layer. Most of the beam is absorbed when it hits a pit, so only a very light is reflected back. When a land is encountered, most of the light is bounced back.

Inside CD-ROM drive:

1. Laser Diode: This produces a highly accurate laser beam that can be targeted to 0.001 mm.

2. Prisms: The laser beam passes through a system of prisms that refine the beam.

3. Mirror: The mirror redirects the incoming light to the read head.

4. Read Head: The read head moves across the radius of the disc, directing the laser beam to the relevant area.

5. Motor (not shown in figure) to rotate the Disk: The disk rotates to bring new data in front of the laser.

6. Return Journey: The light is reflected by the surface of the disc and returns through the read head, the mirror, and prisms. On the return journey, the prism redirects the beam to the photo diode.

Photo Diode: The light sensitive component translates light reflected back from the disk into binary code and then passes it on to the processor.

Writers: The CD burner has a moving laser assembly, just like an ordinary CD player. But in addition to the standard "read laser," it has a "write laser." The write laser is more powerful than the read laser, so it interacts with the disc differently: It alters the surface instead of just bouncing light off it. Read lasers are not intense enough to darken the dye material, so simply playing a CD-R in a CD drive will not destroy any encoded information.

VD:

A Digital Versatile Disc (DVD) too is an optical storage medium. It also has the same shape and size as the CD-ROM, but has a much larger storage capacity. A DVD can store about seven times more data than a CD-ROM. The prices of the DVD media and DVD drives cost more than CD-ROMs and their drives.

DVD has two separate surfaces for writing pits and lands. As shown in Figure, the transparent surface that is near the top is one of the writing layers, and the opaque layer below that is the other. By being able to focus the concentrated light beam on either one of these layers, the capacity of a normal, single-layered disc is essentially doubled. This, in combination with the fact that DVDs have a more tightly wound spiral, and accordingly, a smaller laser to read them, is the reason a digital video disk has immensely more storage space than a typical compact disc. This capacity is furthered doubled in today's technology, because digital video disks now have both a transparent and opaque layer on each side of the disc.

DVD To use a DVD in a computer, you need a DVD drive. It looks similar to that of a CD-ROM drive. The DVD logo placed on the front side of the drive helps distinguish the two types of drives. DVD drives can read both DVD and CD-ROM media. These drives are set to replace CD-ROM drives in the near future.

Many of the components inside a DVD drive are similar to that found inside a CD-ROM drive. The DVD drive has a more complex laser assembly because it has to read both DVDs and CD-ROMs.

When compared to CD-ROM drives, DVD drives spin the disc much slower. A DVD drive though has much better throughput. It means that the drive can transfer data to the computer fast. That is, more kilobytes per second (KBps). Higher throughput results because the data bytes are stored much closer to each other in a DVD. In a CD-ROM the data is stored near the top layer; in a DVD the same is stored in the middle layer. The drive reads and collects the patterns of Os and is and sends them to the computer. It decodes the data to display it on the screen.

Video Content / Details of website for further learning (if any):

https://www.youtube.com/watch?v=LQiEyyKmVRI

Important Books/Journals for further learning including the page nos.:

John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no. : 424 to 425

Course Faculty

Verified by HOD

L - 33

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit IV : Memory Organization **Date of Lecture:**

Topic of Lecture: Multilevel memories

Introduction:
The memory is that part of computer where programs and data are stored. The basical concept is the following: Bits. The basic unit of memory is the binary digit called a bit. A bit may contain a 0 or 1. It is the simplest possible unit

Prerequisite knowledge for Complete understanding and learning of Topic:
Microprocessors and Microcontrollers

Organization of Multilevel Memory System:
A CPU should have rapid, uninterrupted access to its external memories to operate at or near at its maximum speed. But memories that operate at speed near to that of CPU are expensive. That’s why different level of memories is used in terms of performance and costs. Below is the flow-diagram of the conceptual organization of multilevel memories in a computer system.

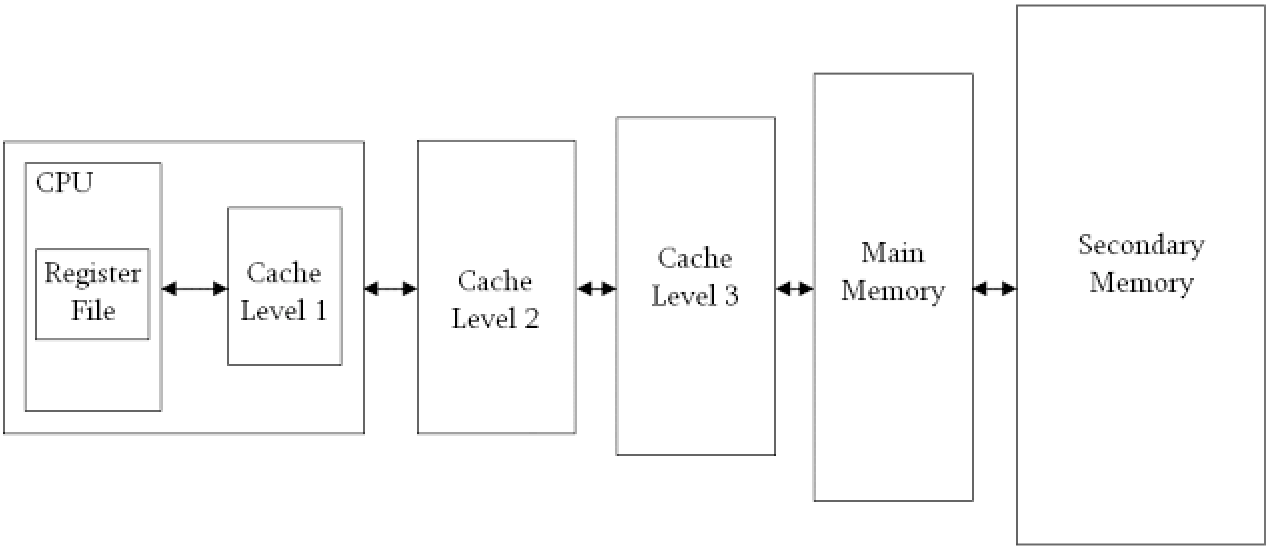
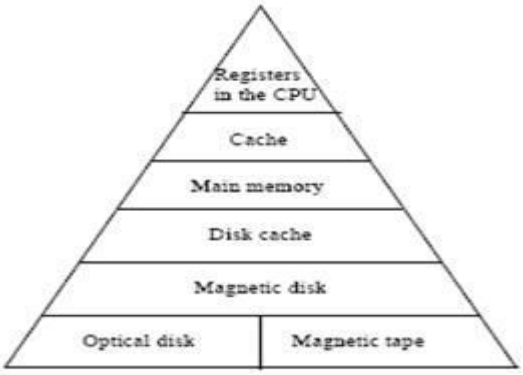


Figure : Conceptual Organization of Multilevel Memories in a Computer System

Three key characteristics increase for a memory hierarchy. They are the access time, the storage capacity and the cost. The memory hierarchy is illustrated in figure 9.1.

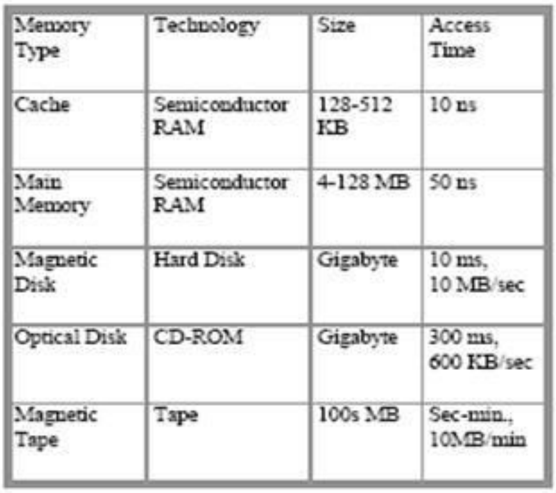
The memory hierarchy

We can see the memory hierarchy with six levels. At the top there are CPU registers, which can be accessed at full CPU speed. Next commes the cache memory, which is currently on order of 32 KByte to a few Mbyte. The main memory is next, with size currently ranging from 16 MB for entry-level systems to tens of Gigabytes. After that come magnetic disks, the current work horse for permanent storage. Finally we have magnetic tape and optical disks for archival storage.



Basis of the memory hierarchy

– Registers internal to the CPU for temporary data storage (small in number but very fast)
– External storage for data and programs (relatively large and fast)
– External permanent storage (much larger and much slower)



Memory Type	Technology	Size	Access Time
Cache	Semiconductor RAM	128-512 KB	10 ns
Main Memory	Semiconductor RAM	4-128 MB	50 ns
Magnetic Disk	Hard Disk	Gigabyte	10 ms, 10 MB/sec
Optical Disk	CD-ROM	Gigabyte	300 ms, 600 KB/sec
Magnetic Tape	Tape	100s MB	Sec-min., 10MB/min

Typical Memory Parameters

Characteristics of the memory hierarchy
– Consists of distinct “levels” of memory components
– Each level characterized by its size, access time, and cost per bit
– Each increasing level in the hierarchy consists of modules of larger capacity, slower access time, and lower cost/bit

Video Content / Details of website for further learning (if any):
https://www.youtube.com/watch?v=8rgp0MxVu74

Important Books/Journals for further learning including the page nos.:
John P. Hayes, “Computer Architecture and Organization”, Tata McGraw-Hill, 3rd edition, 1998, page no. :426 to 427

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

DESIGNING YOUR FUTURE
Estd. 2000

IQAC

L - 34

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit IV : Memory Organization **Date of Lecture:**

Topic of Lecture: cache and virtual memory

Introduction:

Cache memory is a very high speed semiconductor memory which can speed up CPU. It acts as a buffer between the CPU and the main memory. Virtual memory is a technique that allows the execution of processes which are not completely available in memory. The main visible advantage of this scheme is that programs can be larger than physical memory.

Prerequisite knowledge for Complete understanding and learning of Topic:

Microprocessors and Microcontrollers

Cache Memory:

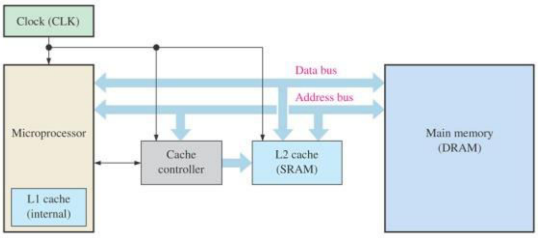
Cache memory is a very high speed semiconductor memory which can speed up CPU. It acts as a buffer between the CPU and the main memory.

It is used to hold those parts of data and program which are most frequently used by the CPU. The parts of data and programs, are transferred from disk to cache memory by operating system, from where CPU can access them. Cache memory, lies in between CPU and the main memory. It is also called CPU memory, that a computer microprocessor can access more quickly than it can access regular RAM. This memory is typically integrated directly with the CPU chip or placed on a separate chip that has a separate bus interconnect with the CPU. Cache memory saves time and increases efficiency because the most recently processing data is stored in it which takes the fetching easier.

Functions of Cache Memory.

Advantages of Cache Memory :Cache memory is faster than main memory. It consumes less access time as compared to main memory. It is stores the program that can be executed within a short period of time. It stores data for temporary use.

Disadvantages of Cache Memory : Cache memory has limited capacity. Cache memory is very expensive.



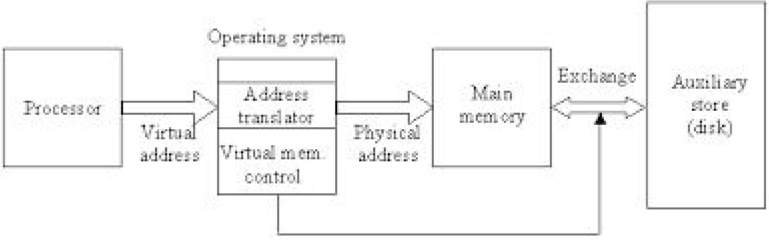
Block diagram showing L1 and L2 cache memories in a computer system.

Virtual Memory:

Virtual memory is a valuable concept in computer architecture that allows you to run large, sophisticated programs on a computer even if it has a relatively small amount of RAM. A computer with virtual memory artfully juggles the conflicting demands of multiple programs within a fixed amount of physical memory. A PC that's low on memory can run the same programs as one with abundant RAM, although more slowly.

It is a technique that allows the execution of processes which are not completely available in memory. The main visible advantage of this scheme is that programs can be larger than physical memory. Virtual memory is the separation of user logical memory from physical memory.

Virtual memory breaks programs into fixed-size blocks called pages. If a computer has abundant physical memory, the operating system loads all of a program's pages into RAM. If not, the OS fits as much as it can and runs the instructions in those pages. When the computer is done with those pages, it loads the rest of the program into RAM, possibly overwriting earlier pages. Because the operating system automatically manages these details, this frees the software developer to concentrate on program features and not worry about memory issues.



Video Content / Details of website for further learning (if any):

https://www.youtube.com/watch?v=jyu5RKDwkTA

https://www.youtube.com/watch?v=o2_iCzS9-ZQ

Important Books/Journals for further learning including the page nos.:

John P. Hayes, “Computer Architecture and Organization”, Tata McGraw-Hill, 3rd edition, 1998, page no. :427 to 443

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

Estd. 2000

L - 35

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit IV : Memory Organization **Date of Lecture:**

Topic of Lecture: Memory allocation

Introduction:

Memory allocation is a process by which computer programs and services are assigned with physical or virtual memory space.

Prerequisite knowledge for Complete understanding and learning of Topic:

Microprocessors and Microcontrollers

Memory allocation:

Memory allocation is primarily a computer hardware operation but is managed through operating system and software applications. Memory allocation process is quite similar in physical and virtual memory management. Programs and services are assigned with a specific memory as per their requirements when they are executed. Once the program has finished its operation or is idle, the memory is released and allocated to another program or merged within the primary memory.

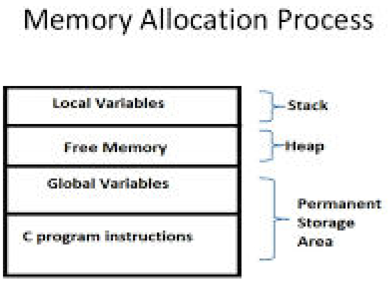
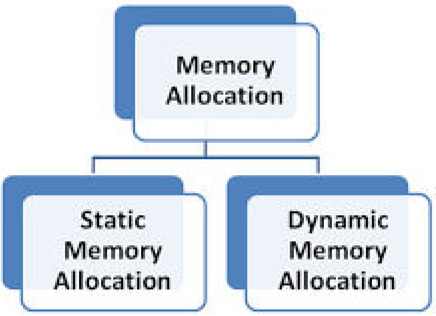
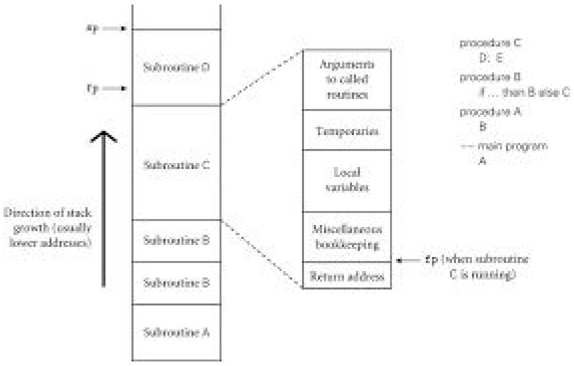


Fig: Storage of a C program

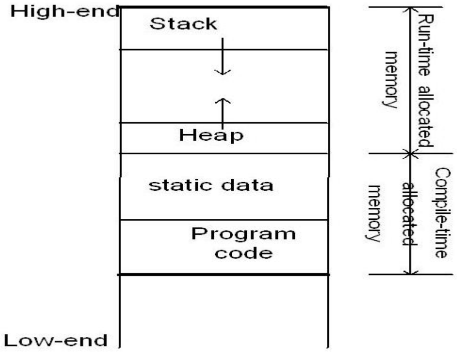
Memory allocation has two core types:
1. Static Memory Allocation: The program is allocated memory at compile time.
2. Dynamic Memory Allocation: The programs are allocated with memory at run time.



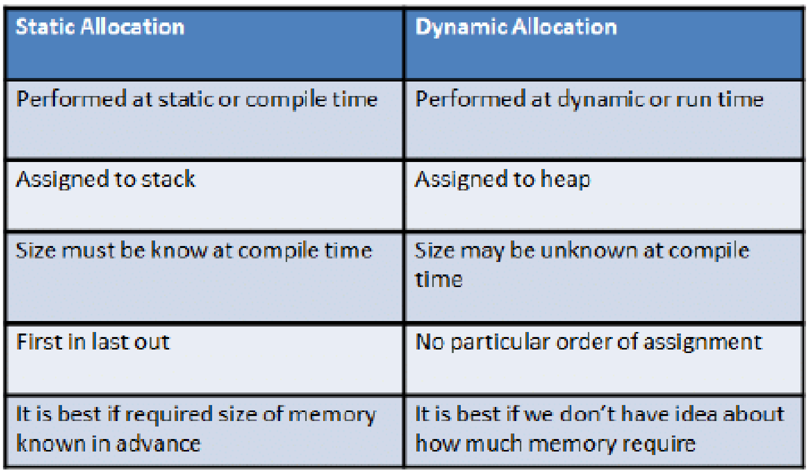
Static memory allocation:



Dynamic memory allocation:



Comparison of static and dynamic memory allocation:



Static Allocation	Dynamic Allocation
Performed at static or compile time	Performed at dynamic or run time
Assigned to stack	Assigned to heap
Size must be know at compile time	Size may be unknown at compile time
First in last out	No particular order of assignment
It is best if required size of memory known in advance	It is best if we don't have idea about how much memory require

Video Content / Details of website for further learning (if any):
https://www.youtube.com/watch?v=N3rG_1CEQkQ

Important Books/Journals for further learning including the page nos.:
John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no. :443 to 452

Course Faculty

Verified by HOD

L - 36

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit IV : Memory Organization **Date of Lecture:**

Topic of Lecture: Associative memory

Introduction:

A type of computer memory from which items may be retrieved by matching some part of their content, rather than by specifying their address (hence also called **associative storage** or *Content-addressable memory* (CAM).) *Associative memory* is much slower than RAM, and is rarely encountered in mainstream computer designs.

Prerequisite knowledge for Complete understanding and learning of Topic:
Microprocessors and Microcontrollers

Associative Memory:

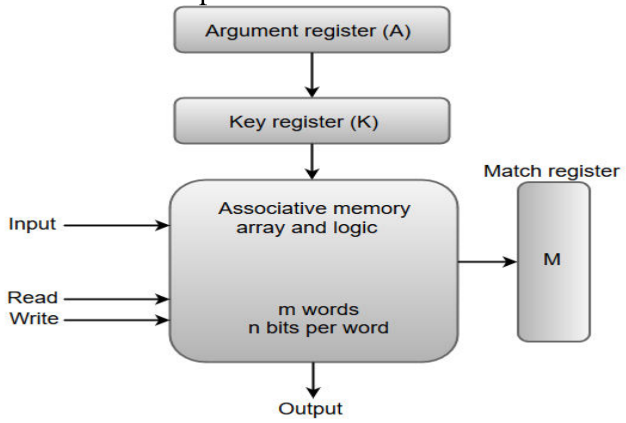
An associative memory can be considered as a memory unit whose stored data can be identified for access by the content of the data itself rather than by an address or memory location.

Associative memory is often referred to as **Content Addressable Memory (CAM)**.

When a write operation is performed on associative memory, no address or memory location is given to the word. The memory itself is capable of finding an empty unused location to store the word.

On the other hand, when the word is to be read from an associative memory, the content of the word, or part of the word, is specified. The words which match the specified content are located by the memory and are marked for reading.

The following diagram shows the block representation of an Associative memory.



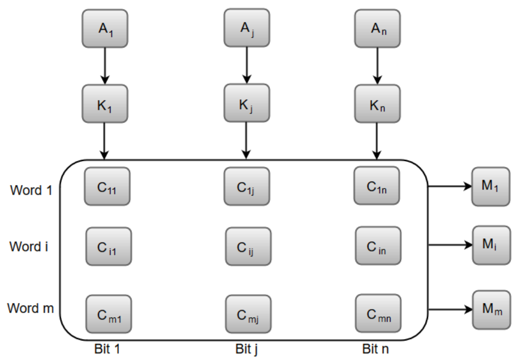
From the block diagram, we can say that an associative memory consists of a memory array and logic for 'm' words with 'n' bits per word.

The functional registers like the argument register **A** and key register **K** each have **n** bits, one for each bit of a word. The match register **M** consists of **m** bits, one for each memory word.

The words which are kept in the memory are compared in parallel with the content of the argument register.

The key register (K) provides a mask for choosing a particular field or key in the argument word. If the key register contains a binary value of all 1's, then the entire argument is compared with each memory word. Otherwise, only those bits in the argument that have 1's in their corresponding position of the key register are compared. Thus, the key provides a mask for identifying a piece of information which specifies how the reference to memory is made.

The following diagram can represent the relation between the memory array and the external registers in an associative memory.



Associative memory of m word, n cells per word:

The cells present inside the memory array are marked by the letter C with two subscripts. The first subscript gives the word number and the second specifies the bit position in the word. For instance, the cell C_{ij} is the cell for bit **j** in word **i**.

A bit A_j in the argument register is compared with all the bits in column **j** of the array provided that $K_j = 1$. This process is done for all columns **j** = 1, 2, 3......, n.

If a match occurs between all the unmasked bits of the argument and the bits in word **i**, the corresponding bit M_i in the match register is set to 1. If one or more unmasked bits of the argument and the word do not match, M_i is cleared to 0.

Video Content / Details of website for further learning (if any):

https://www.youtube.com/watch?v=hZ5_02KW1Y8

Important Books/Journals for further learning including the page nos.:

John P. Hayes, “Computer Architecture and Organization”, Tata McGraw-Hill, 3rd edition, 1998, page no. : 458 to 465

Course Faculty

Verified by HOD

L - 37

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit V : Input/output and System Organization Date of Lecture:

Topic of Lecture: Communication methods, Buses, Bus control unit, Bus interfacing

Introduction:

A **bus** is a **common pathway** through which information flows from one computer component to another. This pathway is used for communication purpose and it is established between two or more computer components

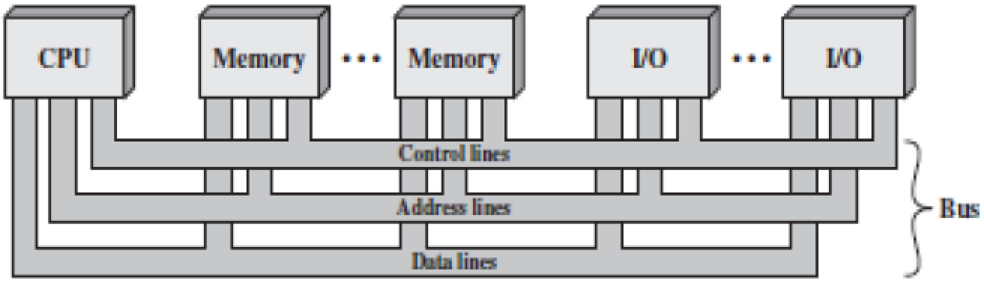
Prerequisite knowledge for Complete understanding and learning of Topic:

Microprocessors and Microcontrollers

Communication methods, Buses, Bus control unit, Bus interfacing:

A bus is a communication pathway connecting two or more devices. A key characteristic of a bus is that it is a shared transmission medium. Multiple devices connect to the bus, and a signal transmitted by any one device is available for reception by all other devices attached to the bus. If two devices transmit during the same time period, their signals will overlap and become garbled. Thus, only one device at a time can successfully transmit. Typically, a bus consists of multiple communication pathways, or lines. Each line is capable of transmitting signals representing binary 1 and binary 0. An 8-bit unit of data can be transmitted over eight bus lines. A bus that connects major computer components (processor, memory, I/O) is called a system bus.

Bus Structure



Bus Interconnection Schemes

On any bus the lines can be classified into three functional groups (Figure 1.17): data, address, and control lines. In addition, there may be power distribution lines that supply power to the attached modules. The **data lines** provide a path for moving data among system modules. These lines, collectively, are called the data bus.

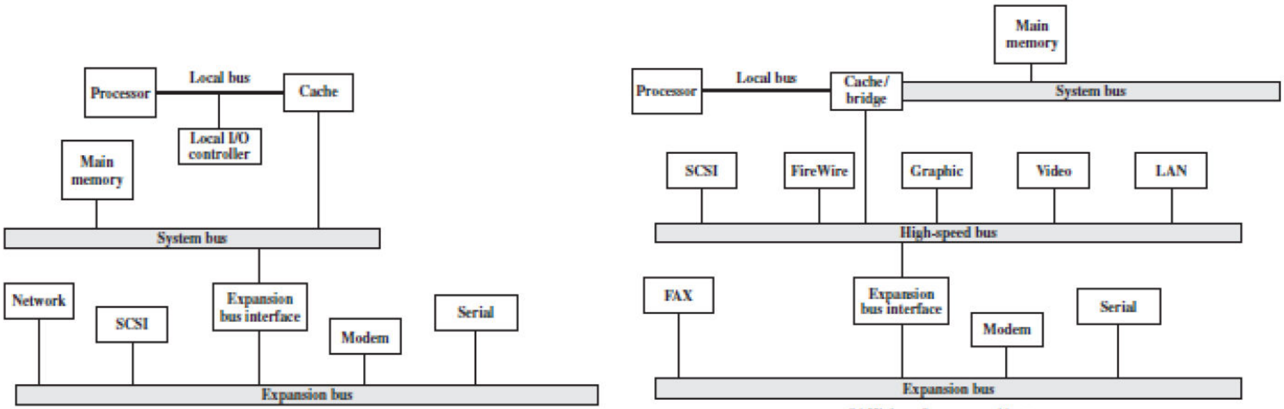
The **address lines** are used to designate the source or destination of the data on the data bus.

For example, on an 8-bit address bus, address 01111111 and below might reference locations in a memory module (module 0) with 128 words of memory, and address 10000000 and above refer to devices attached to an I/O module (module 1).

The **control lines** are used to control the access to and the use of the data and address lines. Control signals transmit both command and timing information among system modules. Timing signals indicate the validity of data and address information. Command signals specify operations to be performed. Typical control lines include

- **Memory write:** Causes data on the bus to be written into the addressed location
- **Memory read:** Causes data from the addressed location to be placed on the bus
- **I/O write:** Causes data on the bus to be output to the addressed I/O port
- **I/O read:** Causes data from the addressed I/O port to be placed on the bus
- • **Bus request:** Indicates that a module needs to gain control of the bus
- **Bus grant:** Indicates that a requesting module has been granted control of the bus
- **Interrupt request:** Indicates that an interrupt is pending
- **Interrupt ACK:** Acknowledges that the pending interrupt has been recognized
- **Clock:** Is used to synchronize operations

The operation of the bus is as follows. If one module wishes to send data to another, it must do two things: (1) obtain the use of the bus, and (2) transfer data via the bus. If one module wishes to request data from another module, it must (1) obtain the use of the bus, and (2) transfer a request to the other module over the appropriate control and address lines. It must then wait for that second module to send the data.



(a) Traditional bus architecture

(b) High-performance architecture

During data transfer operation, one device plays the role of a **Master.**

Master device initiates the data transfer by issuing read / write command on the bus. Hence it is also called as **Initiator**. The device addressed by the master is called as **Slave / Target**.

Types of Buses:

There are 2 types of buses. They are, □ Synchronous Bus and □ Asynchronous Bus.

Synchronous Bus:-

In synchronous bus, all devices derive timing information from a common clock line. Equally spaced pulses on this line define equal time.

Asynchronous Bus:-

An alternate scheme for controlling data transfer on. The bus is based on the use of **handshake** between **Master** & the **Slave**. The common clock is replaced by two timing control lines. They are

□ Master–ready

□ Slave ready.

Video Content / Details of website for further learning (if any):

https://www.youtube.com/watch?v=TgYAj7mlRT8

Important Books/Journals for further learning including the page nos.:

John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no. : 480 to 498

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

Estd. 2000

L - 38

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit V : Input/output and System Organization Date of Lecture:

Topic of Lecture: Bus arbitration, I/O and system control

Introduction:
I/O **control** Either the hardware that **controls** the transfer of data between main memory and peripheral devices, or the part of the system software that in turn **controls** that hardware.

Prerequisite knowledge for Complete understanding and learning of Topic:
Microprocessors and Microcontrollers

Bus Arbitration:
The various methods can be roughly classified as being either centralized or distributed. In a centralized scheme, a single hardware device, referred to as a bus controller or arbiter, is responsible for allocating time on the bus. In a distributed scheme, there is no central controller. Rather, each module contains access control logic and the modules act together to share the bus. With both methods of arbitration, the purpose is to designate either the processor or an I/O module, as master. The master may then initiate a data transfer (e.g., read or write) with some other device, which acts as slave for this particular exchange.

□ **Timing Buses** use either synchronous timing or asynchronous timing. With **synchronous timing**, the occurrence of events on the bus is determined by a clock. A single 1–0 transmission is referred to as a clock cycle or bus cycle and defines a time slot.

Bus Master:
The device that is allowed to initiate data transfers on the bus at any given time is called the bus master.

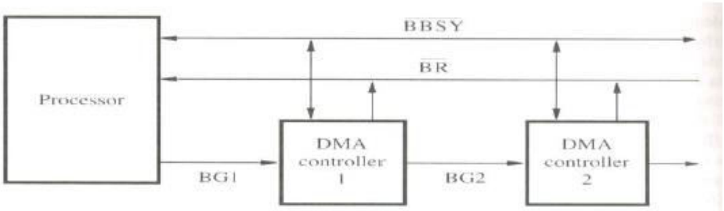
Bus Arbitration:
It is the process by which the next device to become the bus master is selected and the bus mastership is transferred to it.

Types:
There are 2 approaches to bus arbitration. They are,
□ Centralized arbitration (A single bus arbiter performs arbitration)
□ Distributed arbitration (all devices participate in the selection of next bus master).

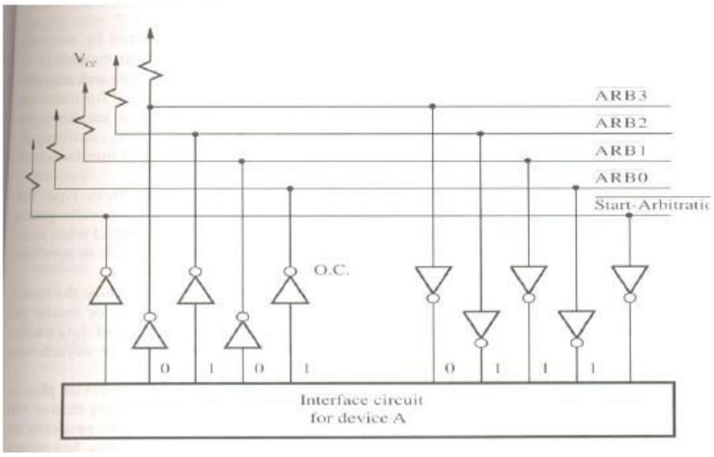
Centralized Arbitration:
Here the processor is the bus master and it may grant bus mastership to one of its DMA controller.

A DMA controller indicates that it needs to become the bus master by activating the Bus Request line (BR) which is an open drain line.

The signal on BR is the logical OR of the bus request from all devices connected to it. When BR is activated the processor activates the Bus Grant Signal (BGI) and indicated the DMA controller that they may use the bus when it becomes free. This signal is connected to all devices using a daisy chain arrangement. If DMA requests the bus, it blocks the propagation of Grant Signal to other devices and it indicates to all devices that it is using the bus by activating open collector line, Bus Busy (BBSY).



A distributed arbitration scheme



Distributed Arbitration:

It means that all devices waiting to use the bus have equal responsibility in carrying out the arbitration process. Each device on the bus is assigned a 4 bit id. When one or more devices request the bus, they assert the Start-Arbitration signal & place their 4 bit ID number on four open collector lines, ARB0 to ARB3. A winner is selected as a result of the interaction among the signals transmitted over these lines. The net outcome is that the code on the four lines represents the request that has the highest ID number. The drivers are of open collector type. Hence, if the i/p to one driver is equal to 1, the i/p to another driver connected to the same bus line is equal to „0‟(ie. bus the is in low-voltage state).

I/O and System control:

Interface Circuits:

The interface circuits are of two types. They are □ Parallel Port and □ Serial Port

Standard I/O Interface

A standard I/O Interface is required to fit the I/O device with an Interface circuit. The processor bus is the bus defined by the signals on the processor chip itself. The devices that require a very high speed connection to the processor such as the main memory, may be connected directly to this bus. **The bridge** connects two buses, which translates the signals and protocols of one bus into another. The bridge circuit introduces a small delay in data transfer between processor and the devices.

We have 3 Bus standards.They are,

□ **PCI** (Peripheral Component Inter Connect)

□ **SCSI** (Small Computer System Interface)

□ **USB** (Universal Serial Bus)

Video Content / Details of website for further learning (if any):

https://www.youtube.com/watch?v=Yp_3OrDhUkQ

https://www.youtube.com/watch?v=vEXX5DJT7vk

Important Books/Journals for further learning including the page nos.:

John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no. :498 to 509

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

DESIGNING YOUR FUTURE
Estd. 2000

IQAC

L - 39

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit V : Input/output and System Organization Date of Lecture:

Topic of Lecture: IO interface circuits, Handshaking

Introduction:

An **I/O interface** consists of the circuitry required to connect an **I/O** device to a **computer** bus. On the side of the **interface** we have the bus signals for address, data and control. On the other side we have data path with its associated controls to transfer data between the **interface** and the **I/O** device. In communication, **handshaking** is the automated process for negotiation of setting up a communication channel between entities. **Handshaking** occurs before the transfer of data or any other communication and just after the establishment of the physical channel between the two entities.

Prerequisite knowledge for Complete understanding and learning of Topic:

Microprocessors and Microcontrollers

IO interface circuits:

An I/O interface consists of the circuitry required to connect an I/O device to a computer bus. On the side of the interface we have the bus signals for address, data and control. On the other side we have data path with its associated controls to transfer data between the interface and the I/O device. This side is called a port, and it can be classified as either a parallel or a serial port.

->A parallel port transfers data in the form of a number of bits, typically 8 or 16 simultaneously to or from the device.

->A serial port transmits and receives data one bit at a time

->The conversation from the parallel to the serial format and vice-versa, takes place inside the interface circuit.

->I/O interface does the following:

- Provides storage buffer for at least one word of data
- contains status flags that can be accessed by he processor to determine whether the buffer is full.

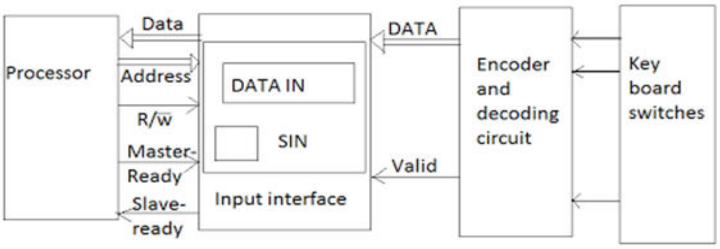


Fig:Keyboard to processor connection

- contains address-decoding circuitry to determine when it is being addressed by the processor.

- Generates the appropriate timing signals required by the bus control scheme.

Performs any format conversion that may be necessary to transfer data between he bus and the I/O device, such as parallel-serial conversion in the case of a serial port.

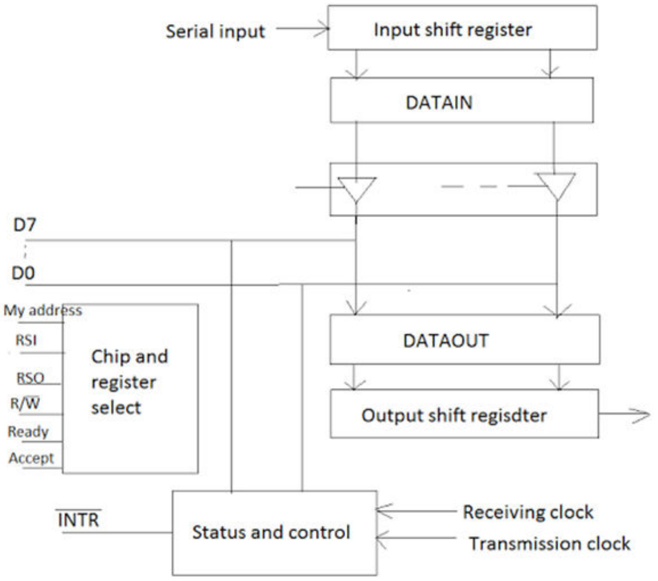
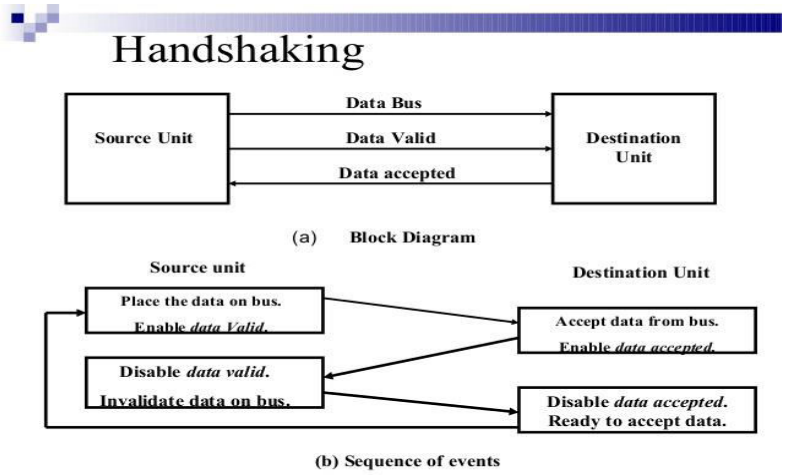


Fig:A serial interface

Handshaking:

The handshake can provide the necessary information or protocols for the sender and receiver. It allows the receiving device to know how to receive the input data from the sender and then output the received data in the necessary format applicable to the receiver. It also provides the provisions of how the communication between the devices should continue. This is especially required when the devices are foreign to each other, like a computer to a modem, server, etc.


Handshaking
Source Unit
Data Bus
Data Valid
Data accepted
Destination Unit
(a) Block Diagram
Source unit
Destination Unit
Place the data on bus. Enable data Valid.
Accept data from bus. Enable data accepted.
Disable data valid. Invalidate data on bus.
Disable data accepted. Ready to accept data.
(b) Sequence of events

Video Content / Details of website for further learning (if any):
https://www.youtube.com/watch?v=1QGcl8NDxCQ

Important Books/Journals for further learning including the page nos.:
John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no. 509 to 511,497 to 499

Course Facult

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

Estd. 2000

L - 40

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit V : Input/Output and System Organization Date of Lecture:

Topic of Lecture: DMA, interrupt and vectored interrupts

Introduction:

A special control unit may be provided to allow the transfer of large block of data at high speed directly between the external device and main memory, without continuous intervention by the processor. This approach is called **DMA.** An **interrupt** is a signal to the processor emitted by hardware or software indicating an event that needs immediate attention. A **vectored interrupt** is where the CPU actually knows the address of the **Interrupt** Service Routine in advance.

Prerequisite knowledge for Complete understanding and learning of Topic:

Microprocessors and Microcontrollers

Direct Memory Access (DMA):

A special control unit may be provided to allow the transfer of large block of data at high speed directly between the external device and main memory, without continuous intervention by the processor. This approach is called **DMA.**

DMA transfers are performed by a control circuit called the **DMA Controller**.

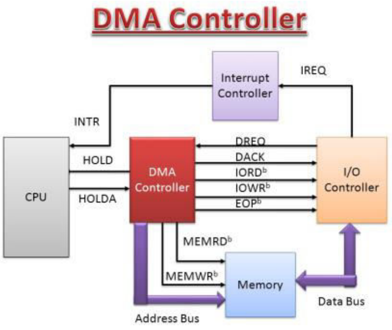
To initiate the transfer of a block of words , the processor sends,

- Starting address
- Number of words in the block
- Direction of transfer.

When a block of data is transferred, the DMA controller increment the memory address for successive words and keep track of number of words and it also informs the processor by raising an interrupt signal.

While DMA control is taking place, the program requested the transfer cannot continue and the processor can be used to execute another program.

After DMA transfer is completed, the processor returns to the program that requested the transfer.



A DMA controller connects a high speed network to the computer bus . The disk controller two disks, also has DMA capability and it provides two DMA channels.

To start a DMA transfer of a block of data from main memory to one of the disks, the program write s the address and the word count inf. Into the registers of the corresponding channel of the disk controller.

Cycle Stealing:

Requests by DMA devices for using the bus are having higher priority than processor requests.

Top priority is given to high speed peripherals such as,

- Disk
- High speed Network Interface and Graphics display device.

Since the processor originates most memory access cycles, the DMA controller can be said to steal the memory cycles from the processor. This interviewing technique is called **Cycle stealing.**

Burst Mode:

The DMA controller may be given exclusive access to the main memory to transfer a block of data without interruption. This is known as **Burst/Block Mode**

Interrupt:

Normal execution of the program may be pre-empted if some device requires urgent servicing. Eg...Monitoring Device in a computer controlled industrial process may detect a dangerous condition.

In order to deal with the situation immediately, the normal execution of the current program may be interrupted & the device raises an interrupt signal.

The processor provides the requested service called the Interrupt Service Routine(ISR). ISR save the internal state of the processor in memory before servicing the interrupt because interrupt may alter the state of the processor. When ISR is completed, the state of the processor is restored and the interrupted program may continue its execution.

Vectored Interrupt:

Here the device requesting an interrupt may identify itself to the processor by sending a special code over the bus & then the processor start executing the ISR. The code supplied by the processor indicates the starting address of the ISR for the device. The code length ranges from 4 to 8 bits. The location pointed to by the interrupting device is used to store the staring address to ISR. The processor reads this address, called the interrupt vector & loads into PC. The interrupt vector also includes a new value for the Processor Status Register. When the processor is ready to receive the interrupt vector code, it activates the interrupt acknowledge (INTA) line.

Video Content / Details of website for further learning (if any):

https://www.youtube.com/watch?v=ltvpkuQRZao
https://www.youtube.com/watch?v=lxBfuyOagS8

Important Books/Journals for further learning including the page nos.:

John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no. :511 to 520

Course Faculty

Verified by HOD

L - 41

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit V : Input/Output and System Organization Date of Lecture:

Topic of Lecture: Pipeline interrupts

Introduction:

Pipelining is the process of accumulating instruction from the processor through a **pipeline**. It allows storing and executing instructions in an orderly process. It is also known as **pipeline** processing. **Pipelining** is a technique where multiple instructions are overlapped during execution.

Prerequisite knowledge for Complete understanding and learning of Topic:

Microprocessors and Microcontrollers

Pipeline Interrupts:

The pipeline has two independent stages. The first stage fetches an instruction and buffers it. When the second stage is free, the first stage passes it the buffered instruction. While the second stage is executing the instruction, the first stage takes advantage of any unused memory cycles to fetch and buffer the next instruction. This is called *instruction prefetch* or *fetch overlap*. This process will speed up instruction execution only if the fetch and execute stages were of equal duration, the instruction cycle time would be halved. However, if we look more closely at this pipeline, we will see that this doubling of execution rate is unlikely for 3 reasons:

1 The execution time will generally be longer than the fetch time. Thus, the fetch stage may have to wait for some time before it can empty its buffer.

2 A conditional branch instruction makes the address of the next instruction to be fetched unknown. Thus, the fetch stage must wait until it receives the next instruction address from the execute stage. The execute stage may then have to wait while the next instruction is fetched.

3 When a conditional branch instruction is passed on from the fetch to the execute stage, the fetch stage fetches the next instruction in memory after the branch instruction. Then, if the branch is not taken, no time is lost .If the branch is taken, the fetched instruction must be discarded and a new instruction fetched. To gain further speedup, the pipeline must have more stages. Let us consider the following decomposition of the instruction processing.

1. **Fetch instruction (FI):** Read the next expected instruction into a buffer.
2. **Decode instruction (DI):** Determine the opcode and the operand specifiers.

Calculate operands (CO): Calculate the effective address of each source operand. This may involve displacement, register indirect, indirect, or other forms of address calculation.

4. **Fetch operands (FO):** Fetch each operand from memory.
5. **Execute instruction (EI):** Perform the indicated operation and store the result, if any, in the specified destination operand location.
6. **Write operand (WO):** Store the result in memory.

For high-performance in pipelining designer must still consider about :

1 At each stage of the pipeline, there is some overhead involved in moving data from buffer to buffer and in performing various preparation and delivery functions. This overhead can appreciably lengthen the total execution time of a single instruction.
2 The amount of control logic required to handle memory and register dependencies and to optimize the use of the pipeline increases enormously with the number of stages. This can lead to a situation where the logic controlling the gating between stages is more complex than the stages being controlled.
3 Latching delay: It takes time for pipeline buffers to operate and this adds to instruction cycle time. Interrupt processing within a processor is a facility provided to support the operating system. It allows an application program to be suspended, in order that a variety of interrupt conditions can be serviced and later resumed.

INTERRUPTS AND EXCEPTIONS An *interrupt* is generated by a signal from hardware, and it may occur at random times during the execution of a program. An *exception* is generated from software, and it is provoked by the execution of an instruction. There are two sources of interrupts and two sources of exceptions:

1. Interrupts

- **Maskable interrupts:** Received on the processor's INTR pin. The processor does not recognize a maskable interrupt unless the interrupt enable flag (IF) is set.
- **Nonmaskable interrupts:** Received on the processor's NMI pin. Recognition of such interrupts cannot be prevented.

Exceptions

- **Processor-detected exceptions:** Results when the processor encounters an error while attempting to execute an instruction.
- **Programmed exceptions:** These are instructions that generate an exception (e.g., INTO, INT3, INT, and BOUND).

Video Content / Details of website for further learning (if any):
https://www.youtube.com/watch?v=zOJ8om06PU4

Important Books/Journals for further learning including the page nos.:
John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no. :522 to 525

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

Estd. 2000

L - 42

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit V : Input/Output and System Organization Date of Lecture:

Topic of Lecture: IOP organization

Introduction:

An **input-output processor** (**IOP**) is a processor with direct memory access capability. In this, the **computer** system is divided into a memory unit and number of processors. ... The **IOP** is similar to CPU except that it handles only the details of I/O processing. The **IOP** can fetch and execute its own instructions.

One of the important jobs of an **Operating System** is to **manage** various I/**O devices** including mouse, keyboards, touch pad, disk drives, display adapters, USB **devices**, Bit-mapped screen, LED, Analog-to-digital converter, On/off switch, network connections, audio I/**O**, printers etc.

Prerequisite knowledge for Complete understanding and learning of Topic:

Microprocessors and Microcontrollers

Input-output processor (**IOP**)

It is a specially designed microprocessor having a local memory of its own, which is used to control I/O devices with minimum CPU involvement.

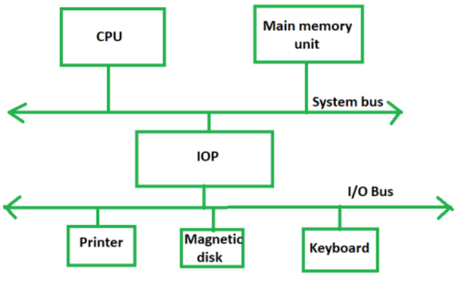
For example −

- DMA (direct Memory Access) controller
- Keyboard/mouse controller
- Graphic display controller
- SCSI port controller

An **input-output processor** (**IOP**) is a processor with direct memory access capability. In this, the **computer** system is divided into a memory unit and number of processors. ... The **IOP** is similar to CPU except that it handles only the details of I/O processing. The **IOP** can fetch and execute its own instructions.

- The Input Output **Processor** is a specialized **processor** which loads and stores data into memory along with the execution of **I/O** instructions. It acts as an interface between system and devices. It involves a sequence of events to executing **I/O** operations and then store the results into the memory.

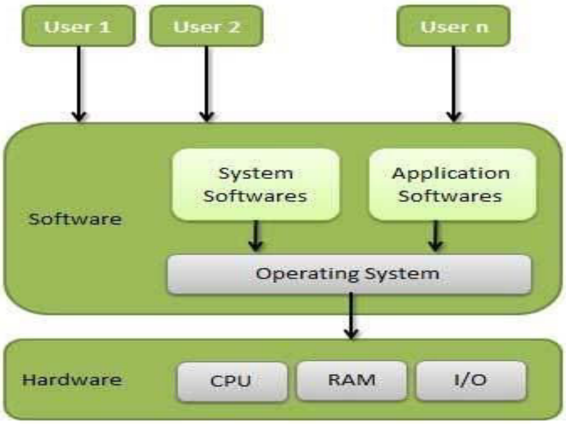
The stream flowing from an **input** device like a keyboard to the main memory, it is called the **Input Operation**. On the other hand, streams that flow from the main memory to an **output** device like a screen is called an **Output Operation**.



Input-Output Processor

Operating Systems:

An operating system is a program that acts as an interface between the user and the computer hardware and controls the execution of all kinds of programs.



Following are some of important functions of an operating System.

- Memory Management
- Processor Management
- Device Management
- File Management
- Security
- Control over system performance
- Job accounting
- Error detecting aids
- Coordination between other software and users

Video Content / Details of website for further learning (if any):

https://www.youtube.com/watch?v=f_AMb0MGRFA

https://www.youtube.com/watch?v=bkSWJJZNgf8&list=PLxCzCOWd7aiGz9donHRrE9I3Mwn6XdP8p

Important Books/Journals for further learning including the page nos.:

John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no. : 525 to 528

Course Faculty

Verified by HOD

L - 43

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit V : Input/Output and System Organization Date of Lecture:

Topic of Lecture: Multiprocessor, Fault tolerance

Introduction:

A **Multiprocessor** is a computer system with two or more central processing units (CPUs) share full access to a common RAM. The main objective of using a **multiprocessor** is to boost the system's execution speed, with other objectives being fault tolerance and application matching.

Prerequisite knowledge for Complete understanding and learning of Topic:

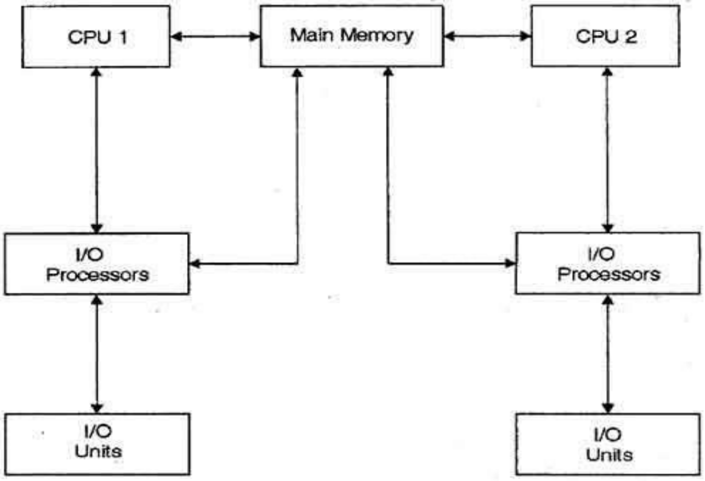
Microprocessors and Microcontrollers

Multiprocessor:

A **Multiprocessor** is a computer system with two or more central processing units (CPUs) share full access to a common RAM. The main objective of using a **multiprocessor** is to boost the system's execution speed, with other objectives being fault tolerance and application matching.

The benefit of having **multiple** cores is that the system can handle more than one thread simultaneously. Each **core** can handle a separate stream of data. This architecture greatly increases the **performance** of a system that is running concurrent applications.

Multi-processing refers to the ability of a system to support more than one processor at the same time. Applications in a **multi-processing** system are broken to smaller routines that run independently. The operating system allocates these threads to the processors improving performance of the system.



Fault Tolerance:

Fault tolerance is the property that enables a system to continue operating properly in the event of the failure of some of its components. If operating quality decreases then decrease is proportional to the severity of the failure, as compared to a naively designed system in which even a small failure can cause total breakdown. ¬Fault tolerance is particularly sought in high-availability or life-critical systems. It is the art and science of building computing systems that continue to operate satisfactorily in the presence of faults.

When a system or module is designed, its behavior is specified. When in service, we can observe its behavior. When the observed behavior differs from the specified behavior, we call it a failure. A failure occurs because of an error, caused by a fault.

Fault tolerance and dependable systems research covers a wide spectrum of applications ranging across embedded real-time systems, commercial transaction systems, transportation systems, and military/space systems -- to name a few.¬The supporting research includes system architecture, design techniques, coding theory, testing, validation, modeling, software reliability, operating systems, parallel processing, and real-time processing.

Fault Tolerance Requirements : The basic characteristics of fault tolerance are :

1.No single point of failure

2.No single point of repair

3.Fault isolation to the failing component

4.Fault containment to prevent propagation of the failure

5.Availability of reversion modes. In addition, fault tolerant systems are characterized in terms of both planned service outages and unplanned service outages.

Video Content / Details of website for further learning (if any):

https://www.youtube.com/watch?v=UnWMG4wRZuc

Important Books/Journals for further learning including the page nos.:

John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no. : 550 to 576

Course Faculty

Verified by HOD

MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu

Estd. 2000

L - 44

LECTURE HANDOUTS

ECE

III/V

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit V : Input/Output and System Organization

Date of Lecture:

Topic of Lecture: RISC architecture

Introduction:

RISC stands for **Reduced Instruction Set Computer**. It is designed to reduce the execution time by simplifying the instruction set of the computer. Using RISC processors, each instruction requires only one clock cycle to execute results in uniform execution time.

Prerequisite knowledge for Complete understanding and learning of Topic:

Microprocessors and Microcontrollers

Reduced Instruction Set Computers:

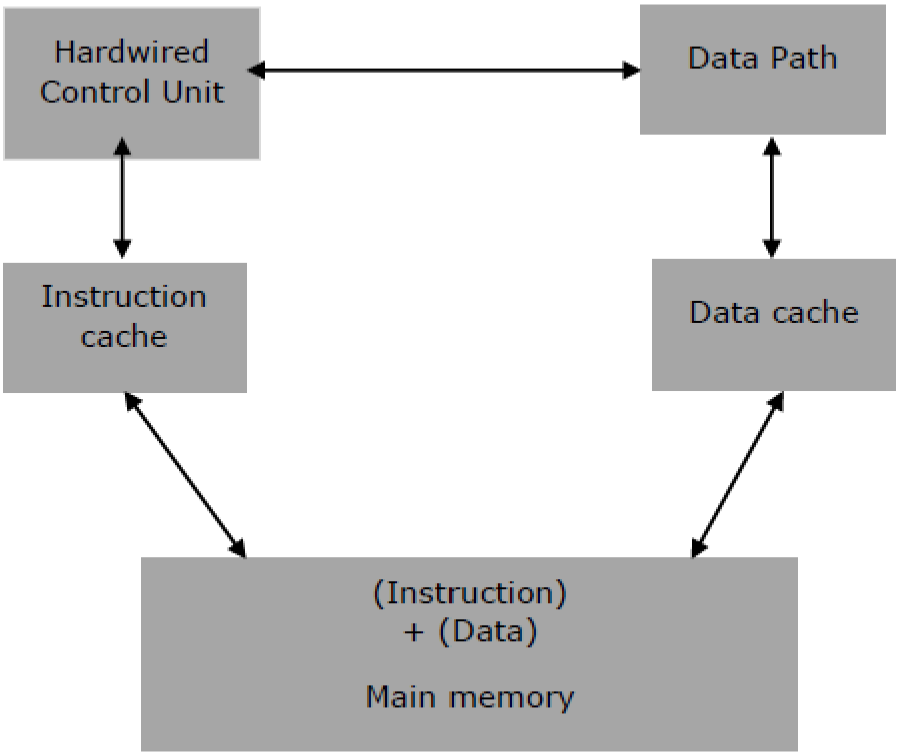
RISC stands for **Reduced Instruction Set Computer**. It is designed to reduce the execution time by simplifying the instruction set of the computer. Using RISC processors, each instruction requires only one clock cycle to execute results in uniform execution time. This reduces the efficiency as there are more lines of code, hence more RAM is needed to store the instructions. The compiler also has to work more to convert high-level language instructions into machine code.

Some of the RISC processors are −

- Power PC: 601, 604, 615, 620
- DEC Alpha: 210642, 211066, 21068, 21164
- MIPS: TS (R10000) RISC Processor
- PA-RISC: HP 7100LC

Architecture of RISC

RISC microprocessor architecture uses highly-optimized set of instructions. It is used in portable devices like Apple iPod due to its power efficiency.



Characteristics of RISC

The major characteristics of a RISC processor are as follows −

- It consists of simple instructions.
- It supports various data-type formats.
- It utilizes simple addressing modes and fixed length instructions for pipelining.
- It supports register to use in any context.
- One cycle execution time.
- "LOAD" and "STORE" instructions are used to access the memory location.
- It consists of larger number of registers.
- It consists of less number of transistors.

Video Content / Details of website for further learning (if any):
https://www.youtube.com/watch?v=fL81WScLwl0

Important Books/Journals for further learning including the page nos.:
John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no. : 577 to 579

Course Faculty

Verified by HOD

L - 45

LECTURE HANDOUTS

ECE **III/V**

Course Name with Code : 19ECE24/ COMPUTER ARCHITECTURE AND ORGANIZATION

Course Faculty : P.Padmaloshani

Unit V : Input/Output and System Organization Date of Lecture:

Topic of Lecture: CISC architecture

Introduction:

CISC stands for **Complex Instruction Set Computer**. It is designed to minimize the number of instructions per program, ignoring the number of cycles per instruction. The emphasis is on building complex instructions directly into the hardware.

Prerequisite knowledge for Complete understanding and learning of Topic:

Microprocessors and Microcontrollers

CISC architecture:

CISC stands for **Complex Instruction Set Computer**. It is designed to minimize the number of instructions per program, ignoring the number of cycles per instruction. The emphasis is on building complex instructions directly into the hardware.

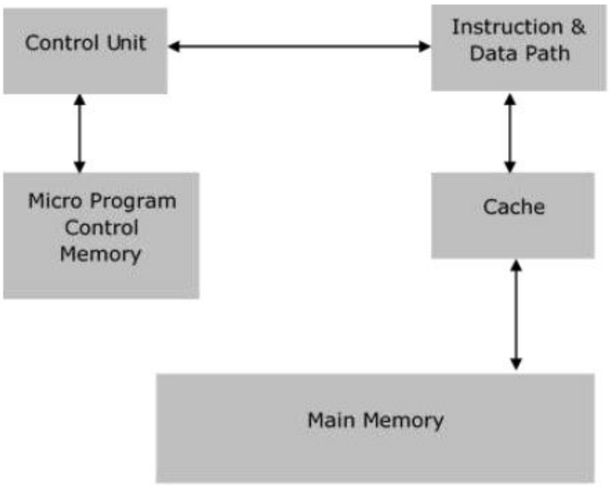
The compiler has to do very little work to translate a high-level language into assembly level language/machine code because the length of the code is relatively short, so very little RAM is required to store the instructions.

Some of the CISC Processors are −

- IBM 370/168
- VAX 11/780
- Intel 80486

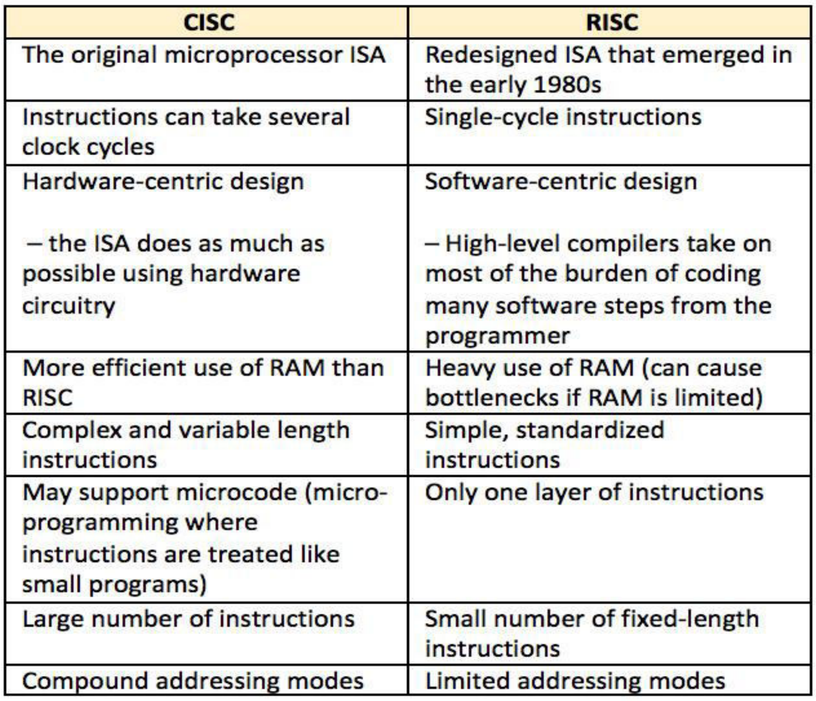
Architecture of CISC

Its architecture is designed to decrease the memory cost because more storage is needed in larger programs resulting in higher memory cost. To resolve this, the number of instructions per program can be reduced by embedding the number of operations in a single instruction.



Characteristics of CISC

- Variety of addressing modes.
- Larger number of instructions.
- Variable length of instruction formats.
- Several cycles may be required to execute one instruction.
- Instruction-decoding logic is complex.
- One instruction is required to support multiple addressing modes.



CISC	RISC
The original microprocessor ISA	Redesigned ISA that emerged in the early 1980s
Instructions can take several clock cycles	Single-cycle instructions
Hardware-centric design – the ISA does as much as possible using hardware circuitry	Software-centric design – High-level compilers take on most of the burden of coding many software steps from the programmer
More efficient use of RAM than RISC	Heavy use of RAM (can cause bottlenecks if RAM is limited)
Complex and variable length instructions	Simple, standardized instructions
May support microcode (micro-programming where instructions are treated like small programs)	Only one layer of instructions
Large number of instructions	Small number of fixed-length instructions
Compound addressing modes	Limited addressing modes

Video Content / Details of website for further learning (if any):
https://www.youtube.com/watch?v=fL81WScLwl0

Important Books/Journals for further learning including the page nos.:
John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill, 3rd edition, 1998, page no. :580 to 581

Course Faculty

Verified by HOD