UNIT 6

CONTROL UNIT

Control Unit and Peripheral Devices

- Control Unit- Micro Operations,
- Hardwired Implementations,
- Micro Programmed control,
- Micro instruction format and applications of microprogramming,
- I/O modules- Programmed I/O,
- I/O modules-Interrupt Driven I/O,
- DMA.
- I/O processors and channels,
- General- Purpose Graphics Processing Unit,
- GPU applications, synchronization, coherence.

Design of Control Unit

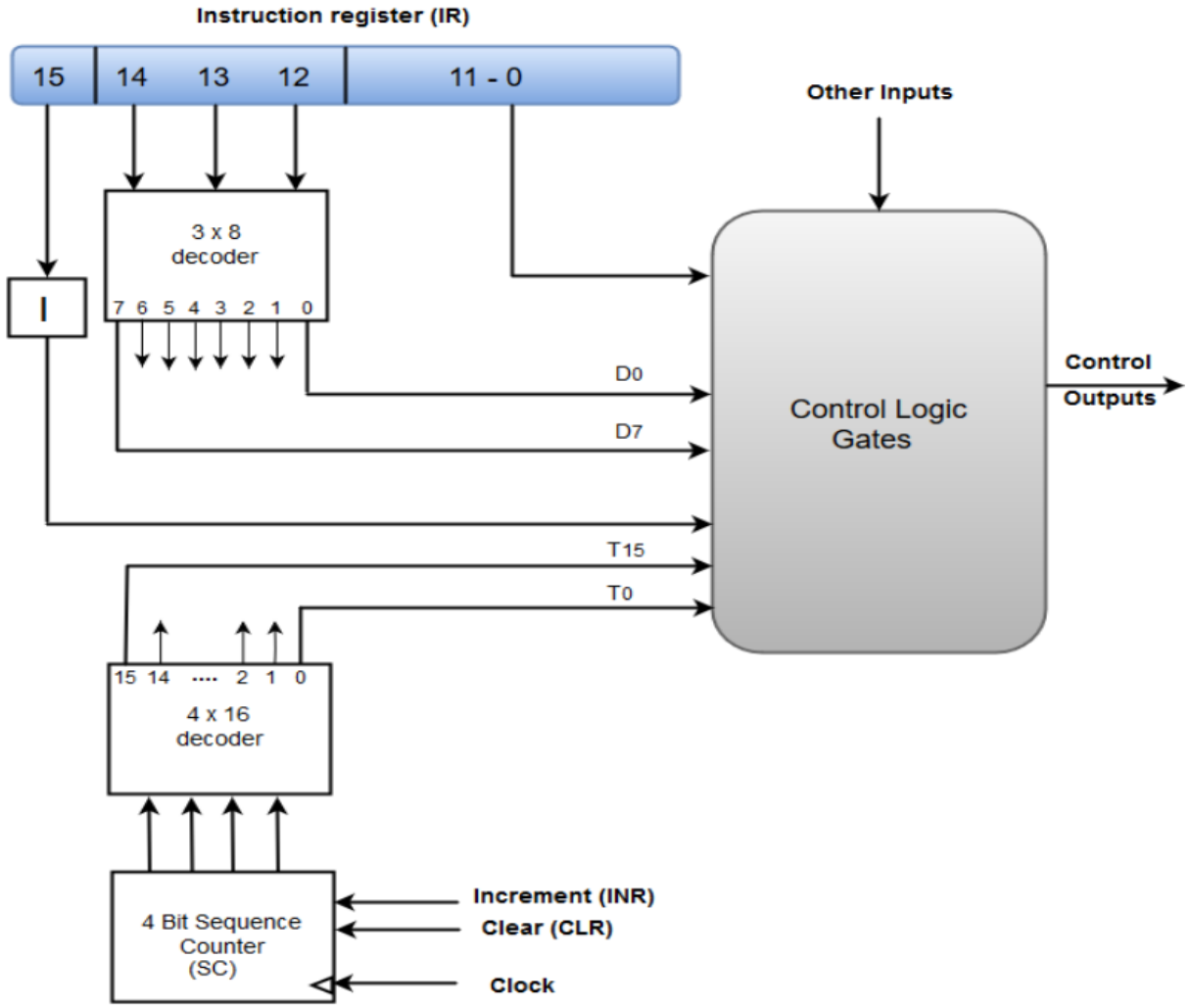
- Control unit generates timing and control signals for the operations of the computer. The control unit communicates with ALU and main memory. It also controls the transmission between processor, memory and the various peripherals. It also instructs the ALU which operation has to be performed on data.

- The Control Unit is classified into two major categories:
- Hardwired Control
- Microprogrammed Control

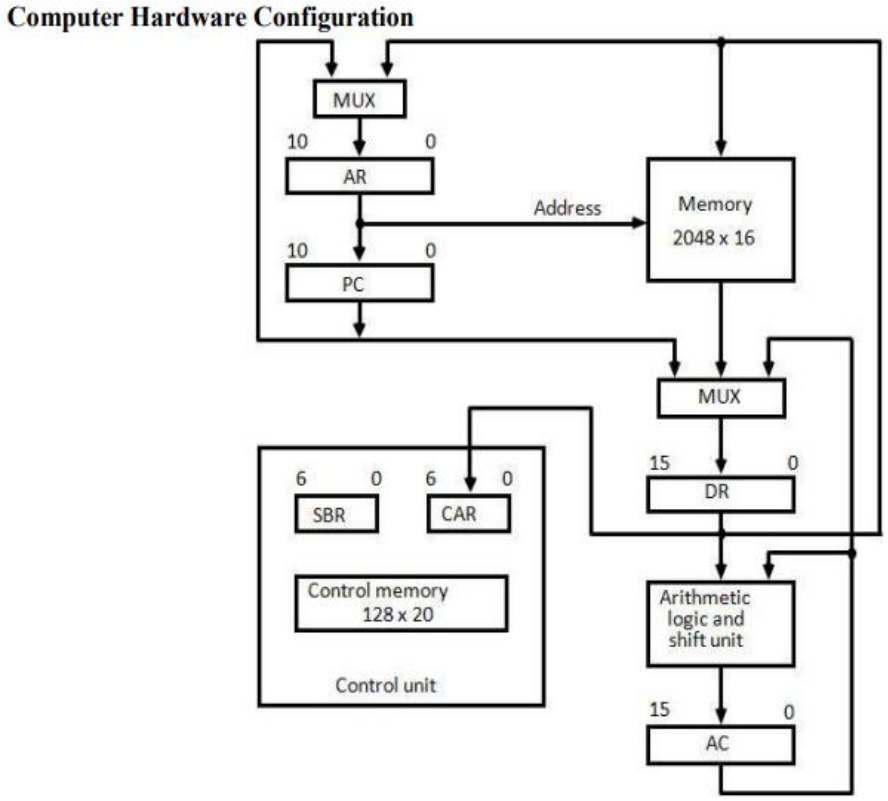
Hardwired Control Unit

- The Hardwired Control organization involves the control logic to be implemented with gates, flip-flops, decoders, and other digital circuits.

- The following image shows the block diagram of a Hardwired Control organization.



Hardwired configuration



The block diagram of the computer is shown in Figure 4.4.
It consists of

1. **Two memory units:**

Main memory - for storing instructions and data, and
Control memory -for storing the microprogram.

2. **Six Registers:**

Processor unit register: AC(accumulator),PC(Program Counter), AR(Address Register), DR(Data Register) Control unit register: CAR (Control Address Register), SBR(Subroutine Register)

3. **Multiplexers:**

The transfer of information among the registers in the processor is done through multiplexers rather than a common bus.

4. **ALU:** The arithmetic, logic, and shift unit performs microoperations with data from AC and DR and places the result in AC. 8 UNIT -III Unit 4 – Microprogrammed Control DR can receive information from AC, PC, or memory.

5. AR can receive information from PC or DR.
   PC can receive information only from AR.

6. Input data written to memory come from DR, and data read from memory can go only to DR.

Hardwired Control Unit

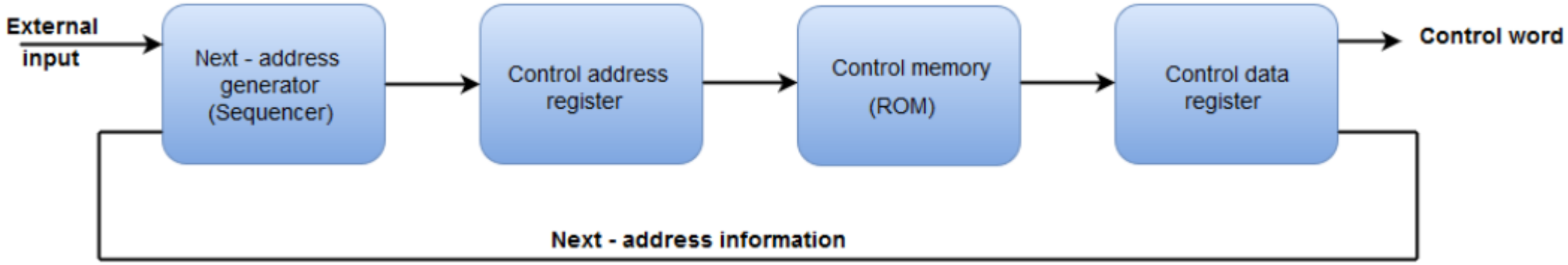
- A Hard-wired Control consists of two decoders, a sequence counter, and a number of logic gates.
- An instruction fetched from the memory unit is placed in the instruction register (IR).
- The component of an instruction register includes; I bit, the operation code, and bits 0 through 11.
- The operation code in bits 12 through 14 are coded with a 3 x 8 decoder.
- The outputs of the decoder are designated by the symbols D0 through D7.
- The operation code at bit 15 is transferred to a flip-flop designated by the symbol I.
- The operation codes from Bits 0 through 11 are applied to the control logic gates.
- The Sequence counter (SC) can count in binary from 0 through 15

Microprogrammed Control
Microoperation, Microinstruction, Micro program, Microcode

- **Microoperations:**
- In computer central processing units, micro-operations (also known as a micro-ops or μops) are detailed low-level instructions used in some designs to implement complex machine instructions (sometimes termed macro-instructions in this context).
- **Micro instruction:**
- A symbolic microprogram can be translated into its binary equivalent by means of an assembler.
- Each line of the assembly language microprogram defines a symbolic microinstruction.
- Each symbolic microinstruction is divided into five fields: label, microoperations, CD, BR, and AD.
- **Microoperations:**
- In computer central processing units, micro-operations (also known as a micro-ops or μops) are detailed low-level instructions used in some designs to implement complex machine instructions (sometimes termed macro-instructions in this context).
- **Micro instruction:** A symbolic microprogram can be translated into its binary equivalent by means of an assembler.
- Each line of the assembly language microprogram defines a symbolic microinstruction.
- Each symbolic microinstruction is divided into five fields: label, microoperations, CD, BR, and AD.

Microprogrammed Control

- The Microprogrammed Control organization is implemented by using the programming approach
- In Microprogrammed Control, the micro-operations are performed by executing a program consisting of micro-instructions.
- The following image shows the block diagram of a Microprogrammed Control organization.



Microprogrammed Control

- **The control memory address** register specifies the address of the microinstruction, and the control data register holds the microinstruction read from memory.
- The microinstruction contains a **control word** that specifies one or more microoperations for the data processor. Once these operations are executed, the control must determine the next address.
- The location of the next microinstruction may be the one next in sequence, or it may be located somewhere else in the control memory.
- While the microoperations are being executed, the next address is computed in the next address generator circuit and then transferred into the control address register to read the next microinstruction.
- Thus a microinstruction contains bits for initiating microoperations in the data processor part and bits that determine the address sequence for the control memory.
- The next address generator is sometimes called a micro-program sequencer, as it determines the address sequence that is read from control memory.

Microprogrammed Control

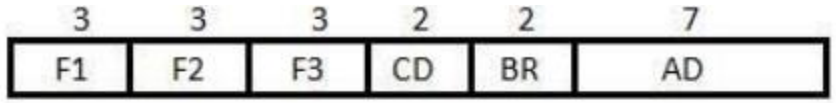
- Typical functions of a micro-program sequencer are incrementing the control address register by one, loading into the control address register an address from control memory, transferring an external address, or loading an initial address to start the control operations.
- The control data register holds the present microinstruction while the next address is computed and read from memory.
- The data register is sometimes called a pipeline register.
- It allows the execution of the microoperations specified by the control word simultaneously with the generation of the next microinstruction.
- This configuration requires a two-phase clock, with one clock applied to the address register and the other to the data register.
- The main advantage of the micro programmed control is the fact that once the hardware configuration is established; there should be no need for further hardware or wiring changes.
- If we want to establish a different control sequence for the system, all we need to do is specify a different set of microinstructions for control memory

Microprogrammed Control

- The Control memory address register specifies the address of the micro-instruction.
- The Control memory is assumed to be a ROM, within which all control information is permanently stored.
- The control register holds the microinstruction fetched from the memory.
- The micro-instruction contains a control word that specifies one or more micro-operations for the data processor.
- While the micro-operations are being executed, the next address is computed in the next address generator circuit and then transferred into the control address register to read the next microinstruction.
- The next address generator is often referred to as a micro-program sequencer, as it determines the address sequence that is read from control memory.

Micro instruction format and applications of microprogramming

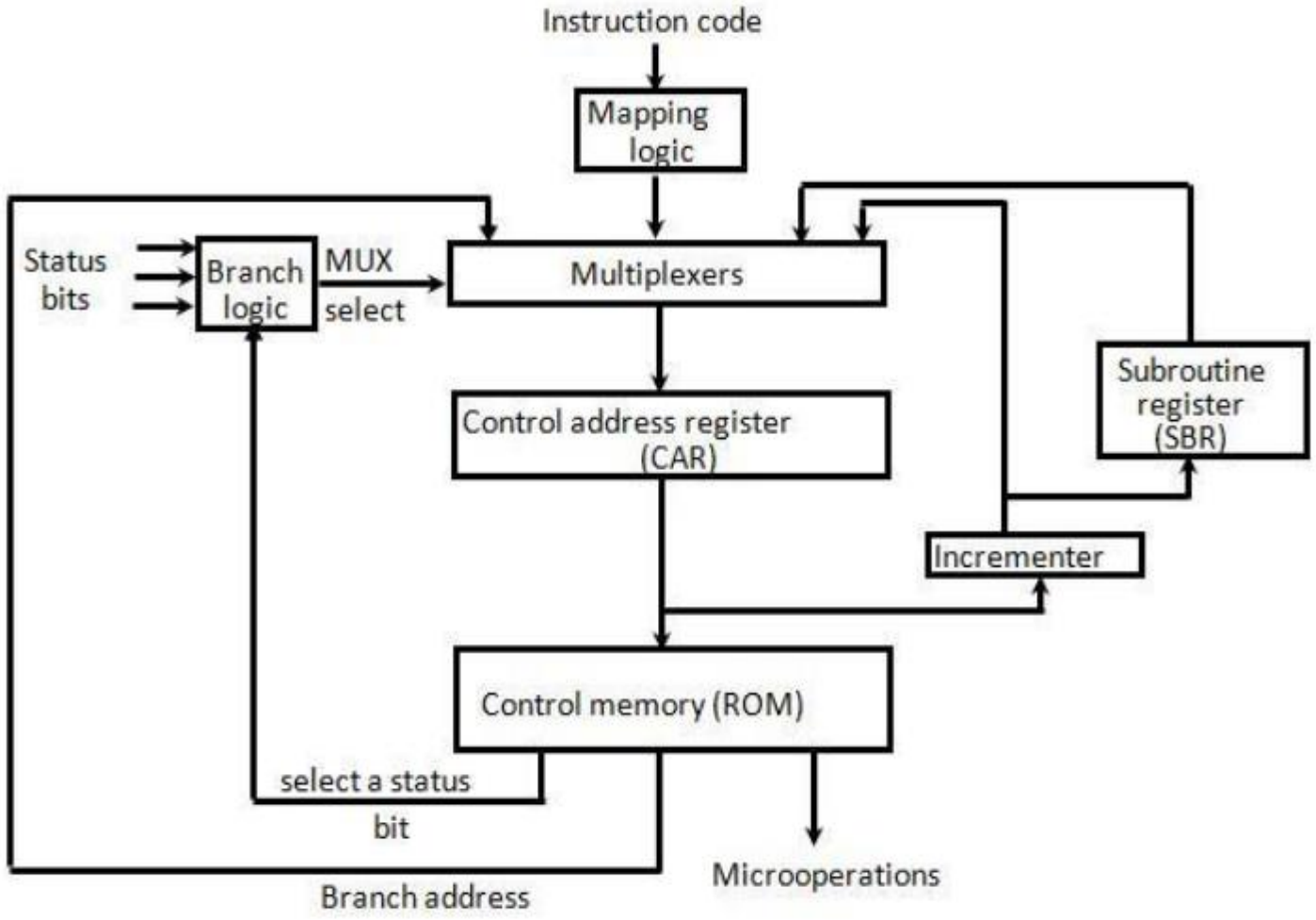
- The microinstruction format for the control memory is shown in figure 4.5. The 20 bits of the microinstruction are divided into four functional parts as follows:
- 1. The three fields F1, F2, and F3 specify microoperations for the computer. The microoperations are subdivided into three fields of three bits each. The three bits in each field are encoded to specify seven distinct micro operations. This gives a total of 21 microoperations.
- 2. The CD field selects status bit conditions.
- 3. The BR field specifies the type of branch to be used.
- 4. The AD field contains a branch address. The address field is seven bits wide, since the control memory has 128 = 27 words.



F1, F2, F3: Microoperation fields
CD: Condition for branching
BR: Branch field
AD: Address field

Figure 4.5: Microinstruction Format

Selection of address for control memory



The address sequencing capabilities required in a control memory are:

1. Incrementing of the control address register.
2. Unconditional branch or conditional branch, depending on status bit conditions.
3. A mapping process from the bits of the instruction to an address for control memory.
4. A facility for subroutine call and return.

Selection of address for control memory

- Above figure 4.2 shows a block diagram of a control memory and the associated hardware needed for selecting the next microinstruction address.
- The microinstruction in control memory contains a set of bits to initiate microoperations in computer registers and other bits to specify the method by which the next address is obtained.
- The diagram shows four different paths from which the control address register (CAR) receives the address.
- The incrementer increments the content of the control address register by one, to select the next microinstruction in sequence.
- Branching is achieved by specifying the branch address in one of the fields of the microinstruction.
- Conditional branching is obtained by using part of the microinstruction to select a specific status bit in order to determine its condition.

Selection of address for control memory

- An external address is transferred into control memory via a mapping logic circuit.
- The return address for a subroutine is stored in a special register whose value is then used when the micro-program wishes to return from the subroutine. 6 UNIT -III Microprogrammed Control
- The branch logic of figure 4.2 provides decision-making capabilities in the control unit.
- The status conditions are special bits in the system that provide parameter information such as the carry-out of an adder, the sign bit of a number, the mode bits of an instruction, and input or output status conditions.
- The status bits, together with the field in the microinstruction that specifies a branch address, control the conditional branch decisions generated in the branch logic.
- A 1 output in the multiplexer generates a control signal to transfer the branch address from the microinstruction into the control address register.
- A 0 output in the multiplexer causes the address register to be incremented

I/O modules- Programmed I/O

- This I/O technique is the simplest to exchange data between external devices and processors. In this technique, the processor or Central Processing Unit (CPU) runs or executes a program giving direct control of I/O operations.
- Processor issues a command to the I/O module and waits for the operation to complete. Also, the processor keeps checking the I/O module status until it finds the completion of the operation.
- The processor's time is wasted, as the processor is faster than the I/O module. I/O module is considered to be a slow module.
- Its application is in certain low-end microcomputers. It has a single output and single input instruction.
- Each one of the instruction selects only one I/O device by number and transfers only a single character by byte.
- Four registers are involved in this technique and they are output status and character and input status and character.

I/O modules- Programmed I/O…..

- Its disadvantage is busy waiting which means the processor consumes most of its time in a tight loop by waiting for the I/O device to be ready to be used. Program checks or polls an I/O hardware component, device, or item.
- **For Example** – A computer mouse that is within a loop.
- It is easy to understand. It is easy to program. It is slow and inefficient.
- The system's performance is degraded, severely. It does not require initializing the stack.
- System's throughput is decreased due to the increase in the number of I/O devices connected in the system. The best example is that of the PC device Advanced Technology Attachment (ATA) interface using programmed I/O.

I/O modules-Interrupt Driven I/O

- It is similar to the programmed-driven I/O technique. The processor does not wait until the I/O operation is completed. The processor performs other tasks while the I/O operation is being performed.
- When the I/O operation is completed, the I/O module interrupts the processor letting the processor know the operation is completed. Its module is faster than the programmed I/O module.
- The processor actually starts the I/O device and instructs it to generate and send an interrupt signal when the operation is finished. This is achieved by setting an interruptenabled bit in the status register.
- This technique requires an interrupt for each character that is written or read. It is an expensive business to interrupt a running process as it requires saving context.
- It requires additional hardware such as a Direct Memory Access (DMA) controller chip. It is fast and efficient.
- It becomes difficult to code, in case the programmer is using a low-level programming language. It can get difficult to get the various pieces to be put to work well together. This is done by the OS developer, for example, Microsoft or the hardware manufacturer.
- The system's performance is enhanced. It requires initializing the stack.
- The system's throughput is not affected despite the number of I/O devices connected in the system increasing as the throughput does not rely on the number.
- **For Example** − The computer mouse triggers and sends a signal to the program for processing the mouse event.
- Interrupt-driven I/O is better as it is fast, efficient. The system's performance is improved and enhanced

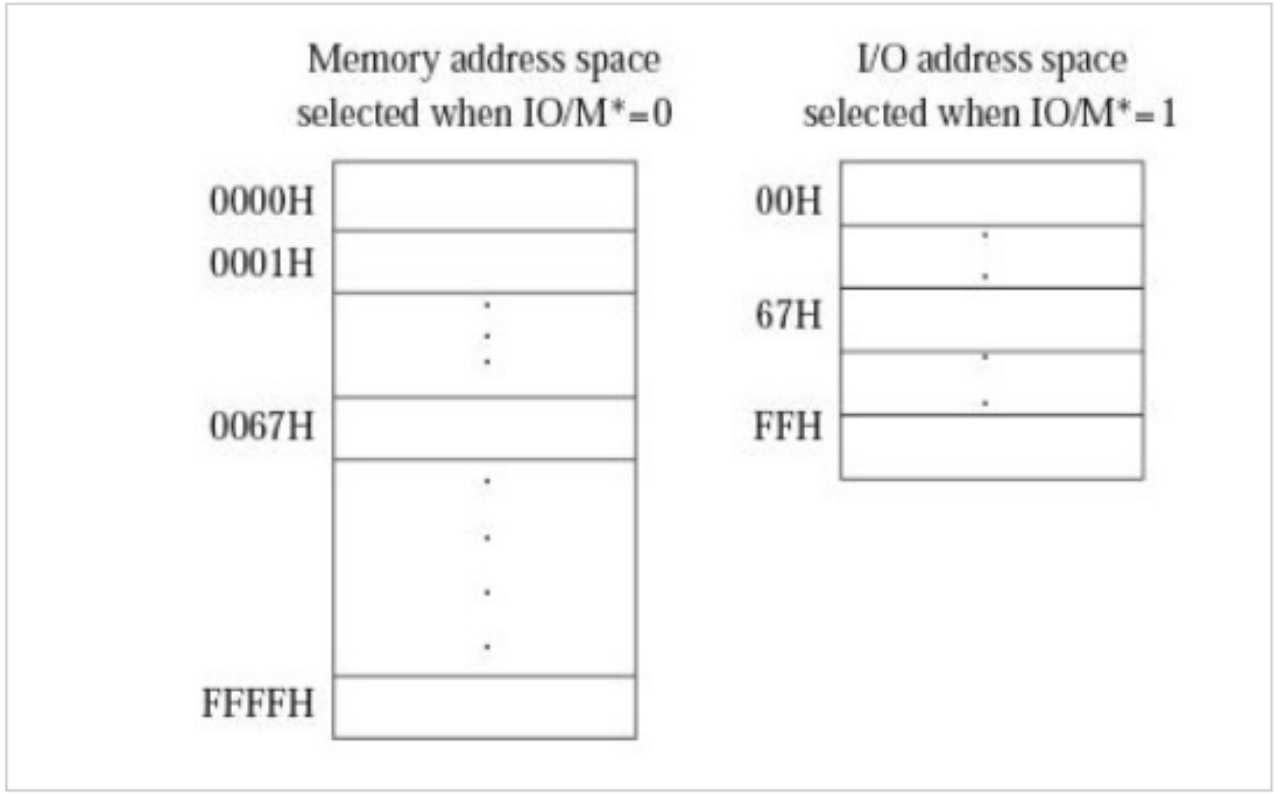
Memory Mapped I/o and I/O mapped IO

- In Memory Mapped Input Output –
- We allocate a memory address to an Input-Output device.
- Any instructions related to memory can be accessed by this Input-Output device.
- The Input-Output device data are also given to the Arithmetic Logical Unit.
- Input-Output Mapped Input Output –
- We give an Input-Output address to an Input-Output device.
- Only IN and OUT instructions are accessed by such devices.
- The ALU operations are not directly applicable to such Input-Output data.

Memory Mapped I/o and I/O mapped IO

- I/O is any general-purpose port used by processor/controller to handle peripherals connected to it.
- I/O mapped I/Os have a separate address space from the memory. So, total addressed capacity is the number of I/Os connected and a memory connected. Separate I/O-related instructions are used to access I/Os. A separate signal is used for addressing an I/O device.
- Memory-mapped I/Os share the memory space with external memory. So, total addressed capacity is memory connected only. This is underutilisation of resources if your processor supports I/O-mapped I/O. In this case, instructions used to access I/Os are the same as that used for memory.
- Let's take an example of the 8085 processor. It has 16 address lines i.e. addressing capacity of 64 KB memory. It supports I/O-mapped I/Os. It can address up to 256 I/Os.
- If we connect I/Os to it an I/O-mapped I/O then, it can address 256 I/Os + 64 KB memory. And special instructions IN and OUT are used to access the peripherals. Here we fully utilize the addressing capacity of the processor.
- If the peripherals are connected in memory mapped fashion, then total devices it can address is only 64K. This is underutilisation of the resource. And only memory-accessing instructions like MVI, MOV, LOAD, SAVE are used to access the I/O devices.
-

Memory Mapped I/o and I/O mapped IO



DMA (Direct Memory Access)

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

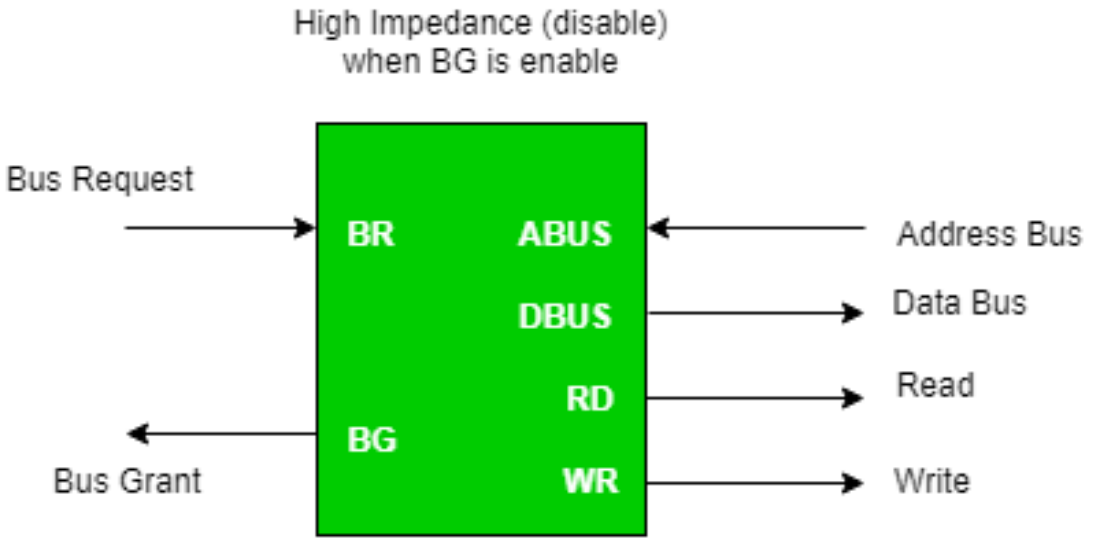


Figure - CPU Bus Signals for DMA Transfer

DMA (Direct Memory Access)

- **Bus Request :** It is used by the DMA controller to request the CPU to relinquish the control of the buses.
- **Bus Grant :** It is activated by the CPU to Inform the external DMA controller that the buses are in high impedance state and the requesting DMA can take control of the buses. Once the DMA has taken the control of the buses it transfers the data. This transfer can take place in many ways.
- **Types of DMA transfer using DMA controller:**
- **Burst Transfer :**
  DMA returns the bus after complete data transfer. A register is used as a byte count, being decremented for each byte transfer, and upon the byte count reaching zero, the DMAC will release the bus. When the DMAC operates in burst mode, the CPU is halted for the duration of the data transfer.
- Steps involved are:
  - Bus grant request time.
  - Transfer the entire block of data at transfer rate of device because the device is usually slow than the speed at which the data can be transferred to CPU.
  - Release the control of the bus back to CPU
    So, total time taken to transfer the N bytes
    = Bus grant request time + (N) * (memory transfer rate) + Bus release control time.

DMA (Direct Memory Access)

```
Where,
X µsec =data transfer time or preparation time (words/block)
Y µsec =memory cycle time or cycle time or transfer time (words/block)
% CPU idle (Blocked)=(Y/X+Y)*100
% CPU Busy=(X/X+Y)*100
```

DMA (Direct Memory Access)

- **Cyclic Stealing :**
- An alternative method in which DMA controller transfers one word at a time after which it must return the control of the buses to the CPU. The CPU delays its operation only for one memory cycle to allow the direct memory I/O transfer to “steal” one memory cycle.
  Steps Involved are:
  - Buffer the byte into the buffer
  - Inform the CPU that the device has 1 byte to transfer (i.e. bus grant request)
  - Transfer the byte (at system bus speed)
  - Release the control of the bus back to CPU.
- Before moving on transfer next byte of data, device performs step 1 again so that bus isn’t tied up and
  the transfer won’t depend upon the transfer rate of device.
  So, for 1 byte of transfer of data, time taken by using cycle stealing mode (T).
  = time required for bus grant + 1 bus cycle to transfer data + time required to release the bus, it will be
  N x T

DMA (Direct Memory Access)

- In **cycle stealing** mode we always follow pipelining concept that when one byte is getting transferred then Device is parallel preparing the next byte.
- “The fraction of CPU time to the data transfer time” if asked then cycle stealing mode is used.

```
Where,
X µsec =data transfer time or preparation time
(words/block)
Y µsec =memory cycle time or cycle time or transfer
time (words/block)
% CPU idle (Blocked) =(Y/X)*100
% CPU busy=(X/Y)*100
```

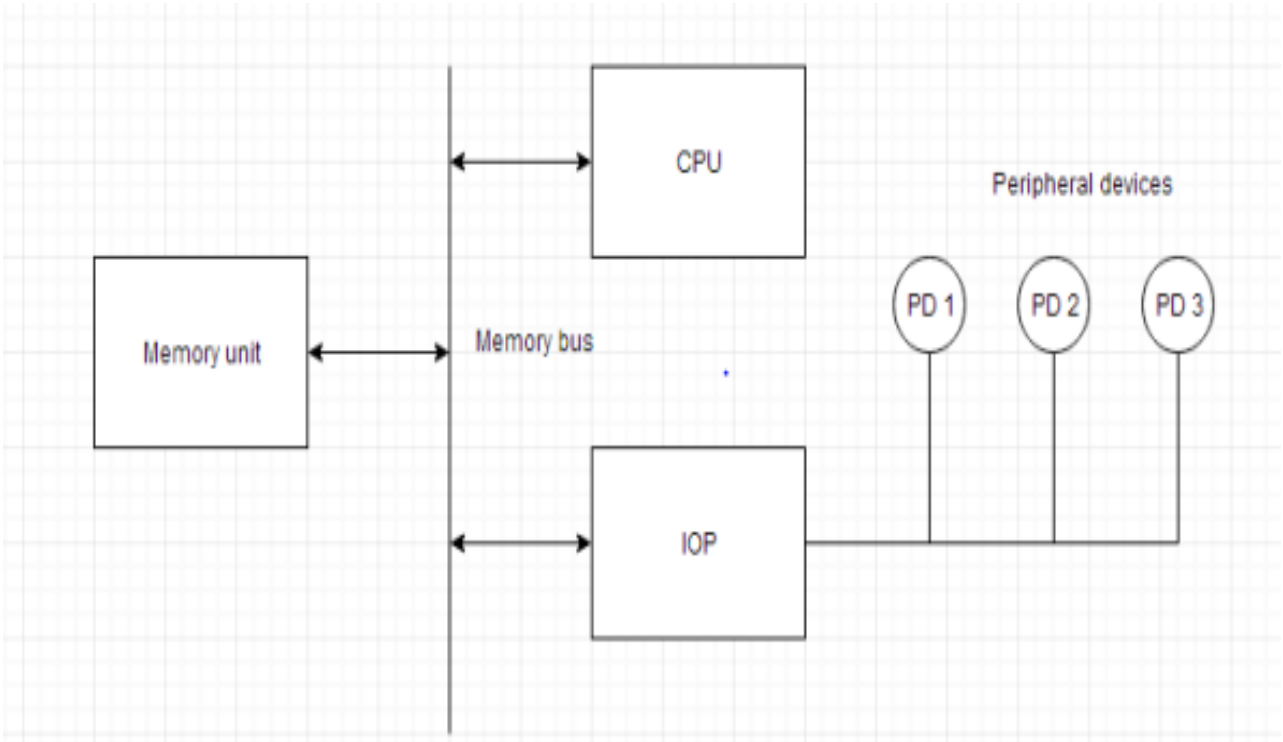
Interleaved mode:
In this technique , the DMA controller takes over the system bus when the microprocessor is not using it.An alternate half cycle i.e. half cycle DMA + half cycle processor.

I/O processors and channels

- **Input / Output Processor**
- 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.
- Each IOP controls and manage the input-output tasks.
- 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.
- These IOP instructions are designed to manage I/O transfers only.

Block Diagram Of I/O Processor

- Below is a block diagram of a computer along with various I/O Processors.
- The memory unit occupies the central position and can communicate with each processor.
- The CPU processes the data required for solving the computational tasks.
- The IOP provides a path for transfer of data between peripherals and memory.
- The CPU assigns the task of initiating the I/O program.
- The IOP operates independent from CPU and transfer data between peripherals and memory.



I/O Processor

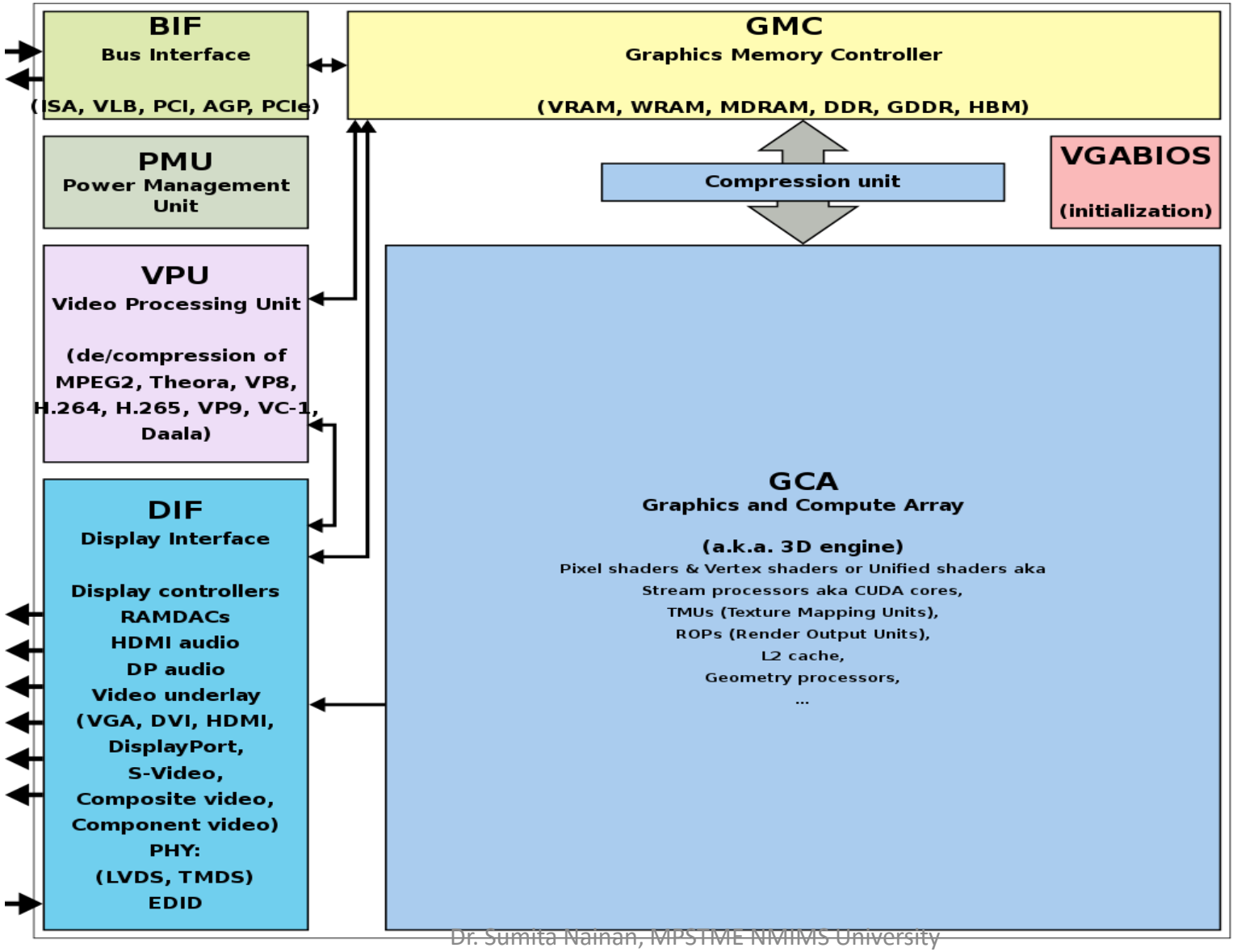
- The communication between the IOP and the devices is similar to the program control method of transfer.
- And the communication with the memory is similar to the direct memory access method.
- In large scale computers, each processor is independent of other processors and any processor can initiate the operation.
- The CPU can act as master and the IOP act as slave processor.
- The CPU assigns the task of initiating operations but it is the IOP, who executes the instructions, and not the CPU.
- CPU instructions provide operations to start an I/O transfer.
- The IOP asks for CPU through interrupt.
- Instructions that are read from memory by an IOP are also called commands to distinguish them from instructions that are read by CPU.
- Commands are prepared by programmers and are stored in memory.
- Command words make the program for IOP.
- CPU informs the IOP where to find the commands in memory.

General- Purpose Graphics Processing Unit

- ***A General-Purpose Graphics Processing Unit (GPGPU) is a graphics processing unit (GPU) that is programmed for purposes beyond graphics processing, such as performing computations typically conducted by a Central Processing Unit (CPU).***
- GPGPU stands for “general-purpose computing on graphics processing units”, or “general-purpose graphics processing units” for short.
- The idea is to leverage the power of GPUs, which are conventionally used for generating computer graphics, to carry out tasks that were traditionally done by central processing units (CPU).
- The combined use of these two kinds of processors is sometimes referred to as heterogeneous computing; it is a key factor in a lot of technological breakthroughs, such as the development of artificial intelligence through machine learningand deep learning.

  GPGPUs excel at parallel computing due to their large number of cores, which operate at lower frequencies than CPUs, but are more suited for data that's in graphical form. Groundbreaking scientific research can benefit greatly from GPGPUs, and many high performance computing (HPC) servers utilize a large number of GPGPUs to reach the level of supercomputing.
- GPGPUs are used for tasks that were formerly the domain of high-power CPUs, such as physics calculations, encryption/decryption, scientific computations and the generation of cypto currencies such as Bitcoin.
- Because graphics cards are constructed for massive parallelism, they can dwarf the calculation rate of even the most powerful CPUs for many parallel processing tasks. T
- The same shader cores that allow multiple pixels to be rendered simultaneously can similarly process multiple streams of data at the same time.
- Although a shader core is not nearly as complex as a CPU, a high-end GPU may have thousands of shader cores; in contrast, a multicore CPU might have eight or twelve cores.

General- Purpose Graphics Processing Unit



General- Purpose Graphics Processing Unit

- There has been an increased focus on GPGPUs since DirectX 10 included unified shaders in its shader core specifications for Windows Vista.
- Higher-level languages are being developed all the time to ease programming for computations on the GPU.
- Both AMD/ATI and Nvidia have approaches to GPGPU with their own APIs (OpenCL and CUDA, respectively).

General Purpose GPUs

- **The history of general-purpose GPUs**
- Nvidia's GeForce 3 was the first GPU that featured programmable shaders.
- At that time, the purpose was making rasterized 3D graphics more realistic; the new GPU capabilities enabled 3D transform, bump mapping, specular mapping and lighting computations.
- ATI's 9700 GPU, the first DirectX 9-capable card approached the programming flexibility of CPUs, although few general purpose calculations were done at the time.
- With the introduction of Windows Vista, bundled with DirectX 10, unified shader cores were specified as part of the standard.
- GPU's new-found potential demonstrated performance increases several orders of magnitude over CPU-based calculations.
- **GPGPUs and the future of computer graphics**
- GPUs that were originally developed to speed rasterized 3D (as raytracing was too expensive calculation-wise) have surpassed the performance of CPUs for ray traced pre-rendered graphics.
- Although raytracing is not yet used in games, there have been real-time demonstrations.
- The advances of GPGPUs mean that in the not-too-distant future, computer graphics should be capable of the same kind of intensive geometry and lighting as 3D movies.

GPU applications, synchronization, coherence

- It goes without saying that if the data you work with is already in graphical form—as would be the case in projects that involve computer vision, like the development of autonomous vehicles or facial recognition systems—you would do well to have GPGPUs in your servers. Other possible applications include:

  Earth science: Whether it's meteorological or astronomical research, the addition of GPGPUs can speed up the process considerably. For example, Japan's Waseda University used GIGABYTE's G221-Z30 to assemble a computing cluster that can run simulations on how climate change will affect coastal regions. Lowell Observatory in Arizona, USA uses GIGABYTE's G482-Z50 to filter out "stellar noise" in the search for habitable exoplanets.

  Scientific knowledge: Academic institutes and research centers are always pushing the envelope of human understanding. GPU Servers outfitted with GPGPUs can accelerate the effort. CERN, the European Organization for Nuclear Research, uses GIGABYTE's G482-Z51 to analyze data produced by the Large Hadron Collider (LHC) in search of the elusive "beauty" quark. The Institute of Theoretical and Computational Chemistry at the University of Barcelona expanded the power of its data center by about 40% with GIGABYTE's G292-Z42 and other servers.

  A better tomorrow: As the world's brightest minds pit advanced processing power against problems faced by humanity, servers equipped with GPGPUs have a clear role to play. IFISC, Spain's Institute for Cross-Disciplinary Physics and Complex Systems, uses GIGABYTE's G482-Z54 for important research that covers climate change, green energy, and ways of combating COVID-19. The National Taiwan Normal University has built a Center for Cloud Computing with G190-H44 and other servers. As long as the data can be converted to graphical form, GPGPUs are a natural fit in the server solution.

GPU_extra

- What Does a GPU Do?
- The graphics processing unit, or GPU, has become one of the most important types of computing technology, both for personal and business computing. Designed for parallel processing, the GPU is used in a wide range of applications, including graphics and video rendering. Although they're best known for their capabilities in gaming, GPUs are becoming more popular for use in creative production and artificial intelligence (AI).
- GPUs were originally designed to accelerate the rendering of 3D graphics. Over time, they became more flexible and programmable, enhancing their capabilities. This allowed graphics programmers to create more interesting visual effects and realistic scenes with advanced lighting and shadowing techniques. Other developers also began to tap the power of GPUs to dramatically accelerate additional workloads in high performance computing (HPC), deep learning, and more.
- GPU and CPU: Working Together
- The GPU evolved as a complement to its close cousin, the CPU (central processing unit). While CPUs have continued to deliver performance increases through architectural innovations, faster clock speeds, and the addition of cores, GPUs are specifically designed to accelerate computer graphics workloads. When shopping for a system, it can be helpful to know the role of the CPU vs. GPU so you can make the most of both.

- GPU vs. Graphics Card: What's the Difference?
- While the terms GPU and graphics card (or video card) are often used interchangeably, there is a subtle distinction between these terms. Much like a motherboard contains a CPU, a graphics card refers to an add-in board that incorporates the GPU. This board also includes the raft of components required to both allow the GPU to function and connect to the rest of the system.
- GPUs come in two basic types: integrated and discrete. An integrated GPU does not come on its own separate card at all and is instead embedded alongside the CPU. A discrete GPU is a distinct chip that is mounted on its own circuit board and is typically attached to a PCI Express slot.

- **Integrated Graphics Processing Unit**
- The majority of GPUs on the market are actually integrated graphics. So, what are integrated graphics and how does it work in your computer? A CPU that comes with a fully integrated GPU on its motherboard allows for thinner and lighter systems, reduced power consumption, and lower system costs.
- Intel® Graphics Technology, which includes Intel® Arc™ and Intel® Iris® Xᵉ graphics, is at the forefront of integrated graphics technology. With Intel® Graphics, users can experience immersive graphics in systems that run cooler and deliver long battery life.
- **Discrete Graphics Processing Unit**
- Many computing applications can run well with integrated GPUs. However, for more resource-intensive applications with extensive performance demands, a discrete GPU (sometimes called a dedicated graphics card) is better suited to the job.
- These GPUs add processing power at the cost of additional energy consumption and heat creation. Discrete GPUs generally require dedicated cooling for maximum performance.

- *Today's GPUs are more programmable than ever before, allowing a broad range of applications that go beyond traditional graphics rendering.*
- **What Are GPUs Used For?**
- Two decades ago, GPUs were used primarily to accelerate real-time 3D graphics applications, such as games. However, as the 21st century began, computer scientists realized that GPUs had the potential to solve some of the world's most difficult computing problems.
- This realization gave rise to the general purpose GPU era. Now, graphics technology is applied more extensively to an increasingly wide set of problems. Today's GPUs are more programmable than ever before, affording them the flexibility to accelerate a broad range of applications that go well beyond traditional graphics rendering.
- **GPUs for Gaming**
- Video games have become more computationally intensive, with hyperrealistic graphics and vast, complicated in-game worlds. With advanced display technologies, such as 4K screens and high refresh rates, along with the rise of virtual reality gaming, demands on graphics processing are growing fast. GPUs are capable of rendering graphics in both 2D and 3D. With better graphics performance, games can be played at higher resolution, at faster frame rates, or both.

- **GPUs for Video Editing and Content Creation**
- For years, video editors, graphic designers, and other creative professionals have struggled with long rendering times that tied up computing resources and stifled creative flow. Now, the parallel processing offered by GPUs makes it faster and easier to render video and graphics in higher-definition formats.
- When it comes to performance, Intel provides no-compromise solutions for both the CPU and GPU. With Intel® Iris® Xe graphics, gamers and content creators can now get even better performance and new capabilities. Optimized for 11th Gen Intel® Core™ processors and perfect for Ultra-thin and light laptops, Intel® Iris® Xe graphics come integrated with the processor. Select laptops also include Intel® Iris® Xe MAX, Intel's first discrete graphics product in 20 years.
- Intel® Iris® Xe MAX was designed to provide advanced graphics performance and media capabilities, as well as enjoy seamless, immersive gameplay anywhere in 1080p. All while on a sleek lightweight laptop. Additionally, by combing 11th Gen Intel® Core™ processors, Iris® Xe MAX discrete graphics, and Intel® Deep Link Technology, you can experience 1.4X AI[1] performance and 2X better performance encoding single stream videos[2] than with a 3rd party discrete graphics.[3]
- **GPU for Machine Learning**
- Some of the most exciting applications for GPU technology involve AI and machine learning. Because GPUs incorporate an extraordinary amount of computational capability, they can deliver incredible acceleration in workloads that take advantage of the highly parallel nature of GPUs, such as image recognition. Many of today's deep learning technologies rely on GPUs working in conjunction with CPUs.

Intel® GPU Technologies

Intel has long been a leader in graphics processing technology, especially when it comes to PCs.
Most recently, the Intel® Iris® Xe graphics and Intel® UHD Graphics that are integrated into our 11th Gen Intel® Core™ processors support 4K HDR,
1080p gaming, and other rich visual experiences.
For laptop users, Intel also offers the Intel® Iris® Xe MAX graphics.
With our first discrete GPU for PCs based on Intel Xe architecture, you get even more performance and new capabilities for enhanced content
creation and gaming.

GPUs in the Data Center

In the data center, Intel supports amazing visual experiences with integrated graphics in Intel® Xeon® processors.
Intel also offers a discrete option with the Intel® Server GPU. This general-purpose GPU is based on Intel® Xe Architecture, and designed to expand exponentially to provide a high-density, low-latency experience in mobile cloud gaming, media streaming, and video encoding.

- Check this out
- https://www.geeksforgeeks.org/i-o-channels-and-its-types/#:~:text=I%2FO%20Channel%20is%20an,execute%20I%2FO%20instructions%20itself.