Interrupt Controller 8259A
DMA Controller
RAID Levels

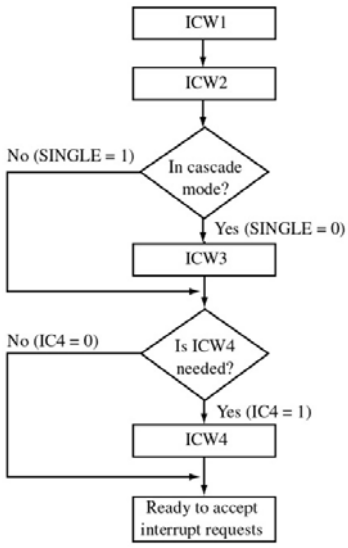
Lecture 22

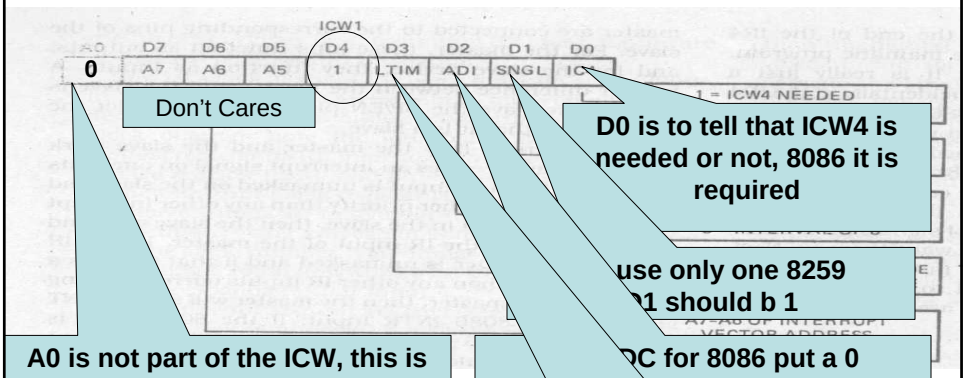
Internal Registers 8259A

- **Priority Resolver :**
  - This unit determines the priorities of the interrupt requests appearing simultaneously.
    - The highest priority is selected and stored into the corresponding bit of ISR during INTA pulse.
    - The IR0 has the highest priority while the IR7 has the lowest one, normally in fixed priority mode.
  - The priorities however may be altered by programming the 8259A in rotating priority mode.
- **• Interrupt Mask Register (IMR)** :
  - This register stores the bits required to mask the interrupt inputs.
    - IMR operates on IRR at the direction of the Priority Resolver.

ICWs

- There are four ICWs for 8259A
- When 8259 is first powered
  - ICW1 and ICW2 are sent
  - ICW3 is only needed when ICW1 is programmed for cascade operation (SNLG =1)
  - ICW4 will be programmed for 8086 operation



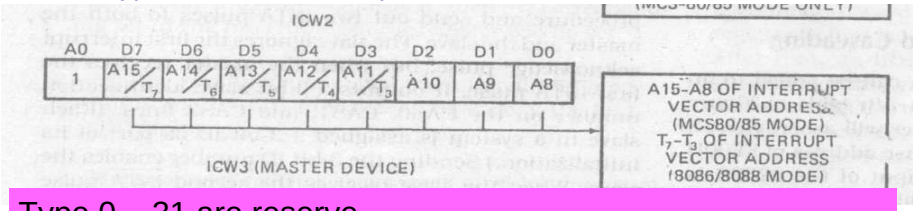


ICW1

0	0	0	0	1	0	0	1	1

ICW2

- Used to tell the 8259A the type number to send in response to an interrupt signal on the IR0 input
  - Interrupt signal to some other IR, 8259 will automatically add the number of the IR input to the base number and send it to 8086 as type number for that input

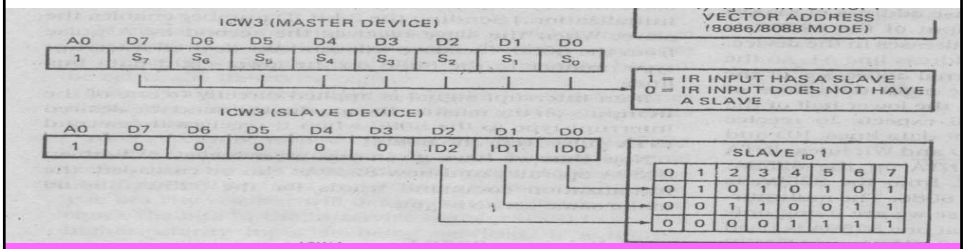


Type 0 – 31 are reserve

Type 32 (00100000h), is the lowest type number for IR0

  - Higher byte of ISR address (8085), or 8 bit vector address (8086).

ICW3



ICW3 is required if a slave is used in the system

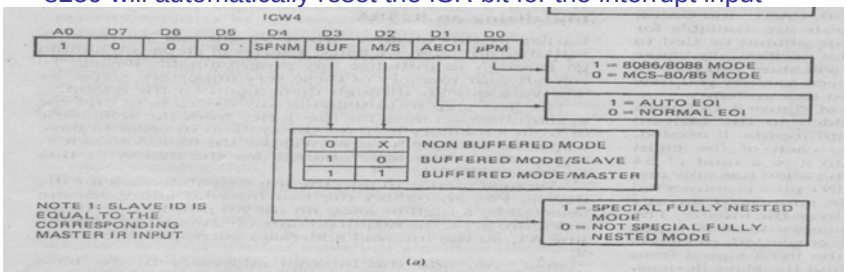
ICW 3 is required to tell the that at which IR input the slave is connected

CAS0 – CAS2 are used as three bit ID for each slave

A0=1
Master mode: 1 indicates slave is present on that interrupt, 0 indicates direct interrupt
Slave mode: ID3-ID2-ID1 is the slave ID number. Slave 4 on IR4 has ICW3=04h (0000 0100)

ICW4

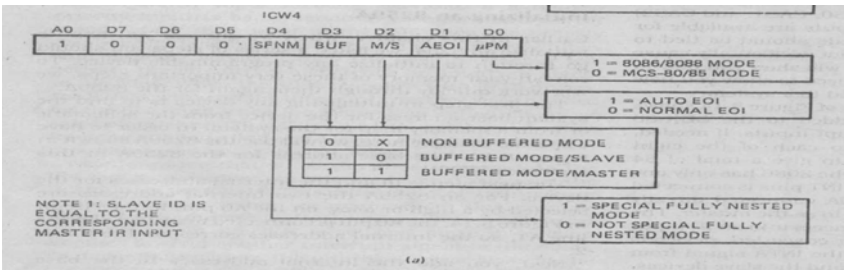
- The purpose of ICW 4 is to tell 8259 that it working is for x86 family
  - D0 as 1 for specifying 8086 system
  - D1 is used for EOI (end of interrupt) bit, if this bit is set in ICW4 the 8259 will automatically reset the ISR bit for the interrupt input



  - SFNM: 1=Special Fully Nested Mode, 0=FNM
  - M/S: 1=Master, 0=Slave
  - AEOI: 1=Auto End of Interrupt, 0=Normal
  - Mode: 0=8085, 1=8086

ICW4

- The purpose of ICW 4 is to tell 8259 that it working is for x86 family
  - D0 as 1 for specifying 8086 system
  - D1 is used for EOI (end of interrupt) bit, if this bit is set in ICW4 the 8259 will automatically reset the ISR bit for the interrupt input



A0=1
SFNM: 1=Special Fully Nested Mode, 0=FNM
M/S: 1=Master, 0=Slave
AEOI: 1=Auto End of Interrupt, 0=Normal
Mode: 0=8085, 1=8086

Operating Modes of 8259

- The 8259A can be programmed in different modes by setting or resting the appropriate bits of the ICW .
- The different modes of operation of 8259A are explained as the following.
  - **Fully Nested Mode** :
    - This is the default mode of operation of 8259A.
      - IR0 has the highest priority and IR7 has the lowest one.
    - When interrupt request are noticed, the highest priority request is determined and the vector is placed on the data bus.
      - The corresponding bit of ISR is set and remains set till the microprocessor issues an EOI command

Modes of Operation

- **Special Fully Nested Mode** :
  - This mode is used where cascading is used and the priority has to be programmed in the master using ICW4.
    - when an interrupt request from a slave is in service, this slave can further send request to the master,
      - if the requesting device connected to the slave has higher priority than the one being currently served.
  - In this mode, the master interrupt the CPU only when the interrupting device has a higher or the same priority than the one current being served.

Modes of Operation

- **End of Interrupt (EOI)** :
  - The ISR bit can be reset either with AEOI bit of ICW4 or by EOI command, issued before returning from the interrupt service routine.
- There are two types of EOI commands specific and non-specific.
  - When 8259A is operated in the modes that preserve fully nested structure, it can determine which ISR bit is to be reset on EOI.
  - When non-specific EOI command is issued to 8259A it will be automatically reset the highest ISR bit out of those already set.

Modes of Operation

- **Automatic Rotation** :
  - This is used in the applications where all the interrupting devices are of equal priority.
    - an interrupt request IR level receives priority after it is served while the next device to be served gets the highest priority in sequence.
      - Once all the device are served like this, the first device again receives highest priority.

Direct Memory Access - DMA

- A system that can control the memory system without using the CPU
  - On a specified stimulus, the module will move data from one memory location or region to another memory location or region
- In computers there is often a need to transfer a large number of bytes of data between memory and peripherals such as disk drives.
- Using the microprocessor to transfer the data is too slow since:
  - The data first must be fetched into the CPU
  - Sent it to its destination.

DMA Concept

- The process of fetching and decoding the instructions themselves adds to the overhead.
- For this reason, Intel created the 8237 DMAC (direct memory access controller) chip,
- The function of 8237 DMAC is to bypass the CPU and provide a direct connection between peripherals and memory, thus transferring the data as fast as possible.
- 8237 can transfer a byte of data between an I/O peripheral and memory in only 4 clocks,

Problem Statement

- There is only one set of buses (one set of each bus: data bus, address bus, control bus) in a given computer and no bus can serve two masters at the same time.
- The buses can be used either by the main CPU 80x86 or the 8237 DMA.
- Since the 80x86 has primary control over the buses, it must give *permission to DMA to use them*.

Processor and DMA for Bus

- DMA sends a signal called HOLD to the CPU
- The CPU will respond by sending back the signal HLDA (hold acknowledge) to indicate to the DMA that it can go ahead and use the buses.
- While the DMA is using the buses to transfer data, the CPU is sitting idle,
- and when the CPU is using the bus, the DMA is sitting idle.
- After DMA finishes its job it will make HOLD go low and then the CPU will regain control over the buses

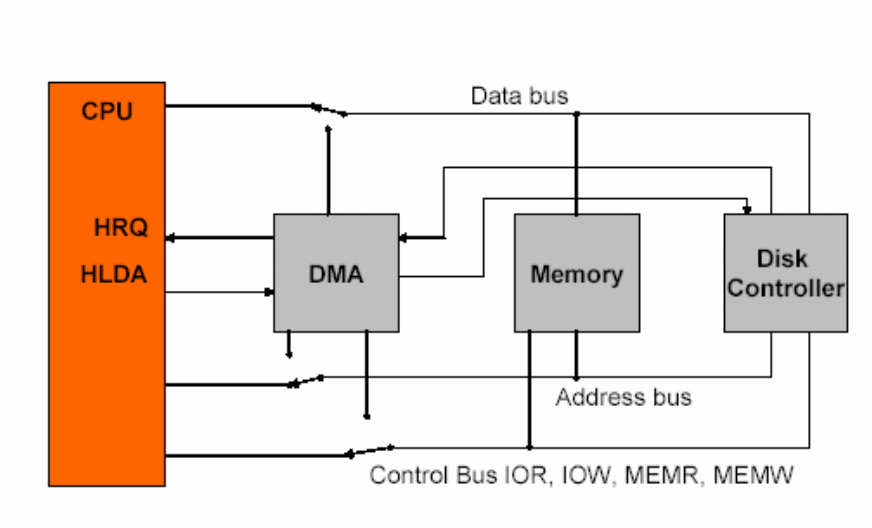
How to transfer

- If the DMA is to transfer a block of data from memory to an I/O device such as a disk, it must know the following:
- Address of the beginning of the block (address of first byte of data)
- The number of bytes (count) it needs to transfer.

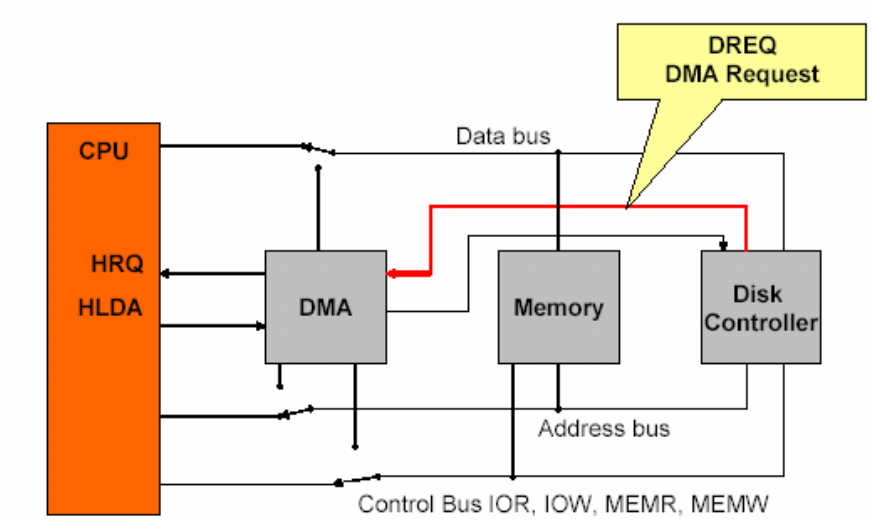
Steps - DMA

1. The peripheral device (such as the disk controller) will request the service of DMA by pulling DREQ (DMA request) high.
2. The DMA will put a high on its HRQ (hold request), signaling the CPU through its HOLD pin that it needs to use the buses.
3. The CPU will finish the present bus cycle and respond to the DMA request by putting high on its HLDA (hold acknowledge), thus telling the 8237 DMA that it can go ahead and use the buses to perform its task.
4. HOLD must remain active high as long as DMA is performing its task.
5. DMA will activate DACK (DMA acknowledge), which tells the peripheral device that it will start to transfer the data.

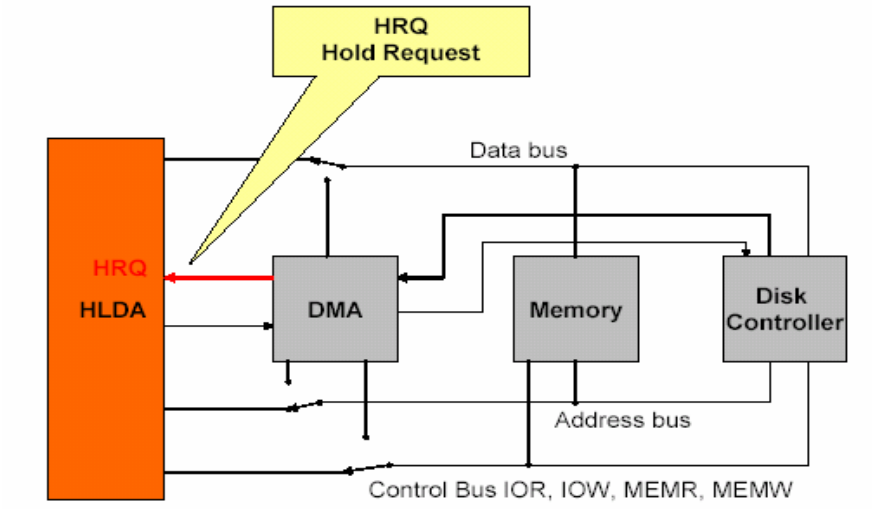
Processor controlled Buses



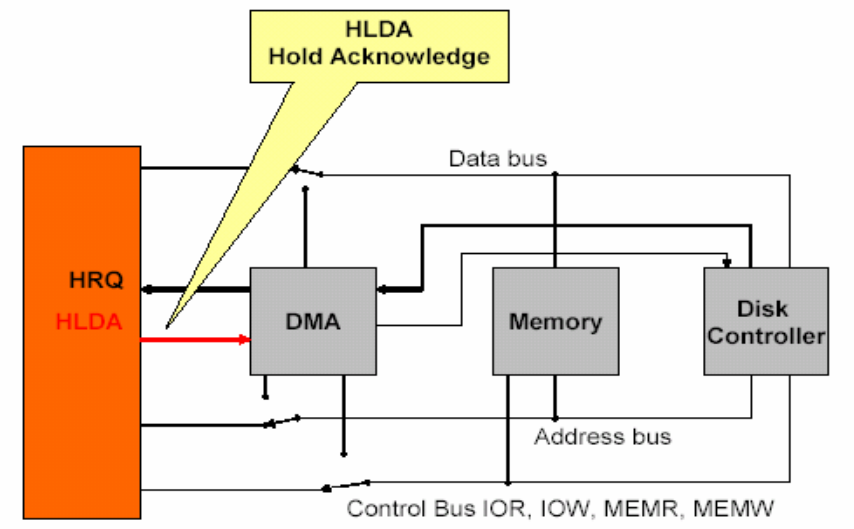
Processor Controlled Busses



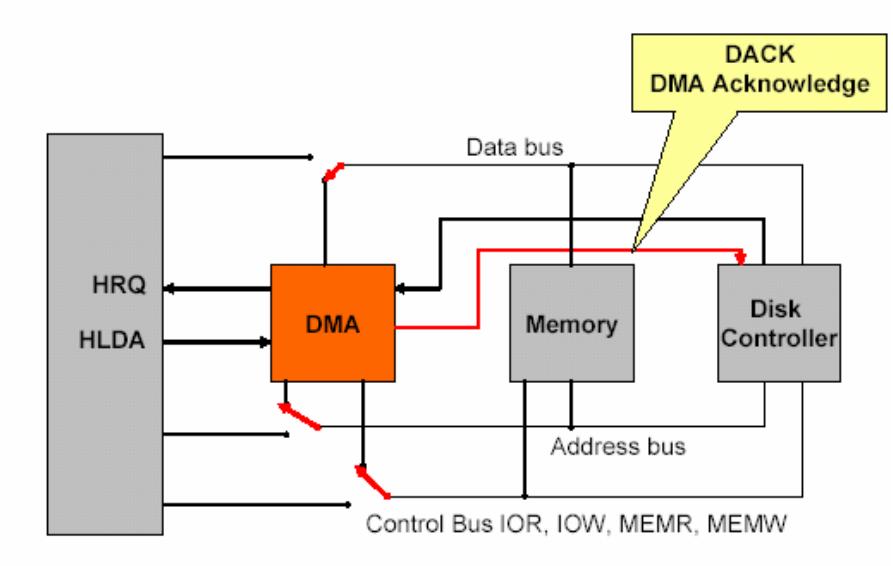
Processor controlled Buses



Processor Controlled Buses



DMA controlled Buses



DMA Steps

6. DMA starts to transfer the data from memory to peripheral as follows
   - DMA puts the address of the first byte of the block on the address bus
   - Activates MEMR,
   - Reads the byte from memory into the data bus
   - Then activates IOW to write it to the peripheral.
   - Then DMA decrements the counter and increments the address pointer
   - Repeats this process until the count reaches zero and the task is finished.
7. After the DMA has finished its job it will deactivate HRQ, signaling the CPU that it can regain control over its buses.

DMA

- This above discussion indicates that DMA can only transfer information; unlike the CPU, it cannot decode and execute instructions.
- when the CPU receives a HOLD request from DMA, it finishes the present bus cycle (but not necessarily the present instruction) before it hands over control of the buses to the DMA.
- This is in contrast to a hardware interrupt, in which the CPU finishes the present instruction before it responds with INTA.
- One could look at the DMA as a kind of CPU without the instruction decoder/executer logic circuitry.

DMA

- For the DMA to be able to transfer data it is equipped with

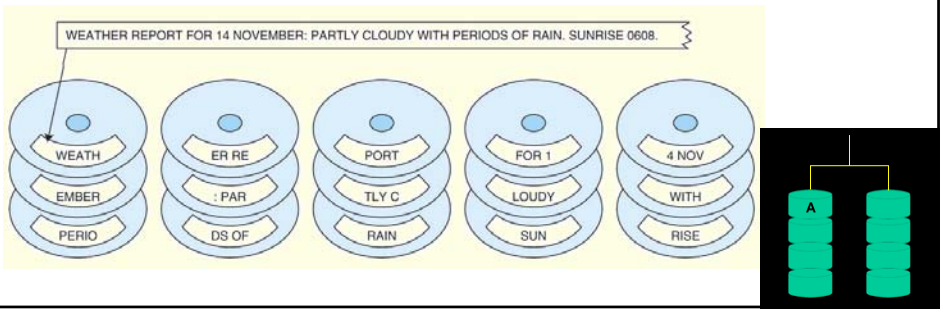
1. **Address bus,**
2. **Data bus,**
3. **Control bus signals IOR, IOW, MEMR, and MEMW.**

RAID

- RAID, *Redundant Array of Independent Disks* was invented to address problems of disk reliability, cost, and performance.
  - used to increase the performance and/or reliability of data storage
  - RAID system consists of two or more disks working in parallel
- In RAID, data is stored across many disks, with extra disks added to the array to provide error correction (redundancy).
- RAID levels
  - RAID level 0 RAID level 1
  - RAID level 2 RAID level 3
  - RAID level 4 RAID level 5

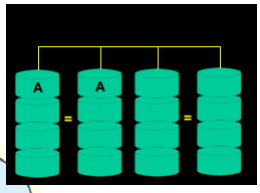
RAID

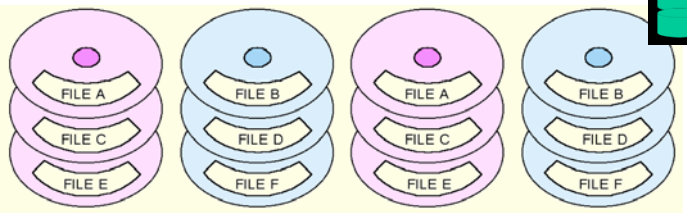
- RAID Level 0, also known as *drive spanning*, provides improved performance, but no redundancy.
  - Data is written in blocks across the entire array
    - Since no data redundancy so performance is good
  - The disadvantage of RAID 0 is in its low reliability.
    - not fault-tolerant. If one disk fails, all data in the RAID 0 array are lost. It should not be used on mission-critical systems



RAID

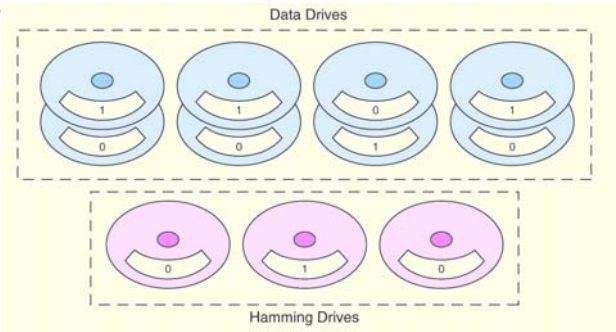
- RAID Level 1, also known as *disk mirroring*, provides 100% redundancy, and good performance.
  - Two matched sets of disks contain the same data.
    - Used for accounting systems
  - The disadvantage of RAID 1 is cost.
    - the effective storage capacity is only half of the total disk capacity because all data get written twice





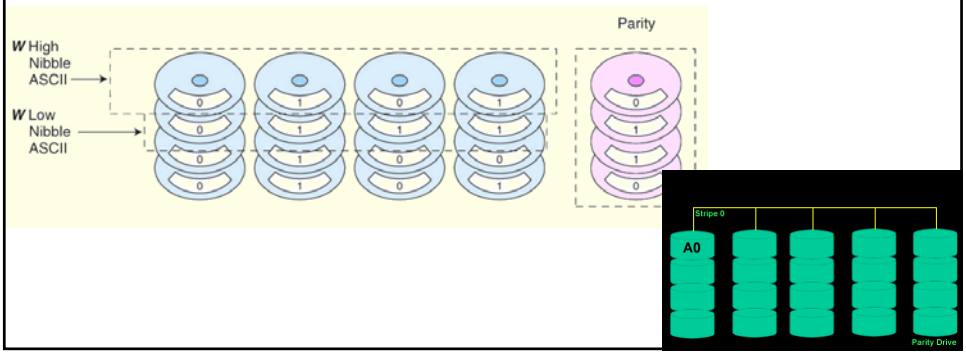
RAID

- A RAID Level 2 configuration consists of:
  - A set of data drives, and
  - A set of Hamming code drives.
    - Hamming code drives provide error correction for the data drives.
  - RAID 2 performance is poor and the cost is relatively high



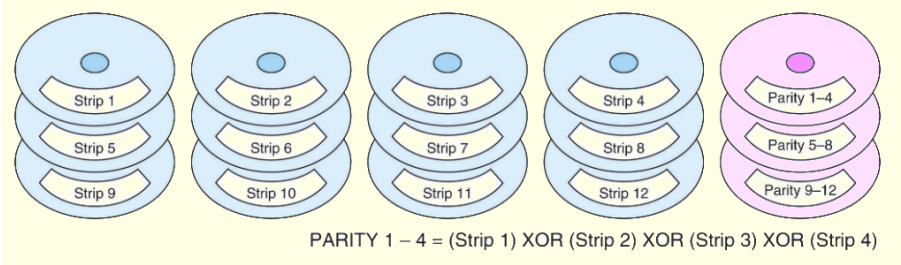
RAID

- RAID Level 3 stripes bits across a set of data drives and provides a separate disk for parity.
  - Parity is the XOR of the data bits.
  - RAID 3 is not suitable for commercial applications, but is good for personal systems.
    - fairly complex and too resource intensive to be done in software.



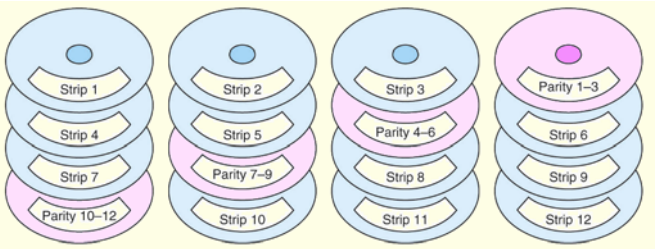
RAID

- RAID Level 4 is like adding parity disks to RAID 0.
  - Data is written in blocks across the data disks, and a parity block is written to the redundant drive.
  - RAID 4 would be feasible if all record blocks were the same size.



RAID

- RAID Level 5 is RAID 4 with distributed parity.
  - With distributed parity, some accesses can be serviced concurrently, giving good performance and high reliability.
  - RAID 5 is used in many commercial systems.



PARITY 1 – 3 = (Strip 1) XOR (Strip 2) XOR (Strip 3)