DEVICE MANAGEMENT

Basic Functions of Device Management:

- Keeps track of the status of all devices, which requires special mechanisms. Eg. Unit Control Block (UCB).
- Deciding on policy to determine who gets a device, for how long, and when. There are three basic techniques for implementing the policies of device management:
    - Dedicated- a technique whereby a device is assigned to a single process.
    - Shared- a technique whereby a device is shared by many processes.
    - Virtual- a technique whereby one physical device is simulated on another physical device.
- Allocation- physically assigning a device to process.
- De allocation may be done on either a process or a job level.

Techniques for Device Management:

Three major techniques are used for managing and allocating devices:

(1) dedicated, (2) shared, and (3) virtual.

Dedicated Devices:

A dedicated device is allocated to a job for the job's entire duration. Some devices lend themselves to this form of allocation.

It is difficult, for example, to share a card reader, tape, or printer.

Shared Devices:

Some devices such as disks, drums and most other Direct Access Storage Devices (DASD) may be shared concurrently by several processes.

Several processes can read from a single disk at essentially the same time.

Process request is to be satisfied first based on (1) a priority list or (2) the objective of achieving improved system output.

Virtual Devices:

Some devices that would normally have to be dedicated (e.g., card readers) may be converted into shared devices through techniques such as **spooling.**

For example, spooling program can read and copy all card input onto a disk at high speed.

Generalized Strategies:

Some professionals have attempted to generalize device management even more than is done here.

For example, the call side of spooling is similar to the file system and buffering bears striking similarity to spooling.

DEVICE CHARACTERISTICS – HARDWARE CONSIDERATIONS

Peripheral devices can be generalized into two major groups:

(1) input or output devices and (2) storage devices.

Input or Output Devices:

Input Device:

An input device is one by which the computer "senses" or "feels" the outside world. These devices may be mechanisms such as thermometers or radar devices.

Output Device:

An output device is one by which the computer "affects" or "controls" the outside world. It may be mechanisms such as temperature control knob or a radar direction control.

Storage Devices:

A storage device is a mechanism by which the computer may store information (a procedure commonly called writing) in such a way that this information may be retrieved at a later time (reading). T_{ij}=time to access item j, given last access was to item i.

Differentiation of storage devices based on the variation of access time (T_{ij}):

1 Serial Access where T_{ij} has a large variance(e.g., tape)

2 Completely Direct Access where T_{ij} is constant (e.g., core)

3 Direct Access where T_{ij} has only a small variance (e.g., drum)

SERIAL ACCESS DEVICES:

A serial access storage device can be characterized as one that relies on strictly physical sequential positioning and accessing of information.

A Magnetic Tape Unit (MTU) is the most common example of a serial access storage device. It is based on same principle as the audio tape desk or tape cassette. A typical serial access device is depicted in Figure 1

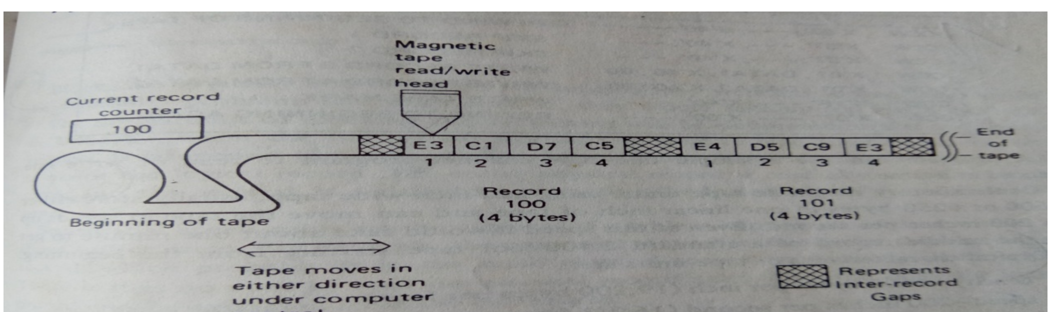


Figure 5-1 Magnetic tape unit

Information is usually stored as groups of bytes, called records, rather than single bytes. A record can be of arbitrary length, e.g., from 1 to 32,768 bytes long.

Each record can be identified by its physical position on the tape; the first record is 1, the second is 2, and so on. In order to help establish groupings of records on a tape, there is a special record type called a tape mark.

Typical characteristics of tape units are:

Density- 1600 bytes per inch(19,200 bytes per foot)

Speed- 200 inches per second (16 2/3 feet per second)

Length- 2400 feet long(i.e., capacity is about 46,000,000 bytes)

Maximum access = 2 minutes

Random (average) access = 1 minute

Serial access (at one record read the next) ~ 4ms

The Inter-record Gap(IRG is necessary due to the physical limitations of starting and stopping a tape. To minimize gap waste, block records are used. Blocking is placing multiple logical records into one physical record(a block). Blocks are typically 800 to 8000 bytes long.

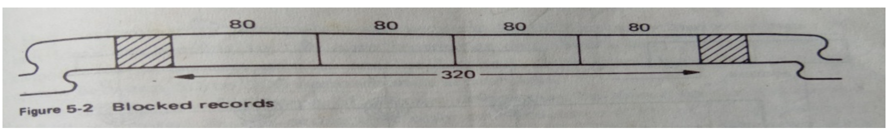


Figure 5-2 Blocked records

There are three major advantages to blocking:

1. Each I/O operations reads or writes multiple logical records at a time.
2. There is less wasted space
3. Smaller tape space is covered when reading many records

There are also three major disadvantages:

1. Software overhead and routines are required to do the blocking, deblocking, and record keeping.
2. Buffer space is wasted.
3. There is more likelihood of tape errors since long records are being read.

COMPLETELY DIRECT ACCESS DEVICES

A completely direct access device is one in which the access time T_{ij} is a constant.

Magnetic core memory, semiconductor memory, read-only wired memories, and diode matrix memories are all examples of completely direct access memories.

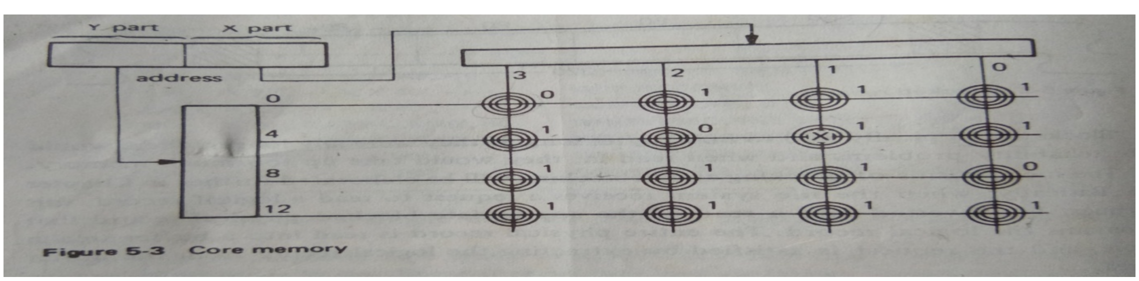


Figure 5-3 Core memory

In Fig 3 the circles represent magnetic ferrite cores. All cores are connected by wires, each core having two selection wires through it.

One property of a magnetic core is that when it changes magnetic state it induces a current. A third sensing wire is passed through all the cores.

For core memory T_{ij} is constant for all I and j and is typically about 500 nanoseconds to 2 microseconds. Usually this type of memory is used as main memory.

DIRECT ACCESS STORAGE DEVICES (DASD):

A device access device is one that is characterized by small variances in the access time $T_{ij.}$ These have been called Direct Access Storage Devices. Although they do not offer completely direct access, the name persists in the field.

Two examples of DASD are fixed-head and moving-head devices.

Fixed-Head Drums and Disks

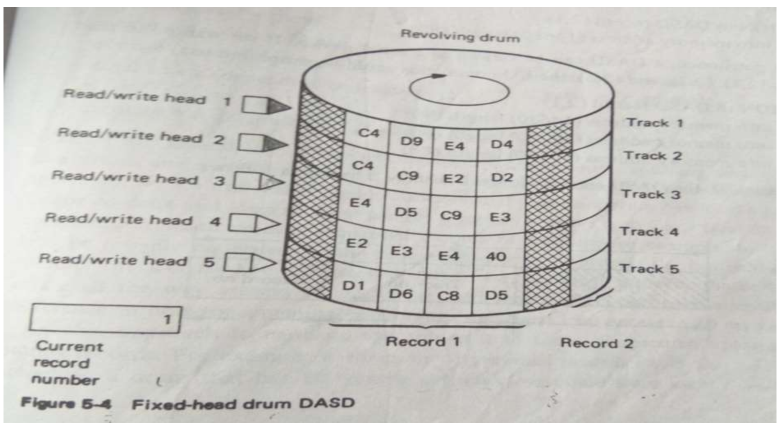


Fig.4 depicts a DASD storage device similar to a magnetic drum. A magnetic drum can be simplistically viewed as several adjacent strips of magnetic tape wrapped around a drum so that the ends of each tape strip join. Each tape strip, called a track, has a separate head/write head.

Each individual record is identified by a track number and then a record number.

It is necessary to specify the DASD address in the I/O channel commands. Logically , the I/O command would be:

READ (1) DASD record (2, 1) into

(2) memory address (14680)

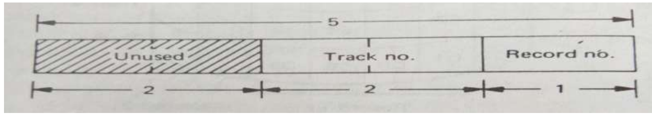
(3) length (80)

In most cases this request is actually broken into two parts: (1) DASD positioning and (2) data transfer operation. Thus the above I/O command would become:

POSITION to DASD record (2,1)

READ into memory address (14680) length (80)

For the 370, the DASD record address is usually 5 bytes, as follows:



The positioning action on the 370 actually requires two channel commands, as follows:

```
GET     CCW X'31',ADDR,X'40',5      SEARCH ID: IS THIS THE RIGHT
                                               RECORD? IF YES, SKIP
                                               NEXT CCW
        CCW X'08',GET,X'40',--      TRANSFER:  IF NO, TRY AGAIN WITH
                                               NEXT RECORD
        CCW X'06',DATA,X'00',80     READ 80 BYTES OF DATA
ADDR    DC  X'0000000201'           TRACK 2, RECORD 1
DATA    DS  CL80                    80-BYTE DATA AREA
```

Typical characteristics of a drum are:

Rotation speed=10 ms

Maximum access=10 ms

Average (random) access=10 ms

Serial access (depending on length of record) < 1ms

Capacity = 8 million bytes (256 heads * 32,000 bytes per track)

} T_{ij}- minor variance (1 order of magnitude)

Blocking is also helpful in improving the performance of sequential processing. If the drum uses Inter-record Gaps, blocking helps to minimize the amount of wasted space due to the IRGs.

Moving-Head Disks and Drums

The magnetic disk is similar to the magnetic drum but is based upon the use of one or more flat disks, each with a series of concentric circles, one per read/write head.

Since read/write heads are expensive, some devices do not have a unique head for each data track.

The heads are physically moved from track to track; such units are called moving-arm or moving-head $DASD_s$. Each arm position is called a cylinder. In order to identify a particular record stored on the moving-head DASD shown in Figure.5

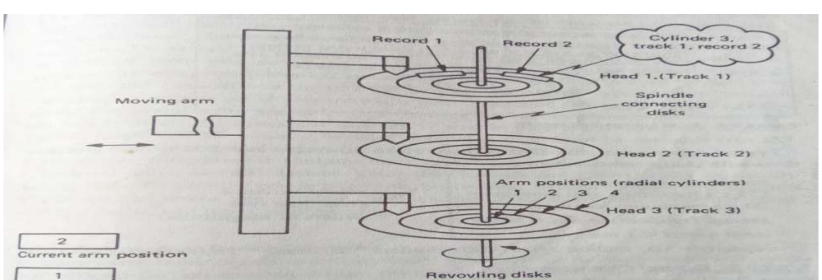
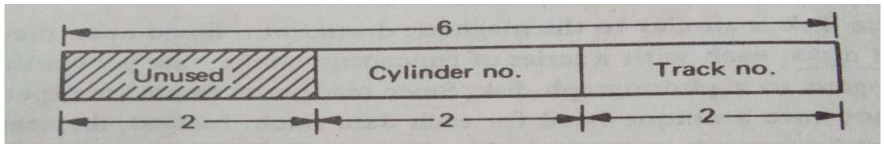


Figure 5-5 Moving-head disk DASD

There is a separate I/O command, called a Seek operation, to position the arm. A seek position is specified using six bytes, as follows:



Typical moving-arm disk DASDs have 10 to 50 tracks and 200 to 400 arm positions with characteristics such as:

Maximum access= 75 ms

(55 ms maximum arm movement,

20 ms maximum rotation)

Average(random) access=30 ms

Serial access < 1 ms

T_{ij}- minor variance (less than two orders of magnitude)

Capacity=100 million bytes(400 cylinders * 20 heads* 16,000 bytes per track)

I/O TRAFFIC CONTROLLER, I/O SCHEDULER, I/O DEVICE HANDLERS

The functions of device management can be conveniently divided into three parts:

1 **I/O traffic Controller-** Keeps track of the status of all devices, control units, and channels.

2 **I/O scheduler-** implements the policy algorithms used to allocate channel, control unit, and device access to I/O requests from jobs.

3 **I/O device handlers-** perform the actual dynamic allocations.

I/O Traffic Controller

The I/O traffic controller is primarily concerned with mechanics. The traffic controller maintains all status information.

The traffic controller attempts to answer at least three key questions:

1 Is there a path available to service an I/O request?

2 Is more than one path available?

3 If no path is currently available, when will one be free?

. This can be accomplished by means of Unit Control Blocks (UCB), Control Unit Control Blocks (CUCB), and Channel Control Blocks (CCB).

Unit Control Block

Device unit identification
Status of device
List of control units connected to this device
List of processes waiting for this device

Channel Control Block (CCB)

Channel identification
Status of channel
List of control units connected to this channel
List of processes waiting for this channel

Control Unit Control Block (CUCB)

Control unit identification
Status of control unit
List of devices connected to this control unit
List of channels connected to this control unit
List of processes waiting for this control unit

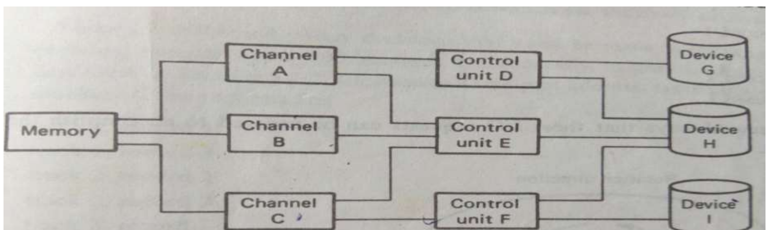


Figure 5-10 Assymetric I/O configuration

I/O scheduler

If there are more I/O requests pending than available paths; it is necessary to choose which I/O requests to satisfy first.

I/O requests are not normally timesliced, that is, once a channel program has been started it is not usually interrupted until it has been completed.

Most channel programs are quite short and terminate within 50-100 ms.

Once the I/O scheduler has determined the relative orderings of the requests, the I/O traffic controller must determine which, if any, of them can be satisfied.

I/O Device Handlers

Handling error conditions, and processing I/O interrupts, the I/O device handlers provide detailed scheduling algorithms that are dependent upon the peculiarities of the device type.

ROTATIONAL ORDERING

Under heavy I/O loads, several I/O requests may be waiting for the same device. Certain orderings of I/O requests may reduce the total time required to service the I/O requests.

Consider drum like device that has only four records stored in it.

The following four I/O requests have been received and a path to the device is now available:

1 Read record 4

2 Read record 3

3 Read record 2

4 Read record 1

There are several ways that these I/O requests can be ordered to accomplish the same result.

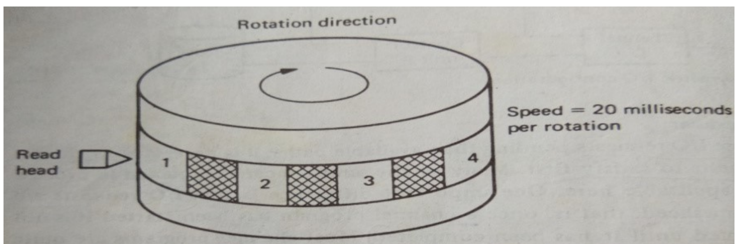


Figure 5-11 Simple drumlike device illustrating notational ordering

ALTERNATE ADDRESSES

The effective access time can be substantially reduced by recording each record at multiple locations on the device. Thus, there are several alternate addresses for reading the same data record. This technique has also been called folding.

If there are n copies per record, the device has been "folded" n times.

SEEK ORDERING

I/O requests require a three part address: cylinder number, track number, and record number. If the requests

Cylinder 1, track 2, record 1

Cylinder 40, track 3, record 3

Cylinder 5, track 6, record 5

Cylinder 1, track 5, record 7

Were processed in the order shown, a considerable amount of time would be spent to move the seek arm back and forth. A more efficient ordering would be:

Cylinder 1, track 2, record 1

Cylinder 1, track 5, record 7

Cylinder 1, track 6, record 5

Cylinder 1, track 5, record 7

Theses requests are in Minimum seek time order, that is the distances between seeks have been minimized.

VIRTUAL DEVICES

Motivation

Devices such as card readers, punches, and printers present two serious problems that hamper effective device utilization.

First, each of the devices perform best when there is a steady, continuous stream of requests at a specified rate that matches its physical characteristics.

Second, these devices must be dedicated to a single job at a time. Thus, since most jobs perform some input and output, we would require as many card readers and printers as we have jobs being multi programmed.

Historical solutions

If it were possible to use Direct Access Devices for all input and output. A single DASD can be efficiently shared and simultaneously used for reading and/or writing data by many jobs as shown in Fig.12.

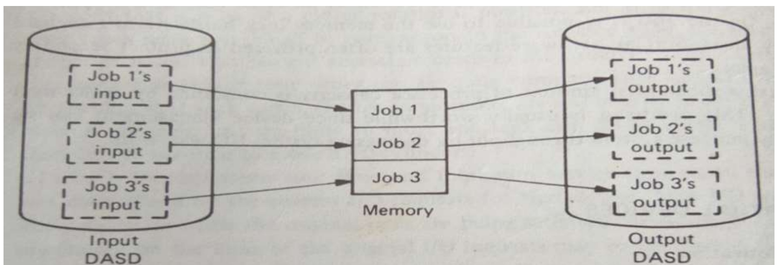


Figure 5-12 Use of DASDs for input and output

DASDs provide very high performance rates , especially if the data are blocked , thereby decreasing the amount of wait time for jobs that require substantial amounts of input/output.

OFFLINE PERIPHERAL OPERATIONS

Fortunately, a solution to this dilemma can be found in the three-step process illustrated in Fig -13.

At step 1 we use a separate computer whose sole function is to read cards at maximum speed and record the corresponding information on a DASD.

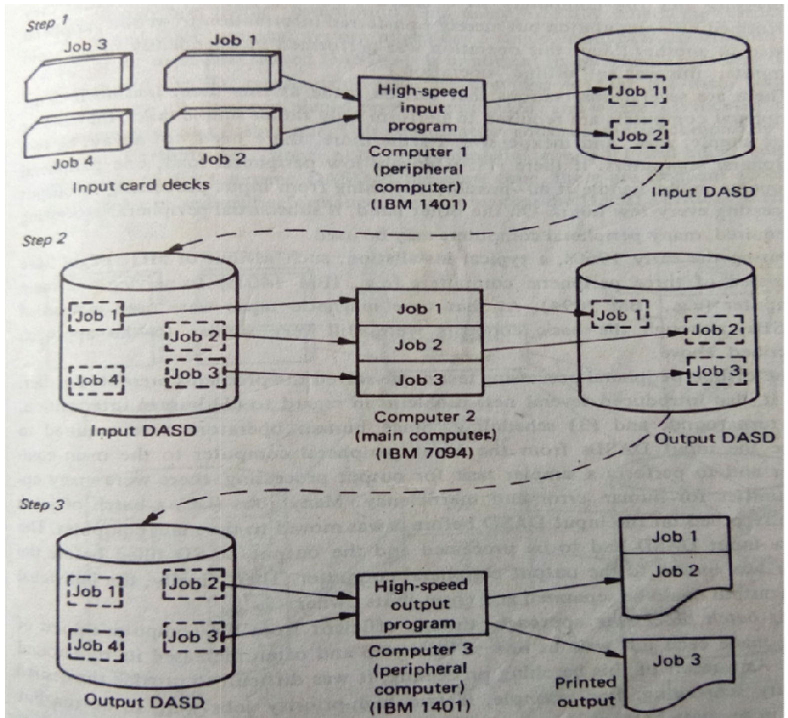


Figure 5-13 Offline peripheral operations

At step 2 the DASD containing the input recorded by computer 1 is moved over to the main processing computer.

Finally, at step 3 the output DASD is moved to a third computer that reads the recorded output at high speed and prints the information on the printers.

The offline peripheral processing technique solved the problems presented earlier, but it also introduced several new problems in regard to (1) Human intervention, (2) turnaround, (3) scheduling.

As a result of this batch processing, it was difficult to provide the desired priority scheduling.

DIRECT-COUPLED SYSTEMS

The major drawback to the offline peripheral processing approach was the need to physically move the output $DASD_s$ between the main computer and the peripheral computer.

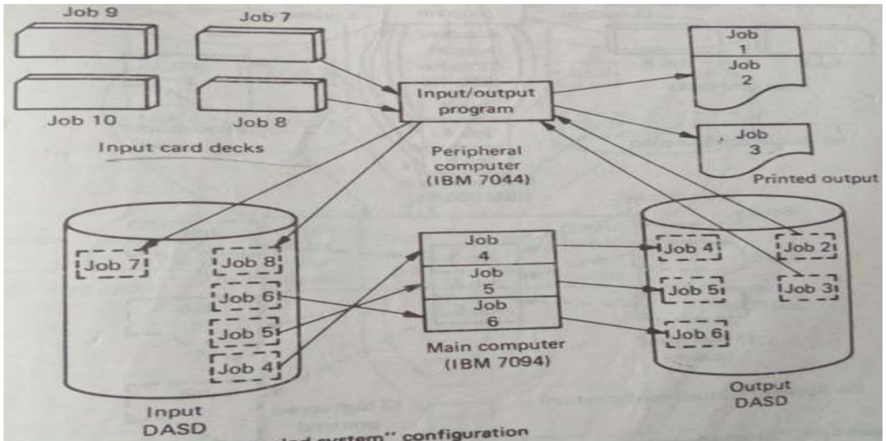


Figure 5-14 "Direct-coupled system" configuration

Fig .14 illustrates a configuration in which the input and output DASDs are physically connected to both the peripheral computer and the main computer, thus eliminating the need for manual handling. This configuration is called a Direct-Coupled Systems (DCS).

The directly coupled system approach eliminates most of the problems of offline peripheral processing. No human intervention is required; there is no "batch processing" turnaround time delay, nor any scheduling restrictions.

ATTACHED SUPPORT PROCESSOR

Another variation on this approach consists of directly connecting the peripheral and main computers together via a high-speed connection as in Fig 1.15a. In this configuration the peripheral computer is called an Attached Support Processor (ASP).

The support processor assumes all responsibility for controlling the input/output peripherals as well as the input and output DASD. It also performs buffering and blocking. The ASP has the appearance of multiple, very high-speed card readers and printers (Fig 15b).

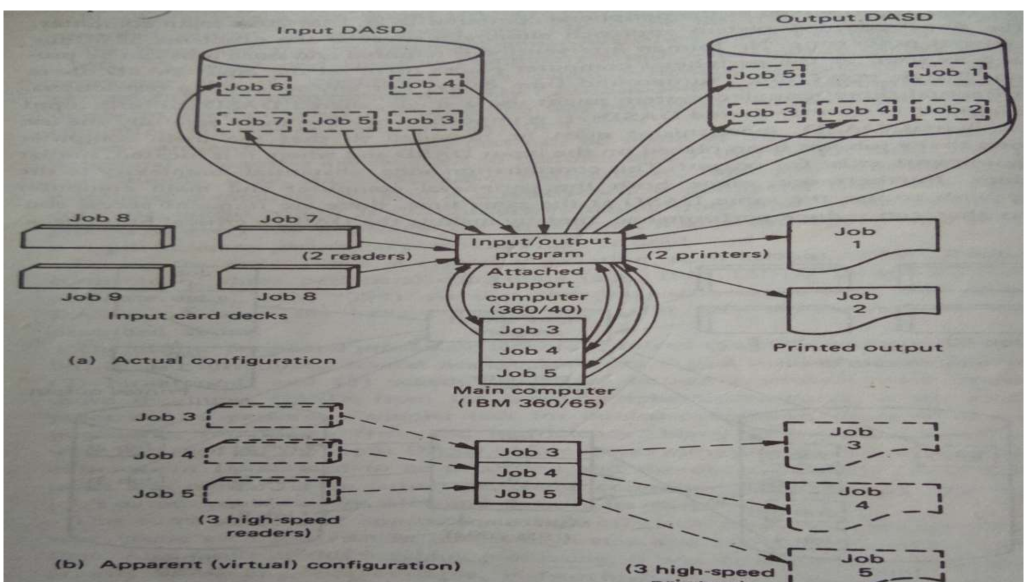


Figure 5-15 "Attached support processor" configuration

The major disadvantages of this ASP approach:

- Two or more processors are required, if it is assigned to some tasks, it is possible that one processor may be idle or underutilized.
- Does not necessarily utilize all resources efficiently.

VIRTUAL SYSTEM

In a SPOOLing system, the main computer performs Simultaneous Peripheral Operations On Line (SPOOL) as illustrated in Fig .16.

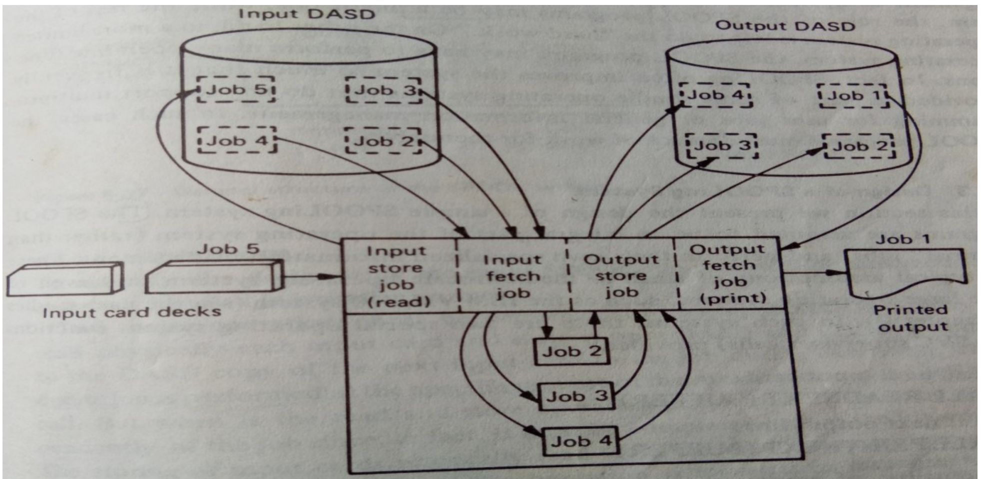


Figure 5-16 SPOOLing system

These jobs are actually system job rather than user jobs, they are often given special names, such as "phantoms" or"daemons".

The jobs that perform the SPOOL functions are special, permanent system jobs, memory management may handle these jobs in different ways. Finally, efficient blocking, buffering, and I/O control must be performed to attain good performance.

Design of a SPOOLing System

The SPOOL programs are assumed to be an integral part of the operating system (rather than "normal" jobs) and perform their own specialized information management. These are typical assumption of small to medium-scale operating system.

In such systems, there are two special operating system functions (e.g., SVC supervisor calls) provided:

1 Read next input card,

CALL READNEXT(BUFFER)

2 Print next output line,

CALL PRINTNEXT(BUFFER)

The card is not physically read, nor the line printed at the time of these calls, since virtual devices are being used.

A general input and output SPOOL system can be subdivided into four components as in Fig .17.

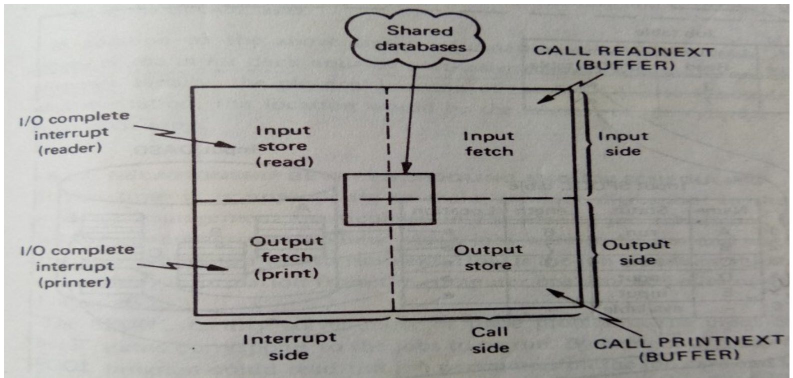


Figure 5-17 General structure of the SPOOL system

The division of a program into a “call side” and an “interrupt side” is a common occurrence in handling various devices and operating system functions.

INPUT SPOOL

Two major operations must be performed by the input SPOOL facility:

1) to read physically each input card and store it on a DASD and
2) to provide access to DASD copy of the next input card to the job during execution.

The operation initiated in response to the I/O complete indication from card reader. This type of operation is called Interrupt driven.

The call side and interrupt side of input SPOOL cannot function independently.

DASD is capable of holding input deck of suitably large size such as 2000 cards.

input: The input deck is still being read.

hold: Input deck has been copied onto the DASD but the corresponding job has not been started yet.

run: The corresponding job is currently running and is reading the input data from the SPOOL area.

RELATIONSHIP BETWEEN SPOOLING AND JOB SCHEDULING

There is a significant relationship between the job scheduler and the SPOOL facility.

The SPOOL table also serves an important database of the job scheduler and is often called the job queue or job hold list.

INPUT SPOOL ALGORITHM

There are supplement databases:

1. the reader table maintains information on the status of teach physical card reader and the corresponding input SPOOL area being used, and
2. the job table maintains information on the status of all jobs currently running and the corresponding SPOOL area being used as a virtual device.

If DASD is organized as one hundred 80-byte records per track and twenty tracks per cylinder, the SPOOL table entry designates the cylinder number, and

Track number = card number/20

Record number = remainder [card umber/20]

Thus card 1= (track 0, record 1) on the cylinder, and card 15=(track 1, record 5), etc.

Fig .19a illustrates the algorithm used by the interrupt side of the input SPOOL program.

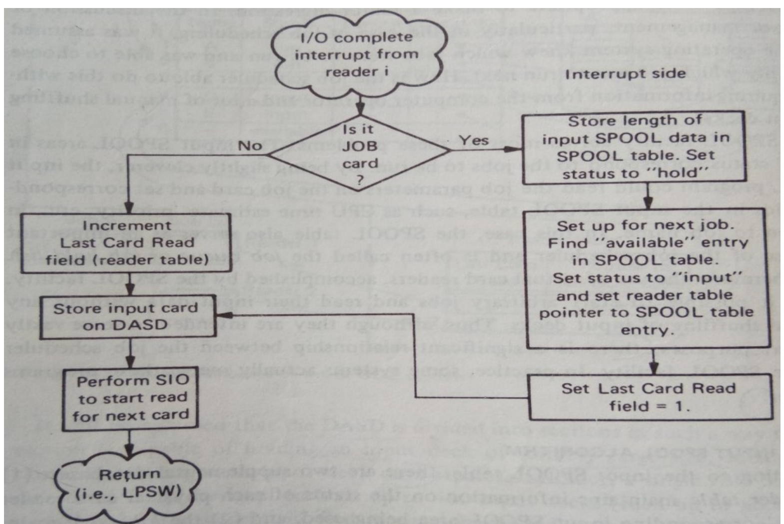


Figure 5-19a Interrupt side of input SPOOL program

Fig .19b illustrates the call side of input SPOOL program

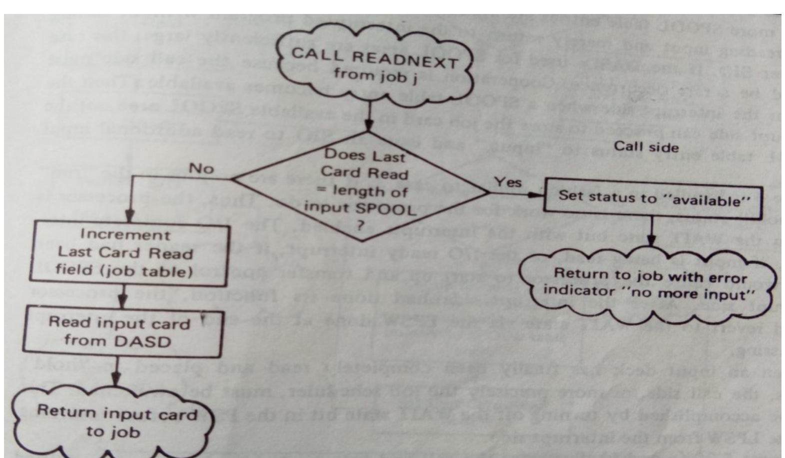


Figure 5-19b Call side of input SPOOL program