ACCESS AND ORGANIZATION OF SECONDARY MEMORY DEVICES

by

INJA CHUN

B. A., Ewha Woman's University, 1966

A MASTER'S REPORT

submitted in partial fulfillment of the requirement
for the degree

MASTER OF SCIENCE

Department of Computer Science

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1973

Approved by

Major Professor

Table of Contents

Chapter

I. INTRODUCTION

This report discusses data storage and access techniques of the general purpose file systems on secondary memory devices. In addition, the idea of segmentation is introduced as an example to show the limitations of file systems.

Various types of computer systems that use operating system techniques (software/hardware composites) are:

(1) real-time control systems: reservation, telephone switching, process control;

(2) data-base systems: management information, credit reporting;

(3) general purpose programming systems: batch, multiprogramming, time sharing;

(4) other computer networks.

Throughout the sections of this report, the discussion of file systems are based upon general purpose programming systems.

II. MULTIPLE NATURE OF SECONDARY STORAGE

A file is an organized collection of related information usually kept in peripheral storage devices such as magnetic tape, disk or drum. The portion of the file system storage which is immediately accessible to the file system, i.e. disk and drums, is called the on-line storage system. Devices which are removable from the storage complex, such as tapes and disk packs, which are used as an extention of on-line facilities, are called the file backup storage system.

Table 1. Parameters of Storage Devices with Typical Values in year 1965 (25, pp122)

Device	Typical Values
Magnetic core Storage :	
Capacity	0.065 - 8
Access Time	0.25 - 4 msec
Cycle Time	0.5 - 8 msec
Brandwidth (flow rate)	8 - 64
Magnetic Drum :	
Capacity	6 - 32
Access Time	8 msec
Brandwidth (flow rate)	1 - 8
Magnetic Disk :	
Capacity	20 - 2,000
Access Time to a Cylinder (seek)	80 msec (average)
Rotational Delay	10 msec (average)
Capacity of One Cylinder	0.2 - 1
Brandwidth (flow rate)	2.4
Magnetic Tape :	
Capacity (one reel)	100 - 500
Start-Stop Time	3 - 10 msec
Flow Rate	0.15 - 2.5

Note: All capacity in millions of bits; all flow rate in millions of bits per second

Among the different types of secondary memory devices, the choice of storage should depend on the cost of information per unit and access time per unit of information. Generally cost increases as access time decreases. Therefore in selecting a storage medium for each application, consideration should be given to the logical organization of data, frequency of use, desired access time and so on. For example, little-used information should be put on devices with longer access time to allow more frequently used files on faster devices; files which need to be accessed directly should be put on directly accessable devices (i.e. disk, drum, but not tape).

The subsequent discussions will be on physical and functional characteristics of the most common auxiliary storage devices: tape, disk and drum.

MAGNETIC TAPES

The most common magnetic tape consists of a reel of $\frac{1}{2}$-inch-wide, $1\frac{1}{2}$ mils (0.0015 inch) thick, and 2400 feet long tape. Magnetic tape has a magnetic surface on which data can be stored by selective magnetization. There are two methods of recording the binary data on the tape. The older method, still in use, records the binary digits with a net magnetic field and binary zeros by the absence of magnetic field. An additional non-data parity bit is also recorded to make the number of 1-bits in a column either even (even parity) or odd (odd parity); this bit is used to check the validity of data by the hardware when the tape is read. The newer method is to record the binary bits with a net magnetic field and binary zeros with a reverse magnetic field. This method is more reliable.

Data are thus recorded on the tape in lengthwise stripes, which are called tracks, one byte at a time in densities of 200, 556, or 800 bits-per-inch (bpi), but 1100, 1600 bpi tape is also used. More data can be stored on a tape at higher densities, and the transmission rate is also increased.

For efficient use of magnetic tape devices, logical records are usually grouped into a block (physical record), which is the amount of information that a computer handles on a single I/O operation. Blocks are written on a tape, separated by an interblock gap, where no information is recorded. This gap must be left for the tape to slow down and get up to speed on, since tape must reach its proper speed for the magnetic code on a tape to be read properly. Fig. 1 shows this arrangement.

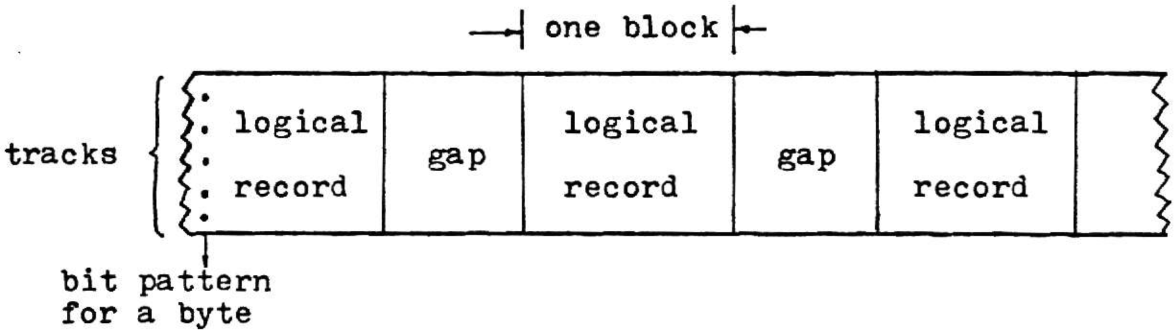


Fig. 1. Unblocked Record on Magnetic Tape

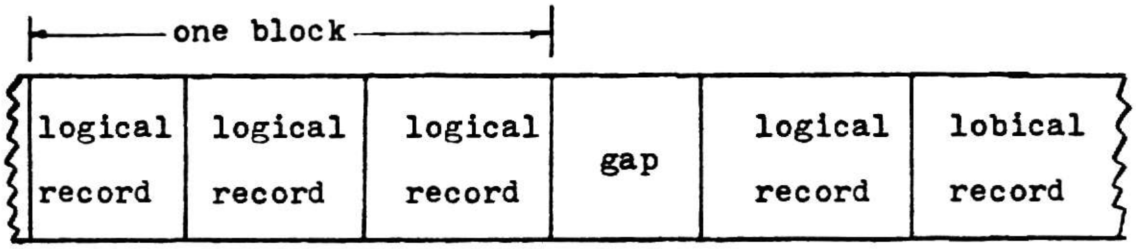


Fig. 2. Blocked record with a blocking factor of 3

An example of the economic use of blocking records is shown below: Suppose we have logical records each of which is 80 characters long, a record occupies $\frac{1}{10}$ inch, at 800 bpi density. If each record is a block, followed by a $\frac{3}{4}$ inch gap, then a series of 100 logical records will occupy 85 inches of tape. On the other hand, if this same tape is blocked with 10 records, then 100 records will occupy 17.5 inches of tape. Thus, the blocking cuts the length of tape necessary by a factor of approximately five. The elapsed time required to read the tape is also cut by the same factor. Therefore, the choice of size of a block is an important programming problem - this depends on many factors, particularly the amount of processor storage that can be allocated for buffering the record, i.e., for accumulating the flow from (or to) tape. A trade-off between available buffer space in main storage and blocking factor must be considered in creating a file. Section III discusses blocking and deblocking of records.

Another factor that affects the efficient use of magnetic tape is keeping the tape in motion between blocks. When a read or write command is given to the magnetic tape unit, it places the magnetic tape in motion. The acceleration time is called the start time. If another read or write command is given before transmission of one block finishes, then the tape will keep on its maximum transmission speed. If not, the tape decelerates to a rest. This deceleration time is called the stop time. Using typical values, the total transport time is approximated by:

Tt: transmission time

s: maximum velocity (transmission speed) = 120 in/sec

ts: start time = stop time = 0.005 seconds

g: length of interblock gap = 3/4 inch

d: density of tape = 800 byte/inch

l: length of tape = 2400 feet

The transport time for n blocks of information (W bytes) with stacked read request is:

$$Tt_n = 2ts + \frac{nW}{d \cdot s} + \frac{(n-1)g}{s}$$

The transport time for the same information with stopping between each read is:

$$nT1 = 2nts = \frac{nW}{ds}$$

Therefore efficient use of magnetic tape device requires the blocking of information and stacking the read and write request to keep the tape in motion between blocks.

Magnetic tape is the least expensive storage media at the present time. Its shortcoming is that it has to be read serially, which is consquently time-consuming if the required information is randomly distributed throughout the tape.

Table 2. IBM 2400 Series Magnetic Tape.

	Milliseconds per Block																					Blocks per Reel					
	2415							2401, 2402, 2403, 2404†														2400 Tape Units					
	Models 1-6					Models 4-6		Model 1			Models 1, 4		Model 4		Models 2**, 5		Model 5		Models 3**, 6	Model 6						
	Seven Track			Nine Track				Seven Track			Nine Track				Nine Track				Nine Track		Seven Track	Seven Track	Seven Track	Nine Track	Nine Track	
	Start Stop Time			Start Stop Time				Start/Stop Time			Start/Stop Time				Start/Stop Time				Start/Stop Time							
	40.0 ms			32.0 ms	16.0 ms*	32.0 ms	16.0 ms*	20.0 ms			16.0 ms	10.6 ms*	16.0 ms	10.6 ms*	8.0 ms	6.7 ms*	8.0 ms	6.7 ms*	5.3 ms	5.3 ms	0.75 IBG	0.75 IBG	0.75 IBG	0.6 IBG	0.6 IBG	
Block Size	200 bpi	556 bpi	800 bpi	800 bpi	800 bpi	1,600 bpi	1,600 bpi	200 bpi	556 bpi	800 bpi	800 bpi	800 bpi	1,600 bpi	1,600 bpi	800 bpi	800 bpi	1,600 bpi	1,600 bpi	800 bpi	1,600 bpi	200 bpi	556 bpi	800 bpi	800 bpi	1,600 bpi	Block Size
20	45.33	41.92	41.33	33.33	17.33	35.40	19.40	22.67	20.96	20.67	16.67	11.27	17.70	12.30	8.33	7.03	8.85	7.55	5.52	5.87	33,459	36,185	36,697	45,504	42,847	20
25	46.67	42.40	41.67	33.67	17.67	35.56	19.56	23.33	21.20	20.83	16.83	11.43	17.79	12.39	8.42	7.12	8.89	7.59	5.58	5.90	32,503	35,775	36,403	45,053	42,647	25
30	48.00	42.88	42.00	34.00	18.00	35.73	19.73	24.00	21.44	21.00	17.00	11.60	17.87	12.47	8.50	7.20	8.93	7.63	5.63	5.93	31,600	35,375	36,114	44,613	42,448	30
35	49.33	43.36	42.33	34.33	18.33	35.90	19.90	24.67	21.68	21.17	17.17	11.77	17.95	12.55	8.58	7.28	8.97	7.67	5.69	5.96	30,745	34,984	35,830	44,179	42,251	35
40	50.67	43.84	42.66	34.67	18.67	36.06	20.06	25.33	21.92	21.33	17.33	11.93	18.04	12.64	8.67	7.37	9.01	7.71	5.74	5.98	29,937	34,601	35,550	43,754	42,055	40
45	52.00	44.32	43.00	35.00	19.00	36.23	20.23	26.00	22.16	21.50	17.50	12.10	18.12	12.72	8.75	7.45	9.05	7.75	5.80	6.01	29,169	34,226	35,274	43,337	41,862	45
50	53.33	44.80	43.33	35.33	19.33	36.40	20.40	26.67	22.40	21.67	17.66	12.26	18.20	12.80	8.83	7.53	9.10	7.80	5.85	6.04	28,440	33,860	35,003	42,928	41,670	50
55	54.67	45.27	43.66	35.67	19.67	36.56	20.56	27.33	22.64	21.83	17.83	12.43	18.29	12.89	8.92	7.62	9.14	7.84	5.91	6.07	27,746	33,501	34,735	42,527	41,480	55
60	56.00	45.75	44.00	36.00	20.00	36.73	20.73	28.00	22.88	22.00	18.00	12.60	18.37	12.97	9.00	7.70	9.18	7.88	5.97	6.10	27,086	33,150	34,473	42,133	41,292	60
65	57.34	46.23	44.33	36.34	20.34	36.90	20.90	28.66	23.12	22.16	18.16	12.76	18.45	13.05	9.09	7.79	9.22	7.92	6.02	6.12	26,456	32,805	42,214	41,747	41,106	65
70	58.67	46.71	44.66	36.67	20.67	37.06	21.06	29.33	23.36	22.33	18.33	12.93	18.54	13.14	9.17	7.87	9.26	7.96	6.08	6.15	25,855	32,469	33,958	41,367	40,921	70
75	60.00	47.19	45.00	37.00	21.00	37.23	21.23	30.00	23.60	22.50	18.50	13.10	18.62	13.22	9.25	7.95	9.30	8.00	6.13	6.18	25,280	32,140	33,707	40,995	40,738	75
80	61.34	47.67	45.33	37.34	21.34	37.39	21.39	30.66	23.84	22.66	18.66	13.26	18.71	13.31	9.34	8.04	9.34	8.04	6.19	6.21	24,730	31,816	33,459	40,629	40,556	80
85	62.67	48.15	45.66	37.67	21.67	37.56	21.56	31.33	24.08	22.83	18.83	13.43	18.79	13.39	9.42	8.12	9.39	8.09	6.24	6.24	24,204	31,499	33,215	40,269	40,376	85
90	64.00	48.63	45.99	38.00	22.00	37.73	21.73	32.00	24.32	23.00	19.00	13.60	18.87	13.47	9.50	8.20	9.43	8.13	6.30	6.26	23,700	31,189	32,974	39,916	40,198	90
95	65.34	49.11	46.33	38.34	22.34	37.89	21.89	32.66	24.56	23.16	19.16	13.76	18.96	13.56	9.59	8.29	9.47	8.17	6.35	6.29	23,216	30,884	32,737	39,569	40,021	95
100	66.67	49.59	46.66	38.67	22.67	38.06	22.06	33.33	24.80	23.33	19.33	13.93	19.04	13.64	9.67	8.37	9.51	8.21	6.41	6.32	22,752	30,585	32,503	39,228	39,845	100
105	68.00	50.07	46.99	39.00	23.00	38.23	22.23	34.00	25.04	23.50	19.50	14.10	19.12	13.72	9.75	8.45	9.55	8.25	6.47	6.35	22,306	30,292	32,272	38,892	39,672	105
110	69.34	50.55	47.33	39.34	23.34	38.39	22.39	34.66	25.28	23.66	19.66	14.26	19.21	13.81	9.84	8.54	9.59	8.29	6.52	6.38	21,877	30,005	32,045	38,563	39,500	110
115	70.67	51.03	47.66	39.67	23.67	38.56	22.56	35.33	25.52	23.83	19.83	14.43	19.29	13.89	9.92	8.62	9.64	8.34	6.58	6.40	21,464	29,723	31,821	38,239	39,329	115
120	72.00	51.51	47.99	40.00	24.00	38.73	22.73	36.00	25.76	24.00	20.00	14.60	19.37	13.97	10.00	8.70	9.68	8.38	6.63	6.43	21,067	29,446	31,600	37,920	39,160	120
125	73.34	51.99	48.33	40.34	24.34	38.89	22.89	36.66	26.00	24.16	20.16	14.76	19.46	14.06	10.09	8.79	9.72	8.42	6.69	6.46	20,684	29,175	31,382	37,607	38,992	125
130	74.67	52.47	48.65	40.67	24.67	39.06	23.06	37.33	26.24	24.33	20.33	14.93	19.54	14.14	10.17	8.87	9.76	8.46	6.74	6.49	20,314	28,908	31,167	37,298	38,826	130
135	76.00	52.95	48.99	41.00	25.00	39.23	23.23	38.00	26.48	24.50	20.50	15.10	19.62	14.22	10.25	8.95	9.80	8.50	6.80	6.52	19,959	28,646	30,955	36,995	38,661	135
140	77.34	53.43	49.32	41.34	25.34	39.39	23.39	38.66	26.72	24.66	20.66	15.26	19.71	14.31	10.34	9.04	9.84	8.54	6.85	6.54	19,614	28,389	30,746	36,697	38,497	140
145	78.67	53.91	49.66	41.67	25.67	39.56	23.56	39.33	26.96	24.83	20.83	15.43	19.79	14.39	10.42	9.12	9.88	8.58	6.91	6.57	19,281	28,136	30,540	36,403	38,335	145
150	80.01	54.39	49.99	42.00	26.00	39.73	23.73	40.00	27.20	25.00	20.99	15.59	19.87	14.47	10.50	9.20	9.93	8.63	6.96	6.60	18,960	27,888	30,336	36,114	38,174	150
160	82.67	55.34	50.66	42.67	26.67	40.06	24.06	41.33	27.68	25.33	21.33	15.93	20.04	14.64	10.67	9.37	10.01	8.71	7.08	6.66	18,348	27,045	29,937	35,550	37,857	160
170	85.34	56.30	51.32	43.34	27.34	40.39	24.39	42.66	28.16	25.66	21.66	16.26	20.21	14.81	10.84	9.54	10.09	8.79	7.19	6.71	17,775	26,938	29,548	35,003	37,545	170
180	88.01	57.26	51.99	44.01	28.01	40.72	24.72	43.99	28.64	25.99	21.99	16.59	20.38	14.98	11.01	9.71	10.17	8.87	7.30	6.77	17,235	26,487	29,169	34,473	37,237	180
190	90.67	58.22	52.65	44.67	28.67	41.06	25.06	45.33	29.12	26.33	22.33	16.93	20.54	15.14	11.17	9.87	10.26	8.96	7.41	6.82	16,729	26,050	28,800	33,958	36,935	190
200	93.34	59.18	53.32	45.34	29.34	41.39	25.39	46.66	29.60	26.66	22.66	17.26	20.71	15.31	11.34	10.04	10.34	9.04	7.52	6.88	16,251	25,628	28,440	33,459	36,638	200
210	96.01	60.14	53.99	46.01	30.01	41.72	25.72	47.99	30.08	26.99	22.99	17.59	20.88	15.48	11.51	10.21	10.42	9.12	7.63	6.94	15,820	25,220	28,089	32,974	36,345	210
220	98.67	61.10	54.65	46.67	30.57	42.06	26.06	49.33	30.56	27.33	23.33	17.93	21.04	15.64	11.67	10.37	10.51	9.21	7.74	6.99	15,373	24,824	27,746	32,500	36,057	220
230	101.34	62.05	55.32	47.34	31.34	42.39	26.39	50.66	31.04	27.66	23.66	18.26	21.21	15.81	11.84	10.54	10.59	9.29	7.85	7.05	14,968	24,440	27,412	32,045	35,774	230
240	104.01	63.02	55.98	48.01	32.01	42.72	26.72	51.99	31.52	27.99	23.99	18.59	21.38	15.98	12.01	10.71	10.67	9.37	7.96	7.10	14,585	24,068	27,086	31,600	35,495	240
250	106.68	63.98	56.65	48.67	32.67	43.06	27.06	53.33	32.00	28.33	24.32	18.92	21.54	16.14	12.17	10.87	10.76	9.46	8.07	7.16	14,220	23,707	26,767	31,167	35,220	250
260	109.34	64.93	57.32	49.34	33.34	43.39	27.39	51.66	32.48	28.66	24.66	19.26	21.71	16.31	12.34	11.04	10.84	9.54	8.19	7.22	13,873	23,357	26,456	30,746	34,949	260
270	112.01	65.89	57.98	50.01	34.01	43.72	27.72	55.99	32.96	28.99	24.99	19.59	21.88	16.48	12.51	11.21	10.92	9.62	8.30	7.27	13,543	23,017	26,152	30,336	34,683	270
280	114.68	66.85	58.65	50.68	34.68	44.05	28.05	57.32	33.44	29.32	25.32	19.92	22.05	16.65	12.68	11.38	11.00	9.70	8.41	7.33	13,228	22,687	25,855	29,937	34,421	280
290	117.34	67.81	59.31	51.34	35.34	44.39	28.39	58.66	33.92	29.66	25.66	20.26	22.21	16.81	12.84	11.54	11.09	9.79	8.52	7.38	12,927	22,366	25,564	29,548	34,162	290
300	120.01	68.77	59.98	52.01	36.01	44.72	28.72	59.99	34.40	29.99	25.99	20.59	22.38	16.98	13.01	11.71	11.17	9.87	8.63	7.44	12,640	22,054	25,280	29,169	33,908	300
310	122.68	69.73	60.65	52.68	36.68	45.05	29.05	61.32	34.88	30.32	26.32	20.92	22.55	17.15	13.18	11.88	11.25	9.95	8.74	7.50	12,365	21,751	25,002	28,800	33,657	310
320	125.34	70.69	61.31	53.34	37.34	45.39	29.39	62.66	35.36	30.66	26.66	21.26	22.71	17.31	13.34	12.04	11.34	10.04	8.85	7.55	12,102	21,455	24,730	28,440	33,410	320
330	128.01	71.65	61.98	54.01	38.01	45.72	29.72	63.99	35.84	30.99	26.99	21.59	22.88	17.48	13.51	12.21	11.42	10.12	8.96	7.61	11,850	21,168	24,465	28,089	33,166	330
340	130.68	72.61	62.64	54.68	38.68	46.05	30.05	65.32	36.32	31.32	27.32	21.92	23.05	17.65	13.68	12.38	11.50	10.20	9.07	7.66	11,608	20,889	24,204	27,746	32,926	340
350	133.35	73.57	63.31	55.34	39.34	46.39	30.39	66.66	36.80	31.66	27.65	22.25	23.21	17.81	13.84	12.54	11.59	10.29	9.18	7.72	11,376	20,616	23,949	27,412	32,690	350
360	136.01	74.52	63.98	56.01	40.01	46.72	30.72	67.99	37.28	31.99	27.99	22.59	23.38	17.98	14.01	12.71	11.67	10.37	9.30	7.78	11,153	20,351	23,700	27,086	32,456	360
370	138.68	75.48	64.64	56.68	40.68	47.05	31.05	69.32	37.76	32.32	28.32	22.92	23.55	18.15	14.18	12.88	11.75	10.45	9.41	7.83	10,938	20,092	23,456	26,767	32,227	370
380	141.35	76.44	65.31	57.35	41.35	47.38	31.38	70.65	38.24	32.65	28.65	23.25	23.72	18.32	14.35	13.05	11.83	10.53	9.52	7.89	10,732	19,840	23,216	26,456	32,000	380
390	144.01	77.40	65.97	58.01	42.01	47.72	31.72	71.99	38.72	32.99	28.99	23.59	23.88	18.48	14.51	13.21	11.92	10.62	9.63	7.94	10,533	19,594	22,982	26,152	31,777	390
400	146.68	78.36	66.64	58.68	42.68	48.05	32.05	73.32	39.20	33.32	29.32	23.92	24.05	18.65	14.68	13.38	12.00	10.70	9.74	8.00	10,342	19,355	22,752	25,855	31,556	400
410	149.35	79.32	67.31	59.35	43.35	48.38	32.38	74.65	39.68	33.65	29.65	24.25	24.22	18.82	14.85	13.55	12.08	10.78	9.85	8.06	10,157	19,120	22,527	25,564	31,339	410
420	152.01	80.28	67.97	60.01	44.01	48.72	32.72	75.99	40.16	33.99	29.99	24.59	24.38	18.98	15.01	13.71	12.17	10.87	9.96	8.11	9,979	18,892	22,306	25,280	31,124	420
430	154.68	81.24	68.64	60.68	44.68	49.05	33.05	77.32	40.64	34.32	30.32	24.92	24.55	19.15	15.18	13.88	12.25	10.95	10.07	8.17	9,807	18,669	22,089	25,002	30,913	430
440	157.35	82.20	69.30	61.35	45.35	49.38	33.38	78.65	41.12	34.65	30.65	25.25	24.72	19.32	15.35	14.05	12.33	11.03	10.18	8.22	9,641	18,451	21,877	24,730	30,704	440
450	160.02	83.15	69.97	62.01	46.01	49.72	33.72	79.99	41.60	34.99	30.98	25.58	24.88	19.48	15.51	14.21	12.42	11.12	10.29	8.28	9,480	18,238	21,669	24,465	30,499	450
460	162.68	84.11	70.64	62.68	46.68	50.05	34.05	81.32	42.08	35.32	31.32	25.92	25.05	19.65	15.68	14.38	12.50	11.20	10.41	8.34	9,325	18,030	21,464	24,204	30,295	460
470	165.35	85.07	71.30	63.35	47.35	50.38	34.38	82.65	42.56	35.65	31.65	26.25	25.22	19.82	15.85	14.55	12.58	11.28	10.52	8.39	9,174	17,827	21,264	23,949	30,095	470
480	168.02	86.03	71.97	64.02	48.02	50.71	34.71	83.98	43.04	35.98	31.98	26.58	25.39	19.99	16.02	14.72	12.66	11.36	10.63	8.45	9,029	17,628	21,067	23,700	29,898	480
490	170.68	86.99	72.63	64.68	48.68	51.05	35.05	85.32	43.52	36.32	32.32	26.92	25.55	20.15	16.18	14.88	12.75	11.45	10.74	8.50	8,888	17,434	20,873	23,456	29,702	490
500	173.35	87.95	73.30	65.35	49.35	51.38	35.38	86.65	44.00	36.65	32.65	27.25	25.72	20.32	16.35	15.05	12.83	11.53	10.85	8.56	8,751	17,244	20,684	23,216	29,510	500
550	186.69	92.75	76.63	68.68	52.68	53.05	37.05	93.32	46.40	38.32	34.31	28.91	26.55	21.15	17.18	15.88	13.25	11.95	11.40	8.84	8,126	16,352	19,764	22,089	28,583	550
600	200.02	97.54	79.96	72.02	56.02	54.71	38.71	99.98	48.80	39.98	35.98	30.58	27.39	21.99	18.02	16.72	13.66	12.36	11.96	9.12	7,584	15,548	18,960	21,067	27,713	600
650	213.35	102.34	83.29	75.35	59.35	56.38	40.38	106.65	51.20	41.65	37.64	32.24	28.22	22.82	18.85	17.55	14.08	12.78	12.51	9.40	7,110	14,820	18,202	20,135	26,894	650

	Milliseconds per Block																				Blocks per Reel					
	2415							2401, 2402, 2403, 2404†													2400 Tape Units					
	Models 1-6					Models 4-6		Model 1			Models 1, 4		Model 4		Models 2**, 5		Model 5		Models 3**, 6	Model 6						
	Seven Track			Nine Track				Seven Track			Nine Track				Nine Track				Nine Track		Seven Track	Seven Track	Seven Track	Nine Track	Nine Track	
	Start Stop Time 40.0 ms				Start Stop Time			Start/Stop Time 20.0 ms				Start/Stop Time				Start/Stop Time			Start/Stop Time							
				32.0 ms	16.0 ms*	32.0 ms	16.0 ms*				16.0 ms	10.6 ms*	16.0 ms	10.6 ms*	8.0 ms	6.7 ms*	8.0 ms	6.7 ms*	5.3 ms	5.3 IBG	0.75 ms	0.75 IBG	0.75 IBG	0.6 IBG	0.6 IBG	
Block Size	200 bpi	556 bpi	800 bpi	800 bpi	800 bpi	1,600 bpi	1,600 bpi	200 bpi	556 bpi	800 bpi	800 bpi	800 bpi	1,600 bpi	1,600 bpi	800 bpi	800 bpi	1,600 bpi	1,600 bpi	800 bpi	1,600 bpi	200 bpi	556 bpi	800 bpi	800 bpi	1,600 bpi	Block Size
700	[illegible]	107.13	86.62	78.69	62.69	58.04	42.04	113.31	53.60	43.31	39.31	33.91	29.06	23.66	19.69	18.39	14.49	13.19	13.07	9.68	6,592	14,156	17,552	19,281	26,122	700
750	240.03	111.93	89.95	82.02	66.02	59.71	43.71	119.98	56.00	44.98	40.97	35.57	29.89	24.49	20.52	19.22	14.91	13.61	13.62	9.96	6,320	13,550	16,853	18,433	25,333	750
800	253.35	116.73	93.28	85.36	69.36	61.37	45.37	126.64	58.40	46.64	42.64	37.24	30.73	25.33	21.36	20.06	15.32	14.02	14.18	10.24	5,987	12,993	16,251	17,775	[illegible]	800
850	[illegible]	121.52	96.61	88.69	72.69	63.04	47.04	133.31	60.80	48.31	44.30	38.90	31.56	26.16	22.19	20.89	15.74	14.44	14.73	10.52	5,688	12,480	15,691	17,107	24,051	850
900	280.03	126.32	99.94	92.03	76.03	64.70	48.70	139.97	63.20	49.97	45.97	40.57	32.40	27.00	23.03	21.73	16.15	14.85	15.29	10.80	5,417	12,007	15,168	16,487	23,432	900
950	293.37	131.11	103.27	95.36	79.36	66.37	50.37	146.64	65.60	51.64	47.63	42.23	33.23	27.83	23.86	22.56	16.57	15.27	15.64	11.08	5,171	11,567	14,679	15,910	22,913	950
1000	306.70	135.90	106.60	98.70	82.70	68.03	52.03	153.30	68.00	53.30	49.30	43.90	34.07	28.67	24.70	23.40	16.98	15.68	16.40	11.36	4,946	11,159	14,220	15,373	22,264	1000
1050	320.04	140.70	109.93	102.03	86.03	69.70	53.70	159.97	70.40	54.97	50.96	45.56	34.90	29.50	25.53	24.23	17.40	16.10	16.95	11.64	4,740	10,779	13,789	14,871	21,751	1050
1100	333.37	145.49	113.26	105.37	89.37	71.36	55.36	166.63	72.80	56.63	52.63	47.23	35.74	30.34	26.37	25.07	17.81	16.51	17.51	11.92	4,550	10,424	13,384	14,400	21,244	1100
1150	346.71	150.29	116.59	108.70	92.70	73.03	57.03	173.30	75.20	58.30	54.29	48.89	36.57	31.17	27.20	25.90	18.23	16.93	18.06	12.20	4,375	10,091	13,001	13,959	20,759	1150
1200	360.04	155.08	119.92	112.04	96.04	74.69	58.69	179.96	77.60	59.96	55.96	50.56	37.41	32.01	28.04	26.74	18.64	17.34	18.67	12.48	4,213	9,779	12,640	13,543	20,296	1200
1250	373.38	159.88	123.25	115.37	99.37	76.36	60.36	186.63	80.00	61.63	57.62	52.22	38.24	32.84	28.87	27.57	19.06	17.76	19.17	12.76	4,053	9,486	12,298	13,151	19,853	1250
1300	386.71	164.67	126.58	118.71	102.71	78.02	62.02	193.29	82.40	63.29	59.29	53.89	39.08	33.68	29.71	28.41	19.47	18.17	19.73	13.04	3,923	9,209	11,975	12,782	19,430	1300
1350	400.05	169.47	129.91	122.04	106.04	79.69	63.69	199.96	84.80	64.96	60.95	55.55	39.91	34.51	30.54	29.24	19.89	18.59	20.28	13.32	3,792	8,949	11,668	12,433	19,023	1350
1400	413.38	174.26	133.24	125.38	109.38	81.35	65.35	206.62	87.20	66.62	62.62	57.22	40.75	35.35	31.38	30.08	20.30	19.00	20.84	13.60	3,670	8,703	11,376	12,102	18,634	1400
1450	426.72	179.06	136.57	128.71	112.71	83.02	67.02	213.29	89.60	68.29	64.28	58.88	41.58	36.18	32.21	30.91	20.72	19.42	21.39	13.88	3,555	8,740	11,099	11,789	18,250	1450
1500	440.05	183.85	139.90	132.05	116.05	84.68	68.68	219.95	92.00	69.95	65.95	60.55	42.42	37.02	33.05	31.75	21.13	19.83	21.95	14.16	3,447	8,249	10,834	11,491	17,901	1500
1550	453.39	188.65	143.23	135.38	119.38	86.35	70.35	226.62	94.40	71.62	67.61	62.21	43.25	37.85	33.88	32.58	21.55	20.25	22.50	14.44	3,346	8,039	10,582	11,208	17,555	1550
1600	466.72	193.44	146.56	138.72	122.72	88.01	72.01	233.28	96.80	73.28	69.28	63.88	44.09	38.69	34.72	33.42	21.96	20.66	23.06	14.72	3,250	7,840	10,342	10,938	17,223	1600
1650	480.05	198.24	149.89	142.05	126.05	89.68	73.68	239.95	99.20	74.95	70.94	65.54	44.92	39.52	35.55	34.25	22.38	21.08	23.61	15.00	3,160	7,650	10,112	10,682	16,903	1650
1700	493.39	203.03	153.22	145.39	129.39	91.34	75.34	246.61	101.60	76.61	72.61	67.21	45.76	40.36	36.39	35.09	22.79	21.49	24.17	15.28	3,075	7,469	9,892	10,437	16,595	1700
1750	506.73	207.83	156.55	148.72	132.72	93.01	77.01	253.28	104.00	78.28	74.27	68.87	46.59	41.19	37.22	35.92	23.21	21.91	24.72	15.56	2,994	7,297	9,682	10,203	16,298	1750
1800	520.06	212.62	159.88	152.06	136.06	94.67	78.67	259.94	106.40	79.94	75.94	70.54	47.43	42.03	38.06	36.76	23.62	22.32	25.28	15.84	2,917	7,132	9,480	9,979	16,011	1800
1850	533.40	217.42	163.21	155.39	139.39	96.34	80.34	266.61	108.80	81.61	77.60	72.20	48.26	42.86	38.89	37.59	24.04	22.74	25.83	16.12	2,844	6,975	9,287	9,755	15,734	1850
1900	546.73	222.21	166.54	158.73	142.73	98.00	82.00	273.27	111.20	83.27	79.27	73.87	49.10	43.70	39.73	38.43	24.45	23.15	26.39	16.40	2,775	6,825	9,101	9,550	15,467	1900
1950	560.07	227.01	169.87	162.06	146.06	99.67	83.67	279.94	113.60	84.94	80.93	75.53	49.93	44.53	40.56	39.26	24.87	23.57	26.94	16.68	2,709	6,680	8,922	9,353	15,209	1950
2000	573.40	231.80	173.20	165.40	149.40	101.33	85.33	286.60	116.00	86.60	82.60	77.20	50.77	45.37	41.40	40.10	25.28	23.98	27.50	16.96	2,646	6,542	8,751	9,174	14,959	2000
2050	586.74	236.60	176.53	168.73	152.73	103.00	87.00	293.27	118.40	88.27	84.26	78.86	51.60	46.20	42.23	40.93	25.70	24.40	28.05	17.24	2,585	6,410	8,585	8,993	14,717	2050
2100	600.07	241.39	179.86	172.07	156.07	104.66	88.66	299.93	120.80	89.93	85.93	80.53	52.44	47.04	43.07	41.77	26.11	24.81	28.61	17.52	2,528	6,282	8,427	8,813	14,482	2100
2150	613.41	246.19	183.19	175.40	159.40	106.33	90.33	306.60	123.20	91.60	87.59	82.19	53.27	47.87	43.90	42.60	26.53	25.23	29.16	17.80	2,473	6,160	8,273	8,651	14,256	2150
2200	626.74	250.98	186.52	178.74	162.74	107.99	91.99	313.26	125.60	93.26	89.26	83.86	54.11	48.71	44.74	43.44	26.94	25.64	29.72	18.08	2,420	6,042	8,126	8,490	14,036	2200
2250	640.08	255.78	189.85	182.07	166.07	109.66	93.66	319.93	128.00	94.93	90.92	85.52	54.94	49.54	45.57	44.27	27.36	26.06	30.27	18.36	2,370	5,929	7,983	8,334	13,823	2250
2300	653.41	260.57	193.18	185.41	169.41	111.32	95.32	326.59	130.40	96.59	92.59	87.19	55.78	50.38	46.41	45.11	27.77	26.47	30.83	18.64	2,322	5,820	7,845	8,184	13,616	2300
2350	666.75	265.37	196.51	188.74	172.74	112.99	95.99	333.26	132.80	98.26	94.25	88.85	56.61	51.21	47.24	45.94	28.19	26.89	31.38	18.92	2,275	5,715	7,713	8,040	13,415	2350
2400	680.08	270.16	199.84	191.08	175.08	114.65	98.65	339.92	135.20	99.92	95.92	90.52	57.45	52.05	48.08	46.78	28.60	27.30	31.94	19.20	2,231	5,613	7,584	7,900	13,220	2400
2450	693.42	274.96	203.17	195.41	179.41	116.32	100.32	346.59	137.60	101.59	97.58	92.18	58.28	52.88	48.91	47.61	29.02	27.72	32.49	19.48	2,189	5,515	7,460	7,765	13,031	2450
2500	706.75	279.75	206.50	198.75	182.75	117.98	101.98	353.25	140.00	103.25	99.25	93.85	59.12	53.72	49.75	48.45	29.43	28.13	33.05	19.76	2,146	5,421	7,339	7,635	12,847	2500
2550	[illegible]	284.55	209.83	202.08	186.08	119.65	103.65	359.92	142.40	104.92	100.91	95.51	59.95	54.55	50.58	49.28	29.85	28.55	33.60	20.04	2,107	5,330	7,223	7,509	12,669	2550
2600	733.42	289.34	213.16	205.42	189.42	121.31	105.31	366.58	144.80	106.58	102.58	97.18	60.79	55.39	51.42	50.12	30.26	28.96	34.16	20.32	2,068	5,241	7,110	7,387	12,494	2600
2650	746.76	294.14	216.49	208.75	192.75	122.98	106.98	373.25	147.20	108.25	104.24	98.84	61.62	56.22	52.25	50.95	30.68	29.38	34.71	20.60	2,031	5,156	7,001	7,259	12,325	2650
2700	760.09	298.93	219.82	212.09	196.09	124.64	108.64	379.91	149.60	109.91	105.91	100.51	62.46	57.06	53.09	51.79	31.09	29.79	35.27	20.88	1,996	5,073	6,895	7,155	12,160	2700
2750	773.43	303.73	223.15	215.42	199.42	126.31	110.31	386.58	152.00	111.58	107.57	102.17	63.29	57.89	53.92	52.62	31.51	30.21	35.82	21.16	1,961	4,993	6,792	7,044	12,000	2750
2800	786.76	308.52	226.48	218.76	202.76	127.97	111.97	393.21	154.40	113.24	109.24	103.84	64.13	58.73	54.76	53.46	31.92	30.62	36.38	21.44	1,928	4,915	6,692	6,937	11,844	2800
2850	800.10	313.32	229.81	222.09	206.09	129.64	113.64	399.91	156.80	114.91	110.90	105.50	64.96	59.56	55.59	54.29	32.34	31.04	36.93	21.72	1,896	4,840	6,595	6,832	11,692	2850
2900	813.43	318.11	233.14	225.43	209.43	131.30	115.30	406.57	159.20	116.57	112.57	107.17	65.80	60.40	56.43	55.13	32.75	31.45	37.49	22.00	1,865	4,767	6,501	6,731	11,543	2900
2950	826.77	322.91	236.47	228.76	212.76	132.97	116.97	413.24	161.60	118.24	114.23	108.83	66.63	61.23	57.26	55.96	33.17	31.87	38.04	22.28	1,835	4,696	6,409	6,633	11,399	2950
3000	840.10	327.70	239.80	232.10	216.10	134.63	118.63	419.90	164.00	119.90	115.90	110.50	67.47	62.07	58.10	56.80	33.58	32.28	38.60	22.56	1,806	4,628	6,320	6,538	11,258	3000
3050	853.44	332.50	243.13	235.43	219.43	136.30	120.30	426.57	166.40	121.57	117.56	112.16	68.30	62.90	58.93	57.63	34.00	32.70	39.15	22.84	1,778	4,561	6,233	6,445	11,122	3050
3100	866.77	337.29	246.46	238.77	222.77	137.96	121.96	433.23	168.80	123.23	119.23	113.83	69.14	63.74	59.77	58.47	34.41	33.11	39.71	23.12	1,750	4,496	6,149	6,355	10,985	3100
3150	880.11	342.09	249.79	242.10	226.10	139.63	123.63	439.90	171.20	124.90	120.89	115.49	69.97	64.57	60.60	59.30	34.83	33.53	40.26	23.40	1,724	4,433	6,067	6,268	10,855	3150
3200	893.44	346.88	253.12	245.44	229.44	141.29	125.29	446.56	173.60	126.56	122.56	117.16	70.81	65.41	61.44	60.14	35.24	33.94	40.82	23.68	1,698	4,372	5,987	6,183	10,727	3200
3250	906.78	351.68	256.45	248.77	232.77	142.96	126.96	453.23	176.00	128.23	124.22	118.82	71.64	66.24	62.27	60.97	35.66	34.36	41.37	23.96	1,673	4,312	5,910	6,100	10,602	3250
3300	920.11	356.47	259.78	252.11	236.11	144.62	126.62	459.89	178.40	129.89	125.89	120.49	72.48	67.08	63.11	61.81	36.07	34.77	41.93	24.24	1,649	4,254	5,834	6,019	10,480	3300
3350	933.45	361.27	263.11	255.44	239.44	146.29	130.29	466.56	180.80	131.56	127.55	122.15	73.31	67.91	63.94	62.64	36.49	35.19	42.48	24.52	1,625	4,198	5,760	5,940	10,361	3350
3400	946.78	366.06	266.44	258.78	242.78	147.95	131.95	473.22	183.20	133.22	129.22	123.82	74.15	68.75	64.78	63.48	36.90	35.60	43.04	24.80	1,602	4,143	5,688	5,864	10,244	3400

At present many tapes with different formats exist. IBM standard formats for 1/2 inch tape are seven-track (6-bit byte and one bit added for parity), even parity (200, 556 and 800 bpi) and nine-track, odd parity (800 and 1600 bpi). Almost every computer can be adapted IBM-compatible tapes. Other computer vendors have noncompatible tape formats which they believe offer advantages over the IBM STANDARD. Honeywell perfers to use odd parity as standard (as it is easier to detect blank characters or dropped bits), and UNIVAC has a Uniservo format (a clocking or synchronizing feature to compensate for tape speed discrepancies). Table 2 shows detailed information about IBM 2400 series magnetic tapes.

DISK

A disk drive is made up of a stack of rotating recording surfaces (disks) similar to a stack of phonograph records. Each disk surface contains many concentric circles called tracks on which information is recorded. There may be only one read/write head for the whole stack (as in IBM 1301) which has to move from disk to disk as well as from track to track, but usually there is one head per recording surface, mounted on a comblike access mechanism. In some disk drives, there is a read/write head on each track; this type of disk is logically equivalent to a drum. Since all heads are positioned together, a single position of the access arm makes a set of tracks (one track per disk) available. Such a set of tracks vertically arranged is called a cylinder. Since all head positioning (seek) time requires the longest delay for random access, the capacity of a cylinder is important.

Typical disks have from 2 to 100 recording surfaces, and from 200 to 400 tracks per surface, and from 2500 to 10000 characters per track. However, modern practice is tending toward a removable module of six disks with 10 recording surfaces per disk pack (i.e. IBM 2311); as a protective measure, top and bottom surfaces are not used.

Tracks are usually divided into sectors (Fig. 3). Sector size may be fixed by the manufacturer, but it is generally left to the user to define sector size for his particular application.

The recording technology of a disk is similar to that of ordinary tape recording with one important difference: the read/write head do not touch the surfaces, but float or fly about one-thousandth of an inch away. This flying-head construction reduces the attainable data density somewhat, but provides essentially no wear of the record surfaces or head with a very high data reliability.

When information on a disk is addressed as a triple (surface number, track number, sector number), the head on that surface 'seeks' the desired track. This seek time varies with the inter-cylinder distance. After seeking, there is latency (rotational delay) to get to the desired sector on the track.

Denote tr: time for one rotation

ts: maximum seek time

W; bytes/sector

b: sectors/track

z: bytes to be read or written

n: number of cylinders needed

Tdsk: transport time of data (W bytes) on disk

$$\text{Tdsk} = \begin{bmatrix}\text{Total Maximum seek}\\ \text{time for all}\\ \text{cylinders involved}\end{bmatrix} + \begin{bmatrix}\text{Total Average}\\ \text{rotational delay}\\ \text{for all cylinders}\\ \text{involved}\end{bmatrix} + \begin{bmatrix}\text{Time to}\\ \text{transmit}\\ \text{information}\end{bmatrix}$$

$$= nts + (n-1)\frac{tr}{2} + \frac{tr}{2} + \frac{tr(z/w)}{b}$$

If the data are stored in consecutive cylinder:

$$\text{Tdsk} = \begin{bmatrix}\text{Maximum}\\ \text{seek time}\\ \text{for one}\\ \text{cylinder}\end{bmatrix} + \begin{bmatrix}\text{Minimum}\\ \text{seek time}\\ \text{for tracks}\end{bmatrix} + \begin{bmatrix}\text{Average}\\ \text{rotational}\\ \text{delay}\end{bmatrix} + \begin{bmatrix}\text{Time to}\\ \text{transmit}\\ \text{data}\end{bmatrix}$$

$$= ts + (n-1)(\min ts) + \frac{tr}{2} + \frac{tr(z/w)}{b}$$

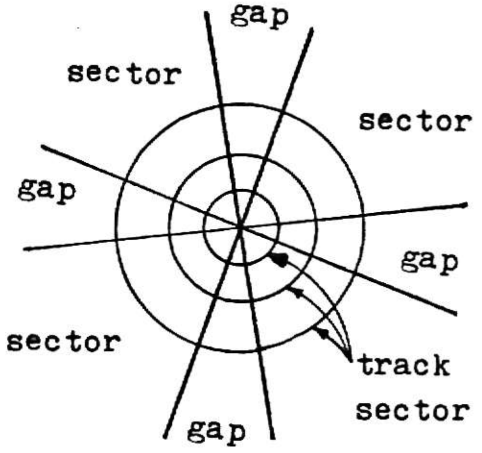
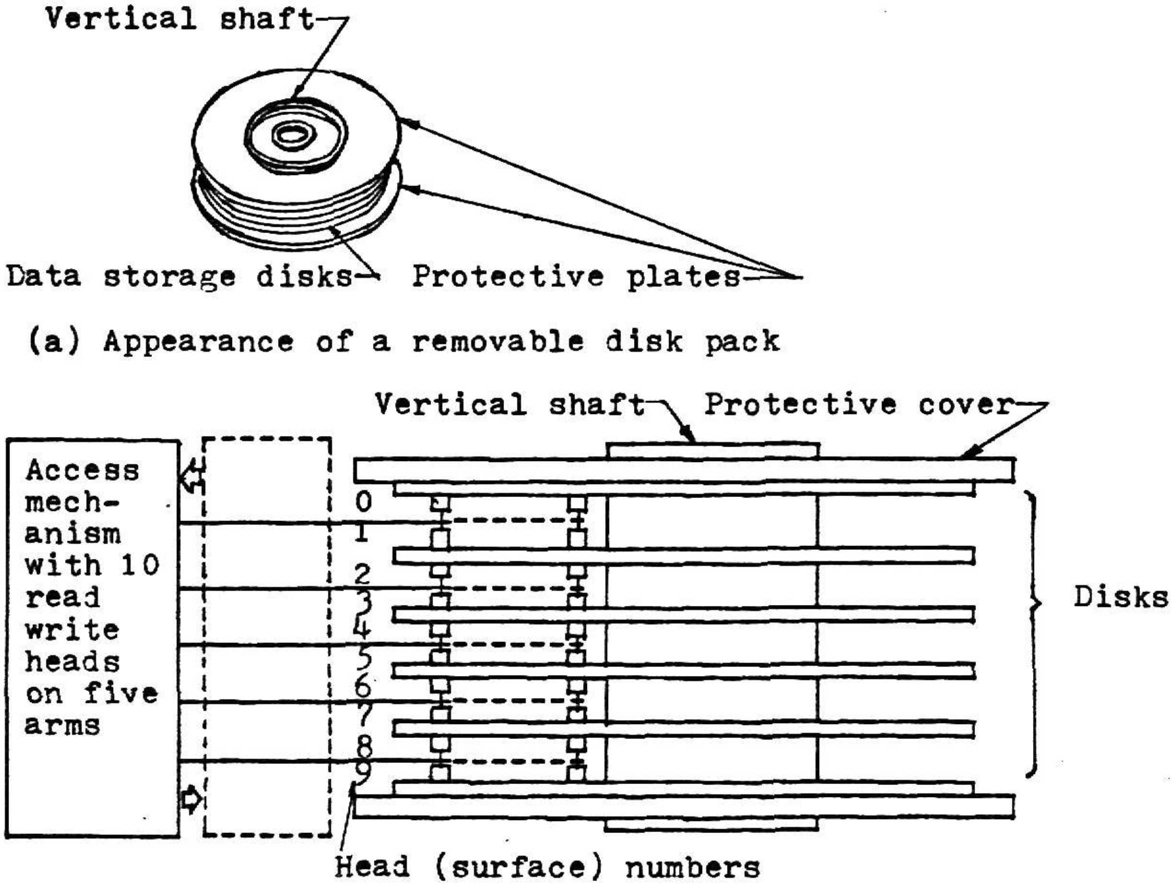


Fig.3. Top View of a disk



(a) Appearance of a removable disk pack

(b) Disks, access mechanism, and read/write heads.

Number of data digits per sector	200
Number of sectors per track	10
Number of data digits per track	2,000
Number of surfaces = number of tracks per cylinder = number of heads	10
Number of data digits per cylinder	20,000
Number of tracks per surface = number of cylinders	100
Total number of data digits per removable pack	2,000,000
Head access (seek time)	400 msec (max) 250 msec (avg)
Rotational delay	40 msec (max)
Flow rate	50,000 digits/sec

Fig. 4. IBM 1311 disk organization for IBM 1620 system.

DASD Capacity and Transmission Time

Table 3. DASD Capacity and Transmission Time.

● WITH KEYS

BYTES PER RECORD		RECORDS PER				TRANSMISSION TIME IN MS PER RECORD	
MINIMUM	MAXIMUM	TRACK	CYLINDER	MODULE	FACILITY	MINIMUM	MAXIMUM
3477	7249	1	20	4000	32000	11.14	23.23
2255	3476	2	40	8000	64000	7.23	11.14
1650	2254	3	60	12000	96000	5.29	7.22
1289	1649	4	80	16000	128000	4.13	5.29
1050	1288	5	100	20000	160000	3.37	4.13
878	1049	6	120	24000	192000	2.81	3.36
751	877	7	140	28000	224000	2.41	2.81
651	750	8	160	32000	256000	2.09	2.40
572	650	9	180	36000	288000	1.83	2.08
507	571	10	200	40000	320000	1.62	1.83
453	506	11	220	44000	352000	1.45	1.62
408	452	12	240	48000	384000	1.31	1.45
369	407	13	260	52000	416000	1.18	1.30
334	368	14	280	56000	448000	1.07	1.18
305	333	15	300	60000	480000	0.98	1.07
278	304	16	320	64000	512000	0.89	0.97
255	277	17	340	68000	544000	0.82	0.89
234	254	18	360	72000	576000	0.75	0.81
216	233	19	380	76000	608000	0.69	0.75
199	215	20	400	80000	640000	0.64	0.69
184	199	21	420	84000	672000	0.59	0.63
169	183	22	440	88000	704000	0.54	0.59
157	168	23	460	92000	736000	0.50	0.54
145	156	24	480	96000	768000	0.46	0.50
134	144	25	500	100000	800000	0.43	0.46
124	133	26	520	104000	832000	0.40	0.43
115	123	27	540	108000	864000	0.37	0.39
106	114	28	560	112000	896000	0.34	0.37
97	105	29	580	116000	928000	0.31	0.34
90	96	30	600	120000	960000	0.29	0.31
83	89	31	620	124000	992000	0.27	0.29
76	82	32	640	128000	1024000	0.24	0.26
70	75	33	660	132000	1056000	0.22	0.24
64	69	34	680	136000	1088000	0.21	0.22
58	63	35	700	140000	1120000	0.19	0.20
52	57	36	720	144000	1152000	0.17	0.18
47	51	37	740	148000	1184000	0.15	0.16
43	46	38	760	152000	1216000	0.14	0.15
38	42	39	780	156000	1248000	0.12	0.13
34	37	40	800	160000	1280000	0.11	0.12
30	33	41	820	164000	1312000	0.10	0.11
26	29	42	840	168000	1344000	0.08	0.09
23	25	43	860	172000	1376000	0.07	0.08
19	22	44	880	176000	1408000	0.06	0.07
15	18	45	900	180000	1440000	0.05	0.06
12	14	46	920	184000	1472000	0.04	0.04
9	11	47	940	188000	1504000	0.03	0.04
5	8	48	960	192000	1536000	0.02	0.03

● WITH KEYS

Models:	1	A1
Average Access Time	75 ms	60 ms
Average Rotational Delay	12.5 ms	12.5 ms

The formulas used to determine capacity and transmission time assume the use of programming systems developed and supported by IBM and are in agreement with Systems Reference Library A26-3599-2, N26-0203 and N26-0230.

These systems use eight bytes of the first record on each track. The formulas are:

a. Bytes per record, except last record on track:
$[2137\ (KL+DL)/2048]^{*} + C+101$

b. Bytes per record, last record on track only:
KL+DL+C

c. Capacity per track in bytes: 7294

d. Records per track:
$$\left[\frac{c-b}{a}\right]^{*} +1$$

e. Data rate (ms per byte): 0.0032051

f. Transmission time (ms per record):
(bytes per record) x (data rate)

KL = Key Length
DL = Data Length
C = 0 when KL = 0
C = 45 when KL ≠ 0

*Truncate any fraction

Models:	1	A1
Average Access Time	75 ms	60 ms
Average Rotational Delay	12.5 ms	12.5 ms

Some examples of how this card may be used follow. A record is considered to be the information recorded between two gaps.

Example 1. Determining the effect of the record length on the number of records that can be stored on the device

Assuming that 73-byte records, without keys, are required, the table indicates that 41 records can be placed on each track. Reducing the record length by one byte permits 42 records per track, an increase of 4000 records per module. Alternatively, the record length can be increased by three bytes without decreasing the number of records on the module.

Example 2. Determining read/write time

Assuming that the data length (DL) is equal to 80 bytes, without key (KL = 0), the table indicates that 40 records can be stored on each track using unblocked records (that is, a gap after each 80 byte record). The chart also indicates a transmission time of 0.26 ms per record. Average read/write time is the sum (87.76 ms) of the average access time (75 ms) plus average rotational delay time (12.5 ms) plus record transmission time (0.26 ms).

Example 3. Effect of blocked records

Assuming 2400 storage locations available for a common input/output area, no keys, and a data length of 80 bytes, a maximum blocking factor of 30 can be established using 2400-byte records. The table indicates that two records of this length can be written on each track for a total of sixty 80-byte logical records. The transmission time for each disk record of thirty logical records is 10.3 ms (by interpolation).

● WITHOUT KEYS

BYTES PER RECORD		RECORDS PER				TRANSMISSION TIME IN MS PER RECORD	
MINIMUM	MAXIMUM	TRACK	CYLINDER	MODULE	FACILITY	MINIMUM	MAXIMUM
3521	7294	1	20	4000	32000	11.29	23.38
2299	3520	2	40	8000	64000	7.37	11.28
1694	2298	3	60	12000	96000	5.43	7.37
1333	1693	4	80	16000	128000	4.27	5.43
1093	1332	5	100	20000	160000	3.50	4.27
922	1092	6	120	24000	192000	2.96	3.50
794	921	7	140	28000	224000	2.54	2.95
695	793	8	160	32000	256000	2.23	2.54
616	694	9	180	36000	288000	1.97	2.22
551	615	10	200	40000	320000	1.77	1.97
497	550	11	220	44000	352000	1.59	1.76
451	496	12	240	48000	384000	1.45	1.59
412	450	13	260	52000	416000	1.32	1.44
378	411	14	280	56000	448000	1.21	1.32
348	377	15	300	60000	480000	1.12	1.21
322	347	16	320	64000	512000	1.03	1.11
299	321	17	340	68000	544000	0.96	1.03
277	298	18	360	72000	576000	0.89	0.96
259	276	19	380	76000	608000	0.83	0.88
242	258	20	400	80000	640000	0.78	0.83
227	241	21	420	84000	672000	0.73	0.77
212	226	22	440	88000	704000	0.68	0.72
200	211	23	460	92000	736000	0.64	0.68
188	199	24	480	96000	768000	0.60	0.64
177	187	25	500	100000	800000	0.57	0.60
167	176	26	520	104000	832000	0.54	0.56
158	166	27	540	108000	864000	0.51	0.53
149	157	28	560	112000	896000	0.48	0.50
140	148	29	580	116000	928000	0.45	0.47
133	139	30	600	120000	960000	0.43	0.45
126	132	31	620	124000	992000	0.40	0.42
119	125	32	640	128000	1024000	0.38	0.40
113	118	33	660	132000	1056000	0.36	0.38
107	112	34	680	136000	1088000	0.34	0.36
101	106	35	700	140000	1120000	0.32	0.34
95	100	36	720	144000	1152000	0.30	0.32
91	94	37	740	148000	1184000	0.29	0.30
86	90	38	760	152000	1216000	0.28	0.29
81	85	39	780	156000	1248000	0.26	0.27
77	80	40	800	160000	1280000	0.25	0.26
73	76	41	820	164000	1312000	0.23	0.24
70	72	42	840	168000	1344000	0.22	0.23
66	69	43	860	172000	1376000	0.21	0.22
62	65	44	880	176000	1408000	0.20	0.21
58	61	45	900	180000	1440000	0.19	0.20
55	57	46	920	184000	1472000	0.18	0.18
52	54	47	940	188000	1504000	0.17	0.17
48	51	48	960	192000	1536000	0.15	0.16
46	47	49	980	196000	1568000	0.15	0.15
44	45	50	1000	200000	1600000	0.14	0.14
41	43	51	1020	204000	1632000	0.13	0.14
38	40	52	1040	208000	1664000	0.12	0.13
35	37	53	1060	212000	1696000	0.11	0.12
33	34	54	1080	216000	1728000	0.11	0.11
31	32	55	1100	220000	1760000	0.10	0.10
28	30	56	1120	224000	1792000	0.09	0.10
26	27	57	1140	228000	1824000	0.08	0.09
24	25	58	1160	232000	1856000	0.08	0.08
23	23	59	1180	236000	1888000	0.07	0.07
21	22	60	1200	240000	1920000	0.07	0.07
19	20	61	1220	244000	1952000	0.06	0.06
17	18	62	1240	248000	1984000	0.05	0.06
15	16	63	1260	252000	2016000	0.05	0.05
13	14	64	1280	256000	2048000	0.04	0.04
12	12	65	1300	260000	2080000	0.04	0.04
10	11	66	1320	264000	2112000	0.03	0.04
8	9	67	1340	268000	2144000	0.03	0.03
7	7	68	1360	272000	2176000	0.02	0.02
5	6	69	1380	276000	2208000	0.02	0.02

● WITHOUT KEYS

The total number of bytes on a track is a function of the recording density and number of sectors/track. The recording density can be expressed as bytes/angle since the same number of bytes are recorded on both inner and outer tracks. An inter-sector gap is needed to distinguish the end of one sector and the beginning of the next; this also is expressed in terms of angle.

The concept of efficient use of disk is similar to that of tape: if the number of sectors per track increases, then the sector size decreases and the total number of bytes per track also decreases due to a higher percentage of gap space.

DRUM

A drum is a cylinder with a magnetizable surface which is divided into circular tracks, each as wide as a read/write head. This device is similar to a disk except it contains a single cylinder of tracks, each with its own read/write head. Drums are made to rotate at high speed (usually 3600 or 7200 rpm) about their axes while many read/write heads float a few millionths of an inch off their external surfaces.

Data are addressed on a drum by specifying the track number and word number within the drum. Often, one or more non-data-storing clock tracks serve to index the drum by providing current word-number indications (as in the IBM 650).

Using the same values used for disk in the previous section, the transport time for drum (Tdrm) can be approximated:

$$\text{Tdrum} = \begin{bmatrix}\text{initial average}\\ \text{rotational delay}\end{bmatrix} + \begin{bmatrix}\text{time to transmit}\\ \text{information}\end{bmatrix}$$

$$= \frac{tr}{2} + \frac{tr(z/w)}{b}$$

Drums offer the fastest secondary storage device available, but its cost per unit of information is most expensive because of the need for many costly read/write heads and for precision machining of surfaces.

The disk drive, which has a read/write head on each track, is logically equivalent to a drum also.

III. OVERVIEW OF DATA MANAGEMENT

In this section an overview of data management is given to show the overall relations of operating system to the file. This section is heavily based on the IBM/360 OS.

Data management refers to the totality of all operating system routines that provide access to data, enforce storage conventions and regulate I/O devices. Therefore data management includes all the data-set organizations, macro instruction languages, random and sequential access methods, the catalog and various indexes, and service routines which permit the operating system to read, write, keep and destroy data.

There are six major kinds of data-set organization:

Data Set Organization	Basic (random)	Queued (sequential)	Comments
Sequential	BSAM	QSAM	Magnetic tape, card and equivalent
Partitioned	BPAM		Program libraries

Indexed Sequential	BISAM	QISAM	A sequential organization with an index permitting both random & sequential access
Direct	BDAM		Typical random access
Telecommunication	BTAM	QTQM	Conversational and transmissions
Graphic	BGAM		CRT, microfilm, plotter, etc.

Table 4. Access Methods available by IBM

The data records of a data set can have any of six record formats:

F: simple fixed-length format used for card and print files

FB: fixed-length, blocked used on tape, disk, etc.

FBS: standard fixed-length, blocked, most efficient record format for disk. "Standard" means that every block contains tha same number of records except possibly the last, which may contain fewer.

V: variable-length. Each record is preceded by a byte-count (non-data) length indicator used to save space.

VB: variable-length, blocked. Each physical record is preceded by a length indicator, and each logical record is also prefixed with a length indicator

U: undefined format. Records are considered to be unblocked.

Accessing of files through the operating system will be shown in the subsequent discussions. In order to find (or set up) the right data set, the operating system must know some information about a file, such as: file organization; block size, record size, blocking factor; label description; space allocation; device residence; disposition of the file. There are three times when file information is presented to the operating system:

(1) in the program using data control block (DCB) macro instruction or by DD (Data Definition) statement

(2) at load time by searching for external segments in the library

(3) at execution time by opening the temporary or permanent user files.

The sequence of operations followed in setting up the data control block is roughly as follows and is summarized in Fig. 5. While the job scheduler (control program that selects the next job to be processed) is in control, the job control statements are read and a job file control block is set up with the information from each DD statement. Each job control block is identified by a ddname (data definition name in DD card) and contains at least a unit specification or a data set name (DSN). While the program is loaded into main storage for execution the expansion of the DCB macro-instruction made by the assembler becomes the skeleton of the data control block. Then during the execution, the user's program "opens" the other user file and the open macro brings in the label on a data set. Label information is placed in its own block, called the Data Set

Control Block or DSCB. The information in the DSCB is then merged with the JFCB and the DCB to form an updated DCB so that I/O macros can be properly managed by the access method.

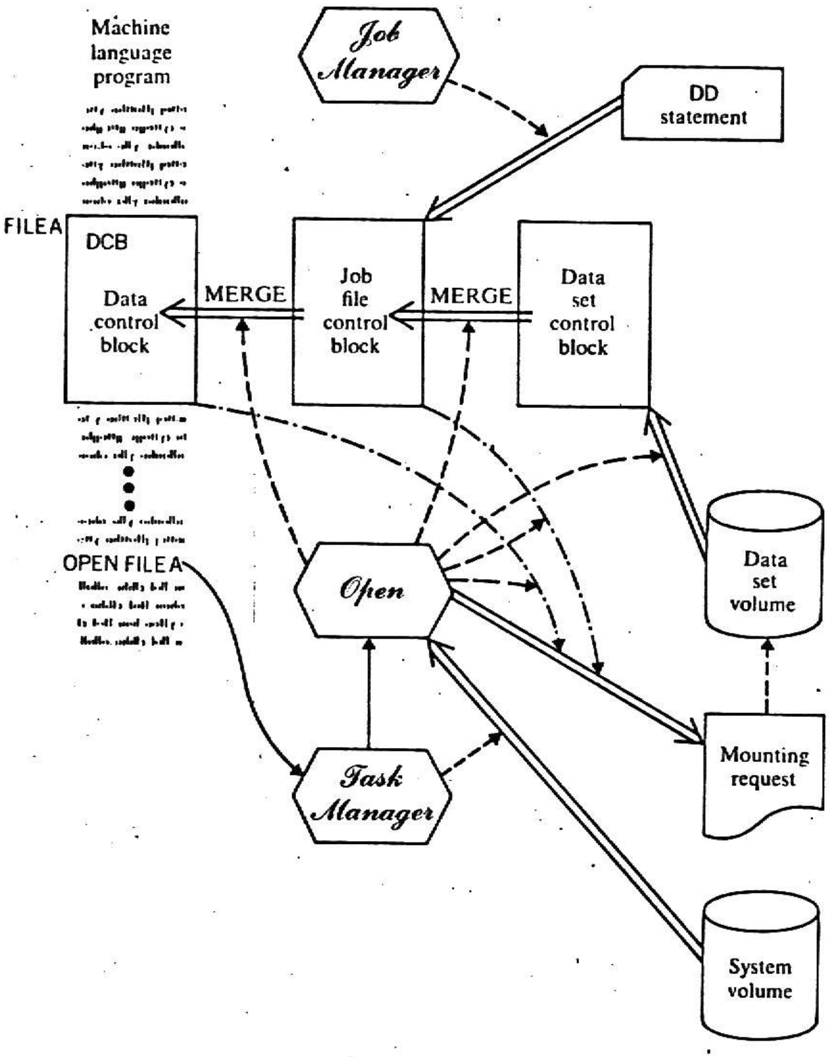
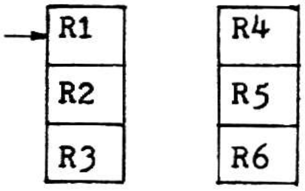


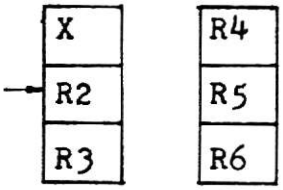
Fig. 5 Finding OPEN in the program starts a set of operations which fully establishes the DCB.

The access method deals with files and buffers. A description of each file managed by the access method is furnished as a data control block (DCB) supplied by the program. The sequential access method build and manipulate buffers, but a random access method the program builds its own buffers and manages buffering. A simple example of automatic buffering with the sequential access method is given below: Suppose we open a sequential file with OPEN (A, number of buffers=2, mode =input); the blocking factor is 3.

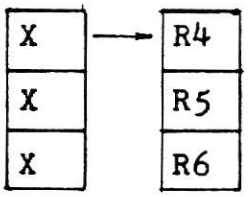
buffer1 buffer2



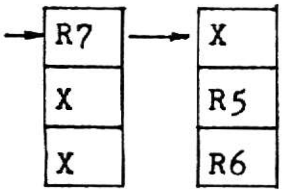
--- As soon as the file is open, the operating system fills the buffers which are established in the main storage with blocks 1 and 2.



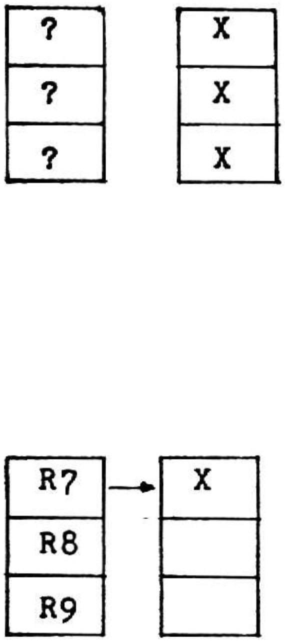
--- Upon execution of a READ operation by software, the first record R1 is copied into program's working space and the pointer is advanced.



--- After two more READ operations, the pointer is advanced to the first logical record of the second buffer. At this time concurrency of I/o begins.



--- The data channel (hardware) is filling the buffer1 while the I/O program returns the control to program which may READ record 4.

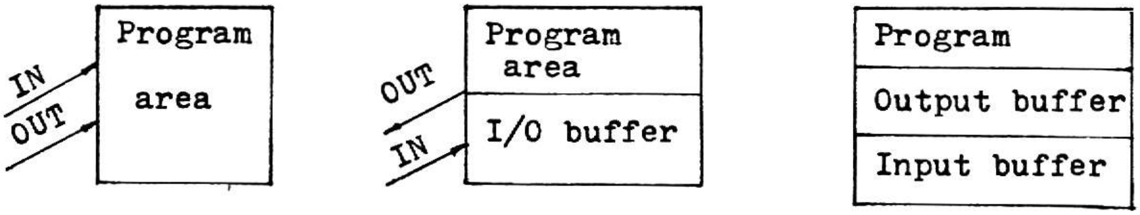


--- At this point two possibilities exist: if the program issues READ R7 before the buffer1 is filled with R7, then it is input-output bound; in this case by increasing the number of buffers, better concurrency of I/O can be performed and shorten the execution time.

--- The other possibility is that the data channel finished filling buffer1 before buffer2 is read, this case is called compute bound or processor bound (CPU bound); in which case to increase the number of buffers does not help the speed of execution of a program.

Fig. 6 A double buffered input stream with blocking factor 3

The above example is illustrated differently to show I/O and computation of program overlap:



(a) no overlap

(b) overlap of Input or Output with Compute

(c) overlap of I/O with Compute

Fig.7 Simple models of Overlap

In addition to providing logical record/buffer supervisory routines and physical record/device supervisory routines, the operating system also includes other services such as password protection of confidential or sensitive data. Password protection requires the operator to make a secret response at the time of OPENing a file, or else the program is not allowed to access the protected data set.

IV. COMPARISON OF COMMON FILE ORGANIZATION TECHNIQUES

A number of people in the field of information processing have proposed different classifications of file organization based upon type, structure, storage media, software-hardware components, and so on of a file. But the most common file organization techniques are those supported with software by computer vendors; these are normally part of the operating system.

Strict Sequential	Indexed Sequential	Direct or Random
IBM	IBM	IBM
RCA	RCA	RCA
CDC	CDC	
UNIVAC	UNIVAC	UNIVAC
Burroughs		
NCR	NCR	NCR
GE		GE
Honeywell	Honeywell	Honeywell

Table 5. --summary of the file organization techniques supported by the eight largest computer vendors

Note: All the computer vendors in the Table 5 have slightly different features in organizing their indexed and direct files. This might have affected the classification. (This table is from 1969 FJCC. pp 417.)

SEQUENTIAL FILE ORGANIZATION

The oldest and most common file organization technique is the strict sequential file organization. In a sequential file, records are organized solely on the basis of their successive physical locations in a file. The magnetic tape is the most common example of sequentially organized data files.

Records may be fixed or variable-length, blocked or unblocked or undefined. The records may be formatted with or without keys. If the file is processed sequentially all the time, there is no point in formatting with keys. This is normally the case with this type of organization. But if for some reason, there is some amount of random accessing of records in the file, records should be formatted with keys so that they can be located more quickly.

The amount of storage required for this type of file organization is simply enough to hold the records in the file. Sequential organization is used primarily for tables and intermediate storage rather than master files unless a high percentage of processing of the master file is sequential.

Random access of a sequential file is very inefficient. The slowest way is to read the record sequentially until the desired one is located. A random access by sequential search takes less time if the records are formatted with keys; the user might construct his own program for binary search of the file in order to determine in which small section of the file the discussed record is located. For example a binary search with an eight cylinder file formatted with keys is as follows: The last record in cylinder 4 is read and compared with search argument. Then depending upon the results, the last record in cylinder 2 or 6 is read and compared. Again depending upon the result, cylinder 1 or 3 or 5 or 7 is tested. This last comparison then indicates which cylinder should be searched in full.

A difference between sequential access and random access is that sequential access devices usually have no addresses to distinguish an individual record or a block whereas random access devices have such addresses. Another important difference is that the sequential access methods are used when the user wishes the operating system to perform anticipatory buffering and scheduling of I/O using the buffers requested by the user; in this case I/O operations and processing can be overlapped. A sequential access method also performs automatic blocking and deblocking if the records are blocked (see Fig 8.).

The direct access methods are used when the programmer wants neither automatic blocking nor anticipatory buffering. Thus all those functions are user's responsibility.

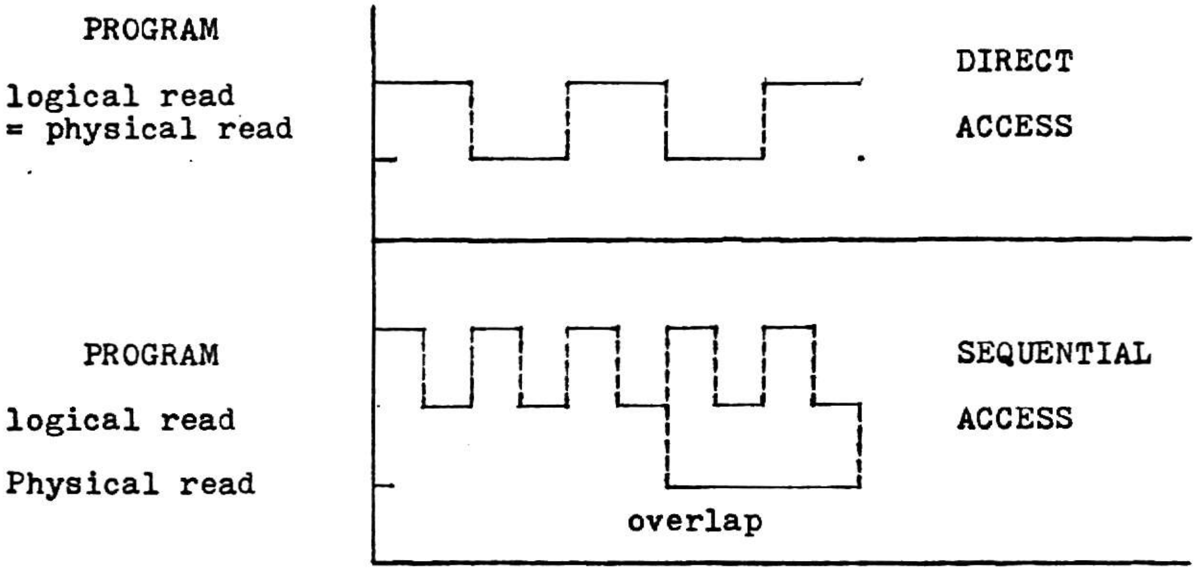


Fig. 8. I/O Operations of Direct Access vs. Sequential Access

INDEXED SEQUENTIAL ORGANIZATION

An indexed sequential file is a sequential file with indexes that permit rapid access to individual records as well as rapid sequential processing. Associated with each record is a key, which is part of the record, and the records are arranged so that they keys are in ascending order. Also a set of indexes is kept, showing the locations of blocks of the data set whose last records are associated with various keys.

When access to a particular record is desired, its key is looked up in the index and the sequential retrieval of records begins with the block containing the record. Thus the file is sequentially organized, but the indexes facilitate jumping into the desired block. This file organization permits fairly easy and automatic appending to or deleting from files and yet fairly fast access.

The demand for storage space for an indexed sequential file is larger because of the additional space required for the indexes. Also an added inefficiency in the use of storage space is the typical requirement for overflow areas to permit insertions in the main files.

Indexed sequential files should be on direct-access storage devices i.e. disk, drum, but not tape. The operating system supports only fixed-length blocks.

An indexed sequential file may have three distinct areas: a prime area, indexes, and overflow area. Storage allocation for the prime area is required but index and independent overflow areas are optional.

The prime area is the area in which records are written when the file is first created or subsequently reorganized. Additions to the file may also be written in the prime area. In the prime area, the records are in key sequence and they may be blocked or unblocked. If blocked, each logical record contains a key and the key area contains the key of the highest record in the block (see Fig. 9).

There are two or more levels of indexes, and they are created and written by the operating system when the file is created. It uses the master directory called a cylinder index and a set of subdirectories called track indexes (see Fig. 10).

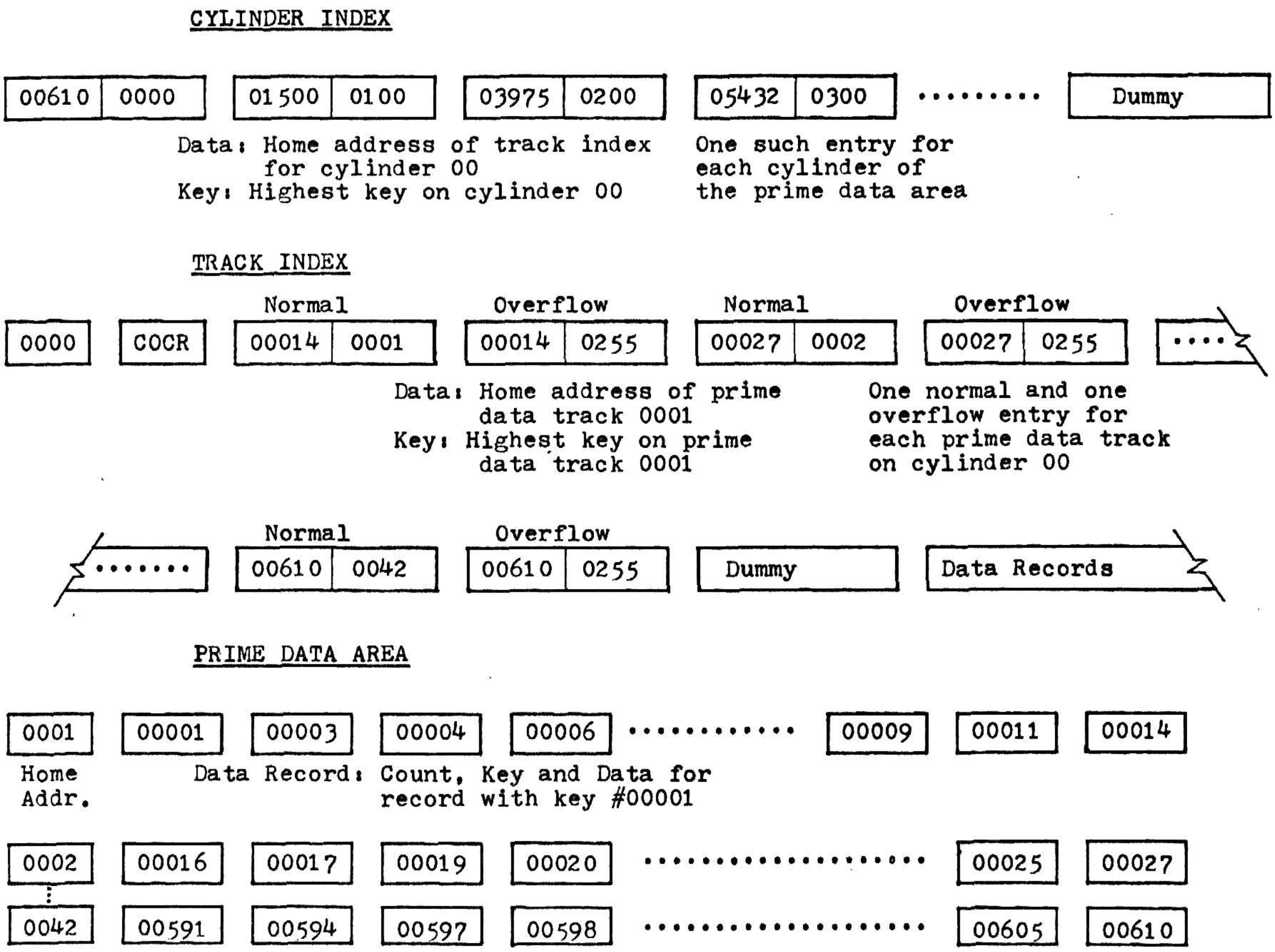

CYLINDER INDEX
00610
0000
01500
0100
03975
0200
05432
0300
Dummy
Data: Home address of track index for cylinder 00
Key: Highest key on cylinder 00
One such entry for each cylinder of the prime data area
TRACK INDEX
0000
COCR
Normal
00014
0001
Overflow
00014
0255
Normal
00027
0002
Overflow
00027
0255
Data: Home address of prime data track 0001
Key: Highest key on prime data track 0001
One normal and one overflow entry for each prime data track on cylinder 00
Normal
00610
0042
Overflow
00610
0255
Dummy
Data Records
PRIME DATA AREA
0001
00001
00003
00004
00006
00009
00011
00014
Home Addr.
Data Record: Count, Key and Data for record with key #00001
0002
00016
00017
00019
00020
00025
00027
0042
00591
00594
00597
00598
00605
00610

Fig. 9. An indexed sequential file with no additions

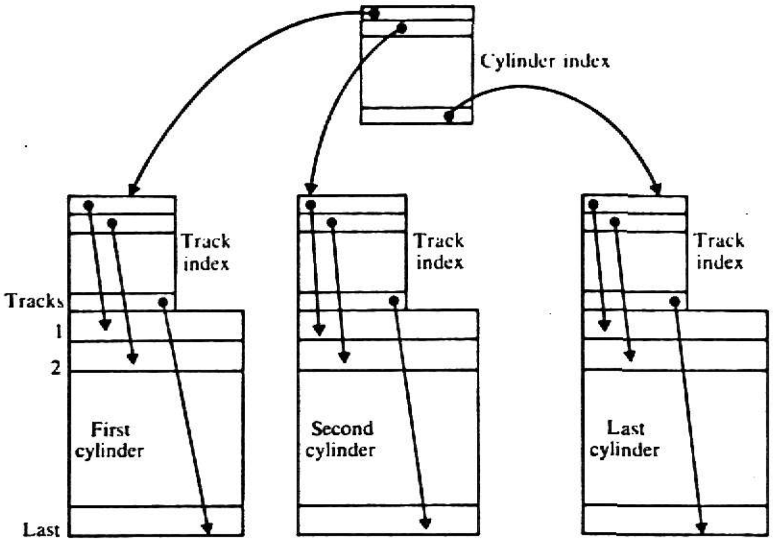


Fig. 10 The cylinder index points to track indexes, one for each cylinder in the file.

The cylinder index is a higher level of index and its entries point to track indexes. There is only one cylinder index for the file, and this may be on different disk than the rest of the file.

The cylinder index consist of one entry for each cylinder in the prime area fɔllowed by a dummy entry. The key area contains the key of the highest record on the cylinder to which the entry points. The data area contains the home address of the track index for that cylinder.

If cylinder index becomes too long, a master index may be created to index it. Usually if the cylinder index exceeds four full tracks, a master index is requested. It is also possible to get a second and third-level master index when necessary. The area reserved for the master index and cylinder index is called the index area.

Track index is the lowest level index and always present. Its entries point to data record. There is one track index for each cylinder in the prime area. It contains a home address entry and an overflow pointer to the first available block. There is one four-part entry in the track index. The first two parts are called the normal entry: first part is the key of highest record on the track, the second part is the home address of the first prime data on the track. The third part is the key of the highest record for that track in overflow area; the fourth is the location of the first overflow record from this track (Fig. 9).

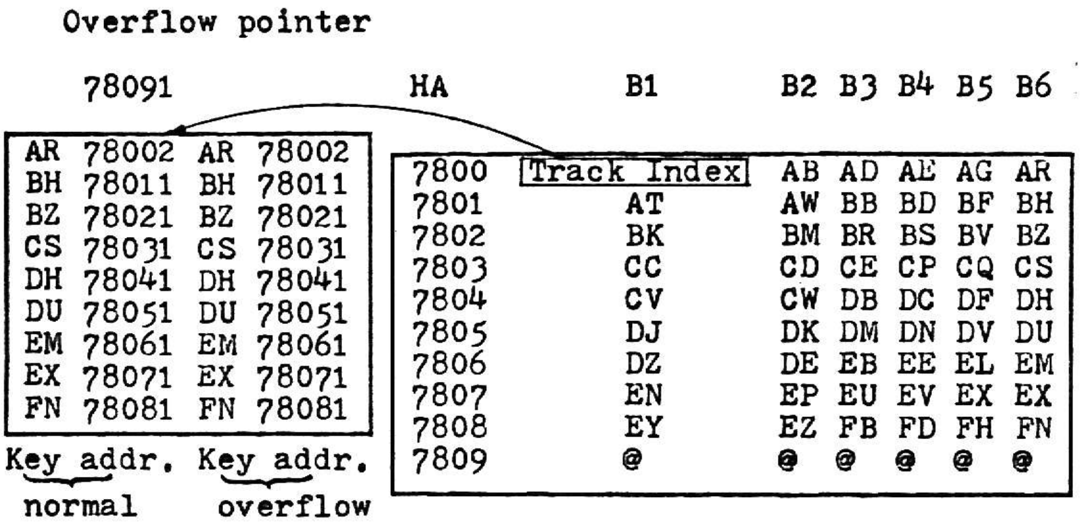
One of the principles of the indexed sequential access method is to keep the records in order. Therefore insertion of records requires moving the records to keep the order and also updating the overflow area. A deleted record is simply flagged as inactive to save a time-consuming moving operation; the only time one is eliminated is when it is forced off a track to an overflow area or the whole file is rewritten.

To illustrate insertions and deletions of records to an indexed sequential file, see Fig. 11 and 12. The first insertion of a record with a key, BP, works like this: Since BP is less than FN, the cylinder index indicates it is in cylinder 78. Next the track index is examined; it shows BP is greater than AH and BH but less than BZ so it belongs to track 2. In track 7802 BP is greater than BM but less than BR so it is inserted in the third position, and BR, BS and BV are moved down to the next positions and BZ is moved to overflow area. The overflow pointer is advanced by 1 and the key of the normal track entry is also changed to contain the highest record in the changed track, BV. The track index overflow pointer states where BZ is: 78091. After inserting several records and deleting one record, the result is shown in Fig. 12.

In managing overflow areas, a cylinder overflow area and an independent overflow area can be reserved either singularly or in combination. A cylinder overflow area is reserved by allocating a specified number of tracks on each cylinder for overflow. This has the advantage of minimizing access time since no additional access-arm movement is necessary, however, this overflow area can not be shared. An independent overflow area which may not be adjacent to the prime area can be shared by all cylinders, conserving storage space at the expense of an increased access time. The best method is reserving reasonable space for both overflow areas.

The sequential access method is used when reading and updating the record in key sequence; the operating system provides anticipatory buffering and automatic blocking.

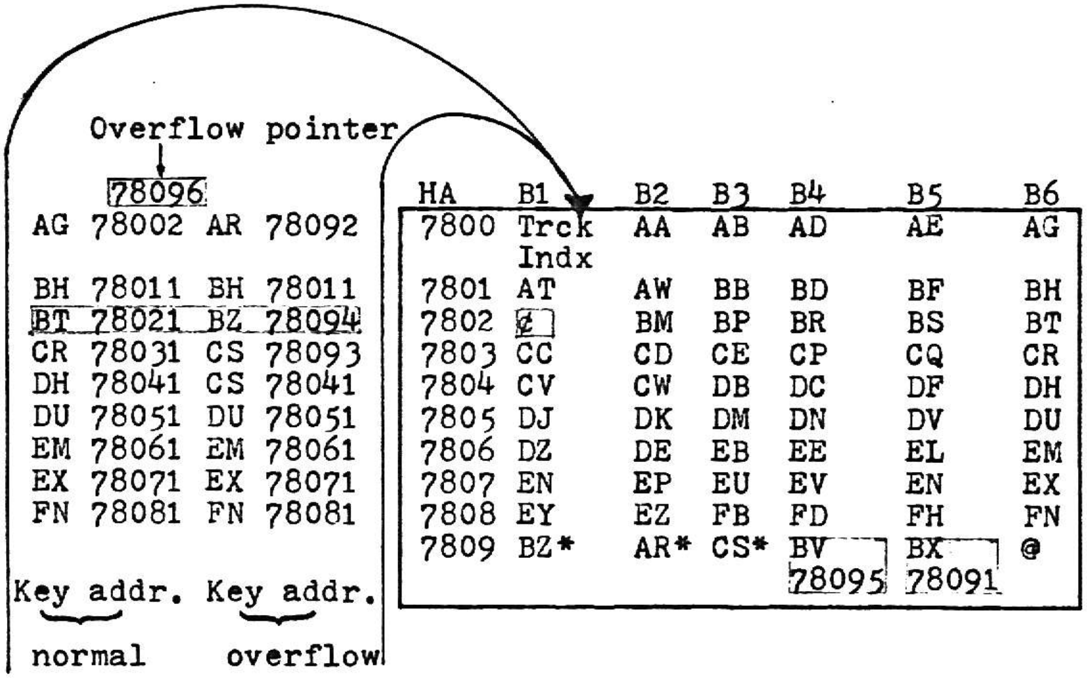
The random access method for indexed sequential files is used when adding, reading or updating records directly; the operating system writes the new record, rewrites existing records, index entries, and link fields as required.



normal		overflow	
AR	78002	AR	78002
BH	78011	BH	78011
BZ	78021	BZ	78021
CS	78031	CS	78031
DH	78041	DH	78041
DU	78051	DU	78051
EM	78061	EM	78061
EX	78071	EX	78071
FN	78081	FN	78081

HA	B1	B2	B3	B4	B5	B6
7800	Track Index	AB	AD	AE	AG	AR
7801	AT	AW	BB	BD	BF	BH
7802	BK	BM	BR	BS	BV	BZ
7803	CC	CD	CE	CP	CQ	CS
7804	CV	CW	DB	DC	DF	DH
7805	DJ	DK	DM	DN	DV	DU
7806	DZ	DE	EB	EE	EL	EM
7807	EN	EP	EU	EV	EX	EX
7808	EY	EZ	FB	FD	FH	FN
7809	@	@	@	@	@	@

Fig. 11. Contents of cylinder 78 with the track index expanded and placed on the left margin. Only block keys appear in the table.



normal		overflow	
AG	78002	AR	78092
BH	78011	BH	78011
BT	78021	BZ	78094
CR	78031	CS	78093
DH	78041	CS	78041
DU	78051	DU	78051
EM	78061	EM	78061
EX	78071	EX	78071
FN	78081	FN	78081

HA	B1	B2	B3	B4	B5	B6
7800	Trck Indx	AA	AB	AD	AE	AG
7801	AT	AW	BB	BD	BF	BH
7802	¢	BM	BP	BR	BS	BT
7803	CC	CD	CE	CP	CQ	CR
7804	CV	CW	DB	DC	DF	DH
7805	DJ	DK	DM	DN	DV	DU
7806	DZ	DE	EB	EE	EL	EM
7807	EN	EP	EU	EV	EN	EX
7808	EY	EZ	FB	FD	FH	FN
7809	BZ*	AR*	CS*	BV 78095	BX 78091	@

Fig. 12. Contents of cylinder 78 after appending BP, AA, CR, BT, and BX have been added in that order. Then BK is deleted.

DIRECT (RANDOM) ORGANIZATION

Direct data organization permits each record to be accessed directly without regard to its position relative to other records. Therefore random access provided by the direct file organization is generally used for files where the access time to an individual record should be kept to an absolute minimum, as in real-time processing, or when only a small percentage of a large file is required at any one time, such as a question-answering service.

Each record of a direct organized file contains a key that describes the record's location in the data set. The addressing can be direct or indirect as the user chooses; since the user is largely responsible for the logic and programming required to locate records. Some commonly used methods of developing operating system to provide random organization will be introduced here.

With direct addressing, every possible key of a record in the file converts to a unique address, therfore, it is possible to locate any record in the file with one seek and one read. One way of setting up a direct addressing scheme is to use the key of a record as the address. In order to do this, the record must be fixed-length and key must be numeric. By dividing the key by the number of records per track; the quotient equals the relative track address and the remainder equals the record number. This method requires minimum time to locate a record, but a possible disadvantage is the wasting of a large amount of unused storage due to the fact that for every key a location must be reserved. For example, employee number ranges from 1 to 999 but only 600 of the possible 999 are actually assigned. Blocking of records can be used to save storage, but the user is responsible for all blocking and deblocking routines.

Another method of direct addressing is to assign an address to each record and maintain a cross-reference list of keys and assigned addresses. The list may be a printed one or it may be a file recorded on direct access storage devices. Although a record can be accessed directly when the address is known, time is required to look up the list. Indexed sequential organization is a variation of this method.

Indexed addressing is used when the range of keys of a file includes a high percentage of unused ones so that direct addressing is not feasible. With indexed addressing, keys are converted into a smaller range of record addresses. In selecting a conversion technique two objectives must be considered: every possible key must be converted to a certain range of address, and addresses should be evenly distributed so that few collisions (two or more keys map into same address) occur. An example of index addressing is the Linear Quotient Method which works as follows: Suppose operating system has allocated disk space for N records where N is a prime interger. Given a key K of a record, operating system computes its address ho relative to the start of the file by

$$\frac{K}{N} = Q + \frac{ho}{N} , \qquad 0 \leq ho < N$$

If that address (ho) was already occupied then the search continues to compute address (h_{i+1}) by

$$h_{i+1} = (h_i + Q')\bmod N$$

$$Q' = \begin{cases} 1 \text{ if } Q \equiv 0 \bmod N \\ Q \text{ otherwise} \end{cases}$$

Another example of the indirect addressing method is Radix Transformation. With this method, the key is transformed to a different radix or base; excess digits are discarded, leaving an address of the required length. The following is an example of converting a key of 42351 to radix 11 to provide four address:

$$(4x11^4) + (2x11^3) + (3x11^2) + (5x11^1) + (1x11^0)$$
$$= 58564 + 2662 + 365 + 55 + 1 = 61645$$

Use 1645 as the relative address.

With an indirectly addressed file, activity of each record has a significant effect on time to locate records. Therefore, if the records of the file have uneven distribution of activity, the file should be sorted into the activity sequence.

For both direct and indirect addressing technique, additions are done through overflow areas. Deletion can be done by simply flagging a record as inactive and omitting that record when the file is reorganized. One of the methods of handling overflow records is the chaining method. With this method, one record on each track is used to point to the overflow track so that next track in the chain can be used when the track is full.

For direct organization, the operating system does not provide automatic buffering and overlap of I/O with processing.

However, if the user knows in advance which record he wants next, he can program these functions.

A direct data set can, of course, be processed sequentially if the user knows the key of each record and the orders of keys sequentially. This operation is very inefficient since each record must go through key-address transformation. However, for performing random access of records, this is the fastest file organization technique.

PARTITIONED ORGANIZATION

A partitioned data organization is made of a directory and number of sequential files which are members of this data set. This data set can be stored only on direct-access devices.

Each partitioned data set begins with a directory recorded in the standard format. This directory contains the name and location of each member as shown in Fig. 13.

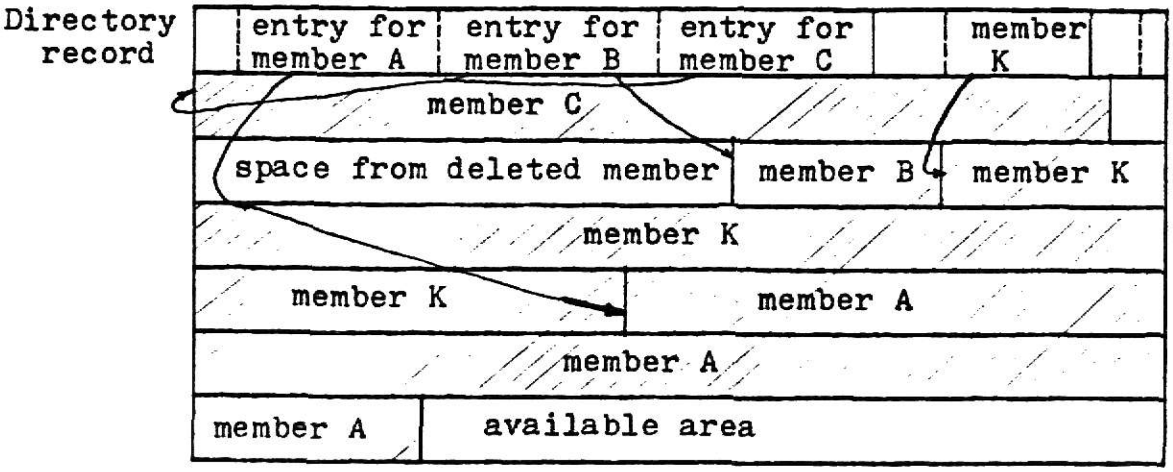


Fig. 13 Partitioned Data Set

Each member has a unique name, 1 to 8 characters long, and is stored in a directory in alphameric collating sequence by name. When a member has more than one name, the additional names are called aliases and a signal bit indicates that is an alias. Every member of any individual partitioned data set has the same record format and organization. The records in a member of a partitioned data set are usually fixed length, blocked or unblocked. Once a member is selected, it is processed like an ordinary sequential data set.

Like other data sets on direct access devices, a partitioned data set is identified by a DCB recorded on the same device. This DCB contains information about the data set name, hardware address, organization of the data set and record format; it can also include a security password, date of creation, and retention period or expiration date. One of the valuable features of a partitioned data set is that it needs only one DCB no matter how many members it contains.

Partitioned data sets are ideally suitable for storing program libraries. Usually members of a partitioned data set are related programs, but this is not necessarily true.

The programmer is responsible for the construction and use of both the partitioned file and its index. To create this file, the programmer asks for space for the data set on one volume, he should also leave room for additions. To make additions to the file, he writes a new member in a proper space by random access methods and adds the member's name to the index.

Deletion can be accomplished by removing the member name from directory entries, but the storage area that it occupied does not become immediately available.

In a partitioned data set, the index location can be passed to the cataloging software to be entered both in the Volume Table of Contents (VTOC) and catalog automatically.

V. FILES, DIRECTORY HIERARCHY

In the previous sections, methods of organizing files were discussed. This section deals with how to retrieve desired files once they are put into the computer. As an example, how the IBM/360 operating system finds a file will be shown here. OS 360 requires two resources: the VTOC and the file catalog to assist in finding the desired information. The VTOC appears on every volumn and list the location of all the files in that volume. The catalog, which is always available to the system by residing on system residence volume, lists all the cataloged files available to the computer, even those on media not presently attached to the system. Should the volume required be absent or on a disconnected device, the catalog conveys this to the operator.

As summarized in Fig. 14, if a data set is cataloged, the user may give the data set name in his DD statement, then the operating system will find it by using the information in the catalog. If the data set is not cataloged, the user directs the operating system to it by designating an input or output unit, such as a magnetic tape drive.

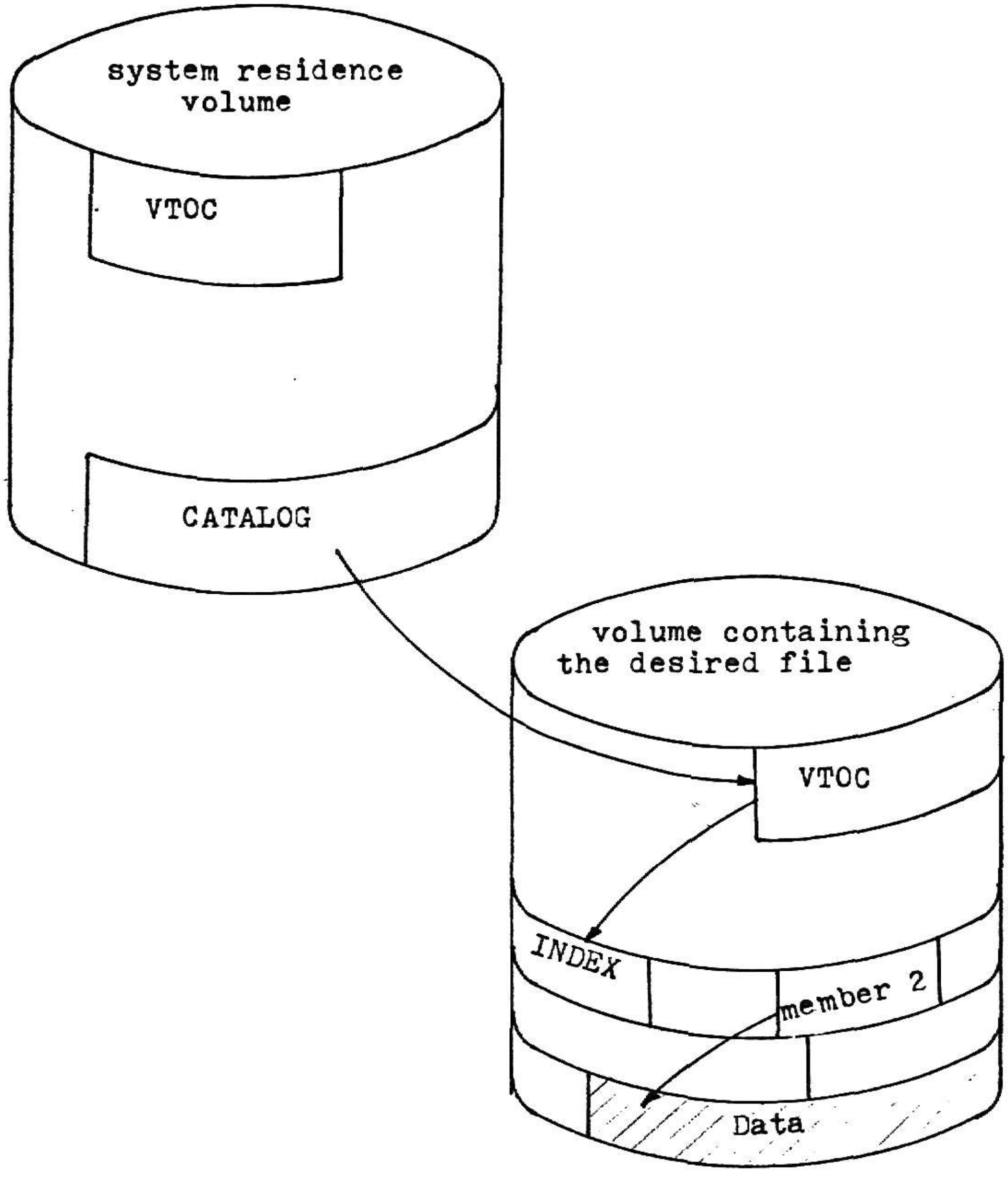


Fig. 14. Structure of Catalog, showing Path used to find a File named member 2 of a Partitioned Data Set.

In order to find any member of the data set, the operating system needs a file directory, in which files are cataloged or indexed as the user chooses. On many systems, the entries entered into a directory may include other directories as well as files, thus giving each user the ability to create a directory tree with his collection of filed procedures and data bases.

For our model, the data base consists of three subcollections: (2)

SYS --- The collections of system procedures available to the user and system routines not available to the user.

LIB --- The collection of public subroutines and permanent user files

USER --- The set of temporary user files currently in execution.

This system is displayed in Fig. 15 as a tree structure showing the hierarchical relationships; circles represent directories and rectangles represent data set control blocks which contain information about where the file is located and how the file is structured in the secondary memory devices.

When the access of a file is requested, the file directory is searched, and the data set control block is copied into a job file control block (JFCB) which is contained in the context block of each process.

For the system to find a file, naturally, its name should be unique. A particular object is specified by a pathname that selects a path from the root of the directory hierarchy to the desired object (i.e. LIB.PAYROLL.E). File names can be nested as deeply as directories can be nested. However, within certain well-defined contexts, it is permissible to omit directory name (i.e. LIB.FORT) and use only the lowest name (i.e. Cos).

Generally a permanent file is referenced by using the fully nested name, while with temporary files only the lowest level name is specified.

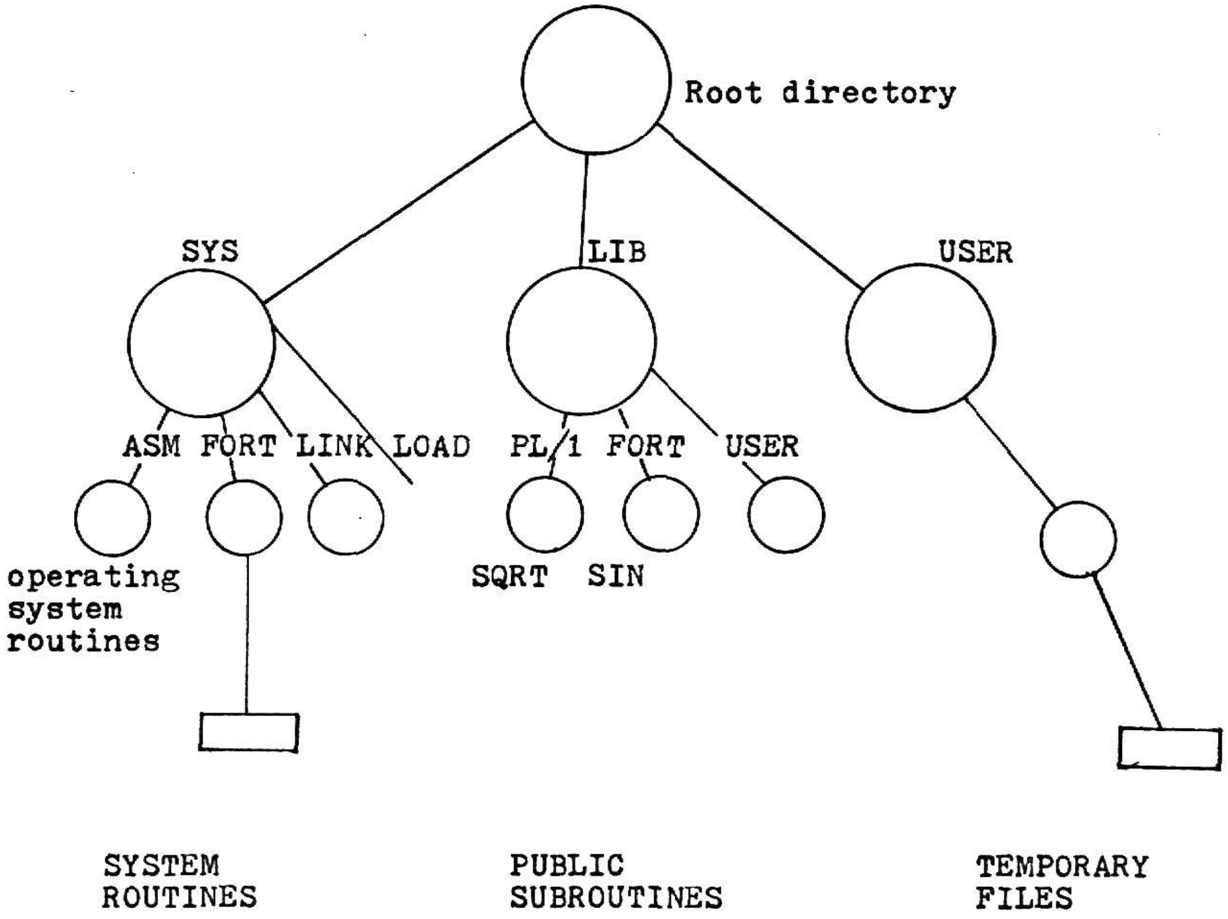


Fig 15 Hierarchy of files

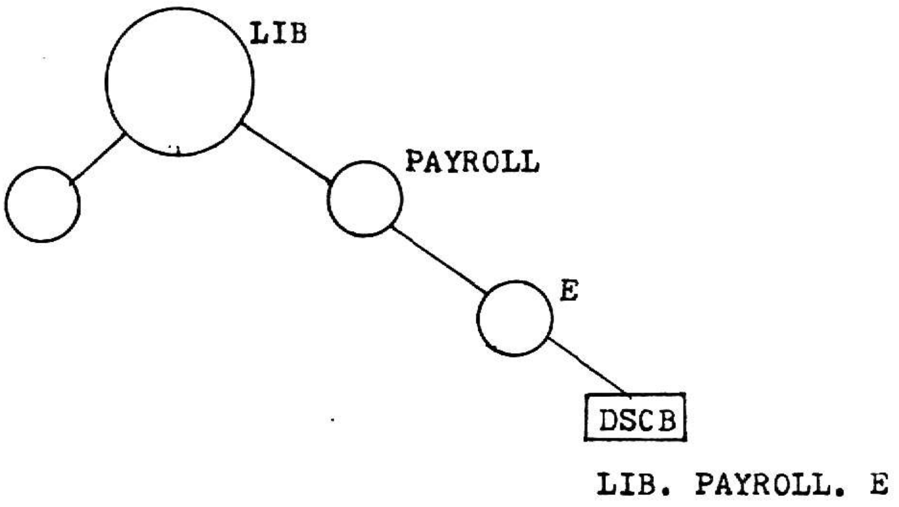


Fig. 16 Use of Nested Name

THE LIMITATIONS OF FILE SYSTEMS

As discussed in the previous sections, file directories provide long term storage for procedures and data, and may include protection and shared access control features. However, there are serious inefficiency problems: A procedure file must be loaded into the main address space prior to execution. This makes it impossible to activate procedures whose names are not known until retrieved from a data base or typed in from a terminal. If a file can be loaded dynamically at execution time, the steps of copying and linking can be eliminated.

These limitations of file systems can be relieved by using the segmented address concept. The next section presents the concept of segmentation as implemented in the segmented and paged virtual memory system.

VI. SEGMENTED & PAGED VIRTUAL MEMORY

Segmentation was introduced to allow

1) dynamic allocation of information to a hierarchy of memory devices
2) a means for programs to reference objects in a manner that is independent of their physical locations in memory
3) sharing of common procedure and data information by many programs.

In the segmented virtual memory system, a file is a linear array of data which is referenced by means of a segment name and a word name. In general, a user does not know how or on what device a file is stored.

A segment consists of contiguous logical locations that is a natural section of a program such as a subroutine or data file. When data, external to the procedure, is referenced, its location in the virtual memory is determined by a directory search. The use of file directories are combined with a large segmented address space.

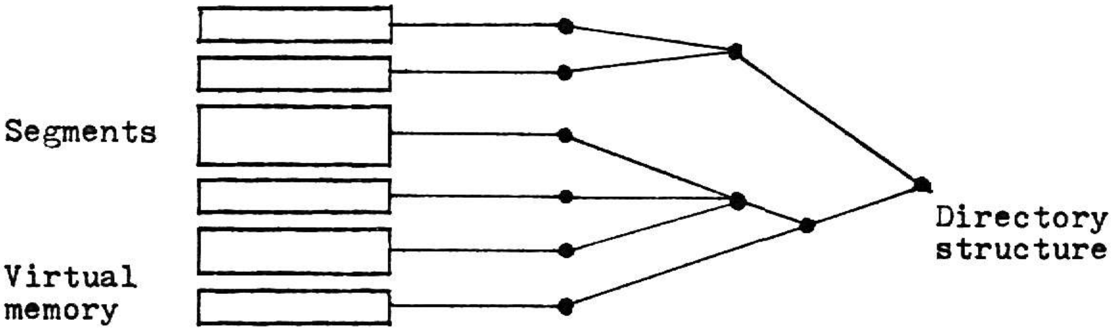


Fig. 17. Virtual Memory in a MULTICS Process

This address space is divided into a large number of segments, each being potentially large enough to hold any procedure or data file indexed in the file directories. Each segment is in turn broken into a number of fixed size pages. Reference by a computation to information in the address space is made by a pair of values (segment number, word number), segment number being assigned to procedures and data files when they first are referenced by a computation. This is called "making a segment known" to the process.

The addressing mechanism requires three memory references: the segment descriptor table, the page table, and the data itself. The segment descriptor table (SDT) is associated with each process and the SDT has an entry for each segment made known to the process. The page table is in main memory and there is one entry for each page of a segment and a corresponding entry shows either core or disk location of page.

Given a virtual address a = (s,w), the following operations are performed:

1. (s,w) is loaded into segment and word register
2. if segment table(s) = empty, then (missing segment fault); for this case, the segment has to be made known
3. if w length, then (overflow fault)
4. p = w/z where z is page size, p is page #
5. w' = w - pz where w' is line #

After (segment#, page#, line#) is computed, a simultaneous associative memory search is done to match (seg.#, page#) with (s", p"). If a match is found, frame address p' is combined with line# to reference memory directly.

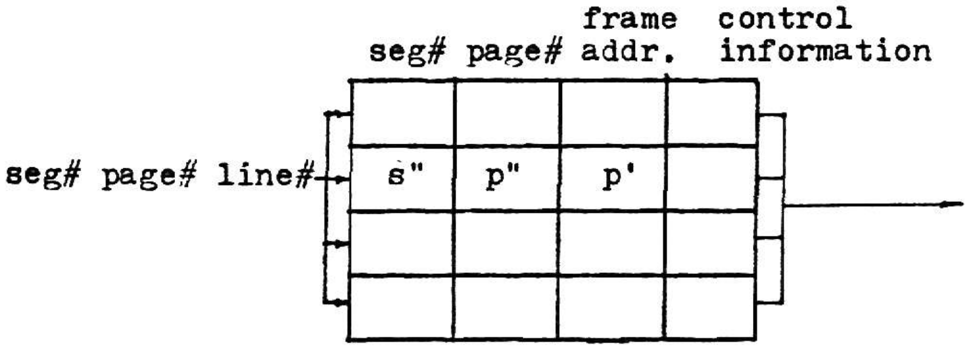


Fig. 18. Associative Memory Map

If a match in the associative memory does not occur, the page table is searched. If the page table is empty, then a missing page fault interrupt occurs, which initiates action to bring that page from secondary memory. Fig. 19 shows how this mechanism works.

Due to the high cost of an associative memory, only a few associative registers exist. Therefore there is a problem concerning what should leave when a new page is brought in. There are two replacement stategies for this: one is FIFO (First-in First-out) and the other; LRU (Least Recently Used). FIFO is the strategy to replace whichever is brought in first; the LRU strategy removes the ones which have not been used recently.

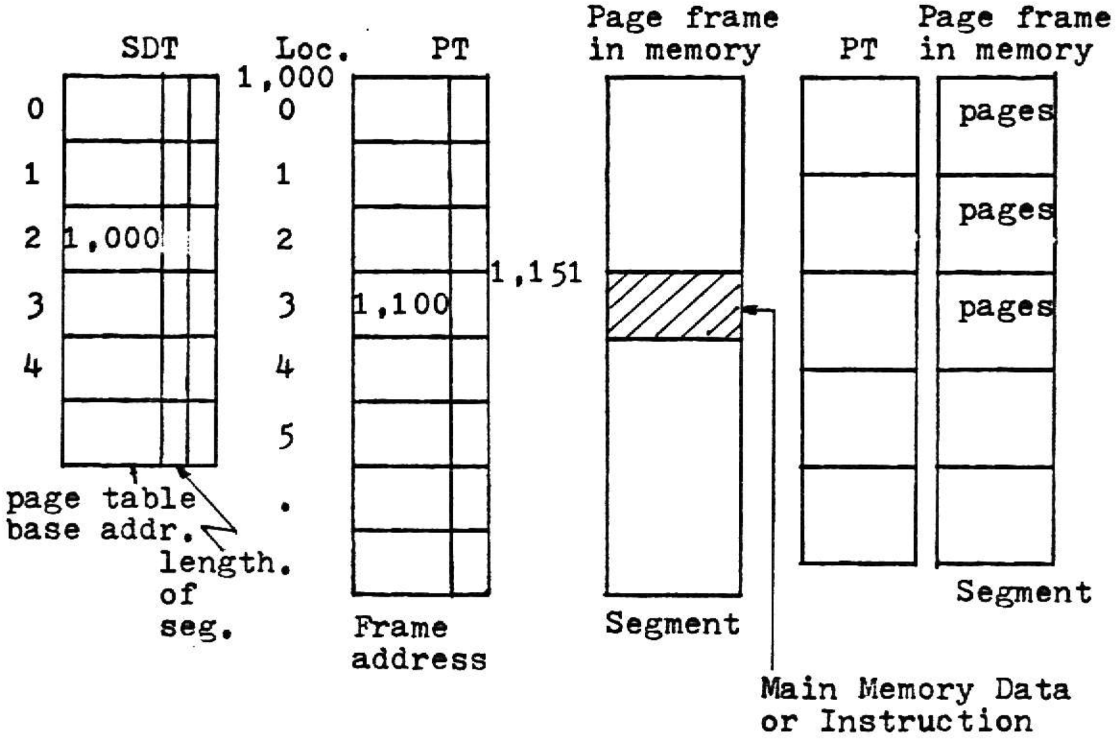
An example in Fig. 19 illustrates an addressing mechanism

	Segment #	Page #	Frame address	Control info.
4 ~~1~~	~~1~~ 2	~~10~~ 3	~~600~~ 1,100	0
2	4	6	800	0
3	6	1	500	1

Associative Memory

Process

Descriptor Base Register



a = (2,351)

page size: 100

therefore a = (2, 3, 51)
Seg. # Page # Line #

Fig. 19. Segment Table, Page Table and Address Map

of segment and paged virtual memory. Given a page size 100; and virtual address: ⟨segment⟩/[address] : a = (2,351) the following mapping is performed:

The address is divided by page size: thus a triple address (segment#, page#, line#) is divided from the given virtual address. Associative memory is searched to find a match of (segment#, page#), but fails. The segment table is then searched for segment #2 and the base address of that page table is found, which is 1,000. The next step is to go to location 1,000 and get the frame address of page #3, which contains 1,100. Finally line #51 is added to frame address 1,100 to obtain the correct main store location from which data or instructions are to be fetched. Under the LRU strategy, a reference to a page table also results in an updating of associative memory.

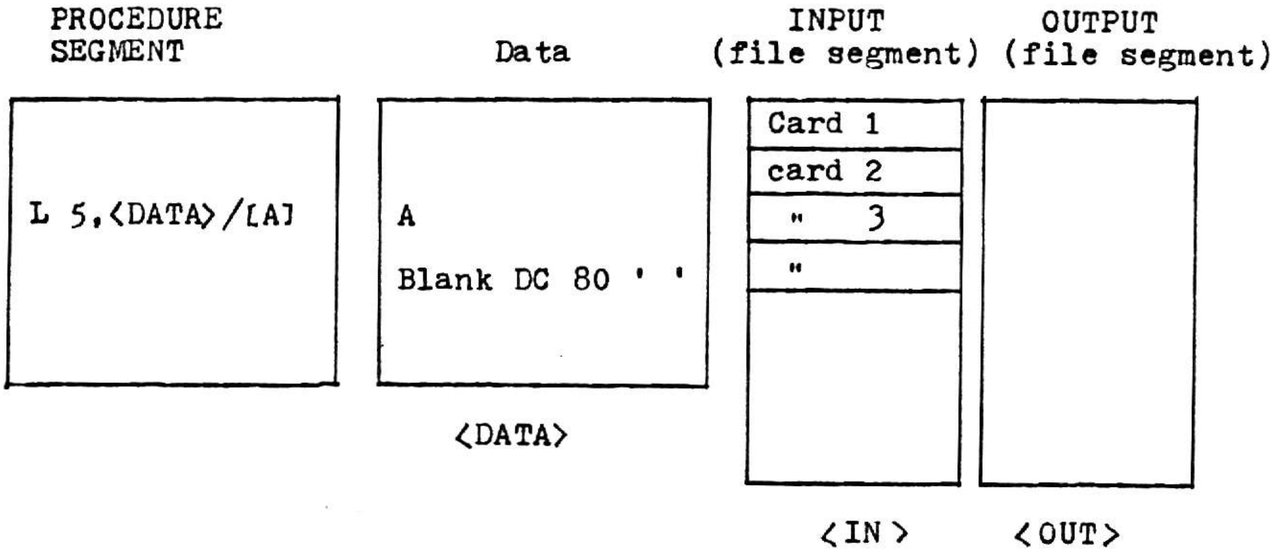
As was shown, once the segment is known the given procedure or data set is bound to a particular segment or address and, thereafter, the object may be referenced efficiently as a resident of address space rather than by a search of path name in the directory hierarchy.

VII. COMPARISON OF VIRTUAL MEMORY VERSUS FILE SYSTEM.

As pointed out in Fig. 20, the segmented and paged virtual memory differs from the file system of directories in the following ways:

First, the segmented address system eliminates the explicit buffering of information by the READ and WRITE functions.

Comparison of Simulated one-level memory vs. traditional file system



[Virtual Memory Instructions]	[File System Instructions]
L 5, DATA / A	load data from another file. work like READ
CLC <DATA>/<BLANK> IN /0(11)	compare data on different segment
LA 11, 80(11)	load address of next CARD; work like READ command
MVC OUT/0(9), <DATA>/[LINE]	move constant from one file to another. work like WRITE command
B <SUB>/ 0	branch to another subroutine

Fig. 20. Example Using IBM Basic Assembler Language

Whenever a particular element of data is required, it is referenced by virtual address, and the address mapping mechanism then brings the required page into main memory. Therefore with virtual addressing system, a user is not aware of the multiple levels of storage: he can access any file as if it were in the main store. Furthermore, with a file system, the user must know on which specific secondary memory device the file is located in order to access it; and the user must access the file with explicit Read and Write operations. Thus a virtual memory system lessens the user's burden at the expense of the operating system and requires more sophisticated hardware.

Second, segments may grow dynamically by allowing page space in auxiliary memory. Binding of segments to address space is performed at execution time, whereas with file system the procedure file should be loaded into memory and bound prior to program execution. Therefore a virtual memory system would be especially valuable for inter-active computer systems where a file name can be retrieved from a data base or typed in from a terminal at execution time; this file segment then will be loaded dynamically at execution time and activated without the user's knowledge of that file segment's physical location, i.e. specific disk or main memory.

Many of today's operating systems for example IBM and Honeywell support both a file system and a virtual memory system. In spite of their limitations, file systems which implement a long-term store external to the main store are still most widely used, while segmented name space, which provides a long-term store identical to the main store, is not so nearly wide spread as file systems. This is probably due to the additional hardware required to support a virtual memory system, recent years, however, more computer vendors are implementing segmented address space and this trend is likely to continue, at least until such time as main memory becomes less expensive than a complete virtual memory system.

REFERENCES

1. Rosove, Perry E., "Developing Computer-based Information System," John Wiley and Sons, Inc., New York, N.Y. 1967

2. Arthur E. Oldehoelf, and Wallentine, Virgil E., "Principles of Operating System," Unpublished lecture note, Iowa State University, Ames, Iowa

3.Flores, Ivan, "Data Structure & Management," Prentice-Hall, Inc., Englewood Cliffs, N.J. 1970

4. Knuth, Donald E., "Art of Computer Programming," Addison-Wesley Publishing Co., Vol. 1, 1969

5. Denning, Peter J. et al., "An Undergraduate Course on Operating Systems Principles," An Interim Report of COSINE COMMITTEE of the Commission of Education of the National Academy of Engineering, Washington, D.C. 1971

6. Senko, M. E. et al., "Data Structures and accessing in Data-Base Systems," IBM Systems Journal, No. 4, pp30-93 (1973)

7. Lefkovitz, David, "File Structures for On-line Systems," Spartan Books, New York, 1969

8. Gupta, Roger, "Electronic Information Processing," The MacMillan Co., New York,1971

9. Daley, R. C., and Dennis, J. B., "Virtual Memory Processes, and Sharing in MULTICS," CACM 11, (May, 1968), pp306-312

10. Daley, R. C., and Newmann, P. G., "A General Purpose File Systems for Secondary Storage," AFIPS Conf., Proc. 27 (1965 FJCC), pp213-229

11. Dennis, J. B., "Segmentation and the Design of Multiprogrammed Computer System," JACM 12, 4 (Oct. 1965), pp589-602

12. Brook, Frederick, and Iverson, Kenneth E., "Automatic Data Processing," John Wiley and Sons, Inc., New York, N.Y., 1969

13. Chapin, Ned, "Common File Organization Techniques Compared," AFIPS Conf., Proc. 35 (1969 FJCC)

14. Denning, P. J., "Third Generation Computer System," ACM Computing Surveys, Dec. 1969

15. Brown, G. D., "System/360 Job Control Language," John Wiley & Sons, Inc., New York, N.Y., 1970

16. International Business Corporation. " Information Management System/360 for IBM System/360 Description Manual"; Form GH20-0524, IBM Systems Reference Library, White Plains, New York 10604

17. International Business Corporation. "System/360 Generalized Information System Application Description Manual," Form GH20-0571, IBM Systems Reference, White Plains, New York 10604

18. International Business Corporation. "Introduction to IBM Direct-Access Storage Devices and Organization Method," Form GC20-1649, South Road, Poughkeepsie, New York 12602

19. International Business Corporation. "OS Data Management Service Guide," Form GC20-3746, White Plains, New York 10604

20. International Business Corporation. "OS Data Management Macro Instructions," Form GC26-3794, White Plains, New York 10604

21. International Business Corporation. "OS Data Management for System Programmers," Form GC28-6550, White Plains, New York 10604

22. Jordain, Pjilip B., "Condensed Computer Encyclopedia," McGraw-Hill Book Co., New York, 1969

23. Hellerman, Herbert, "Digital Computer System Principles," McGraw-Hill Book Co., New York, 1967

24. Morris, R., "Scatter Storage Techniques," CACM 11 (Jan. , 1968)

25. Saltzer, J., and Gintell, J., "The Instrumentation of MULTICS," CACM 13, Aug., 1970

ACCESS AND ORGANIZATION OF SECONDARY MEMORY DEVICES

by

INJA CHUN

B.A., Ewha Woman's University, 1966

AN ABSTRACT OF A MASTER'S REPORT

submitted in partial fulfillment of the requirement

for the degree

MASTER OF SCIENCE

Department of Computer Science

KANSAS STATE UNIVERSITY

Manhattan, Kansas

1973

ABSTRACT

This report discusses the general purpose file system on the secondary memory devices. Most common auxiliary storage media: tape, disk, drum are discussed in terms of access time, general availability of space, relative cost of storage, and factors that affect the efficiency of each device.

Comparisons of different file organization techniques: sequential organization, indexed sequential organization, partitioned organization and direct or random organization are presented based on their appropriate usage, storage requirement, file maintenance, and sequential - versus - direct access methods for each organization.

The concept of segmentation and its addressing mechanism is presented to show the limitations of the file system.