ASYMMETRIC MULTIPROCESSORS

by

J. S. Whitlock, Jr.
Department of Computer and
Information Science
University of North Carolina
Chapel Hill, North Carolina

21 November, 1968

Term Paper
Info. Science 222
Theory and Design of Control Programs

Asymmetric Multiprocessors

Asymmetric Multiprocessors

I. Introduction

The use of two unequal CPU's, interconnected by shared main storage and/or external storage, with the operations of both controlled by an integrated operating system has been a frequent solution to the problems associated with high-thruput computation centers. The processors involved have usually consisted of a primary processor with great computational power, paired with a support processor oriented toward i/o capability. The interconnection of the two CPU's has usually been via a shared i/o channel (i.e., a main storage-to-main storage connection via a modified i/o channel). Other possible interconnections are shared core storage and shared i/o device storage (e.g., a shared disk drive). The main feature that distinguishes the class of systems we are concerned with is the use of an integrated operating system, functionally divided between the two processors. Control information and data is passed between the operating system routines in the separate processors allowing coordination of their activities. The main design objectives of these systems are:

1. Maximize the utilization of the primary computer for computational tasks, usually involving word operands (in particular floating point operands) rather than character operands.[1] The system should minimize i/o waiting and

[1]Relative to the support processor, the primary processor is assumed to be powerful at arithmetic operations. All efforts are concentrated on keeping the primary computer busy on tasks that use this power.

system overhead functions (e.g., job accounting) on the primary processor.

2. Maximize utilization of all sytem components. For example, printers and card reader/punches should be operated efficiently; the system should permit advance mounting and deferred dismounting of tape reels without blocking the primary computer.
3. Minimize operator intervention and interaction with the primary processor. (This implies mechanization of all possible decisions.)
4. Inclusion of priority processing as an essential feature of the system. This includes priority scheduling of both execution and printing/punching.

The remainer of this paper will describe the organization of four systems employing asymmetric multiprocessing techniques using IBM computers. The first two systems, DCS and ASP, are multiprocessors; the second two systems are not multiprocessors, but they employ multiprogramming techniques for functions that could be logically associated with a separate processor. They are the CLASP (or LASP) and HASP systems.

II. Approaches to Asymmetric Multiprocessing

A. Direct Couple System (DCS)

1. Background

The Direct Couple System utilizing an IBM 709x[2] as a primary processor and an IBM 704x as a support processor succeeded in reducing turnaround for short 709x jobs to less than 2 hours while thruput increased under DCS to 800-1000 jobs/day. (This was more than a 100% improvement over existing 709x systems.) Prior to the introduction of this system in 1964, the typical mode of operation for a 709x installation was to prepare input tapes and print/punch output tapes on an off-line 1401 computer. DCS was able to reduce turnaround for short jobs by queuing input jobs on disk and scheduling their execution based on estimated running time. Similarly output was queued on disk and printed as soon as the printer became available. Thus, delays resulting from "batching" of jobs were eliminated. 709x thruput was improved by elimination of tape setup waiting and by an efficient i/o scheme that used the 704x in effect as an efficient (and intelligent) channel.

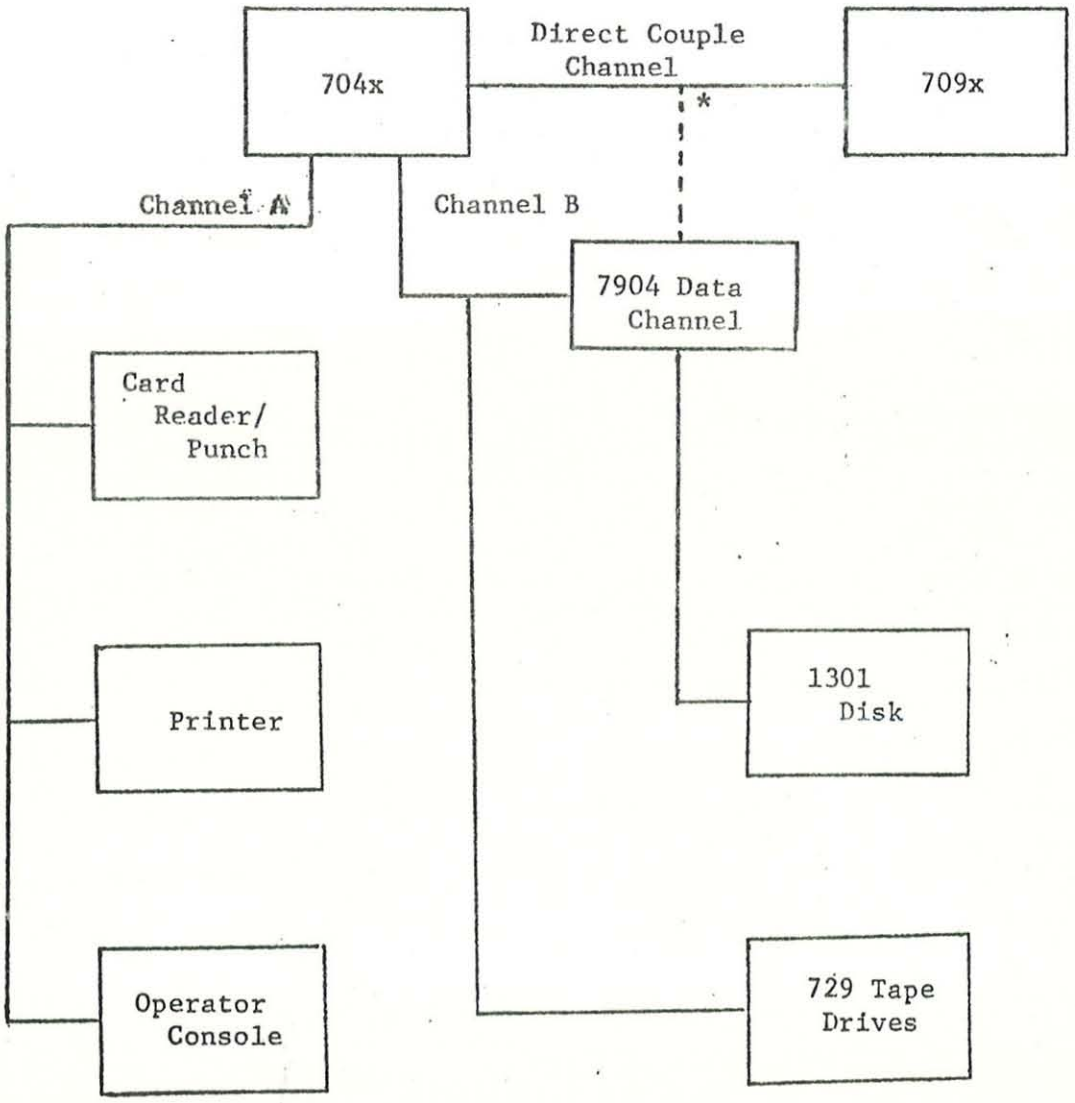
2. Equipment Configuration

Figure 1 shows a typical DCS system. The important features are:

(1) an IBM 709x and an IBM 704x, each with 32k of core storage, to serve an primary and support processors respectively.

(2) interconnection of the two processors by a Direct Couple channel and associated control lines for interruption of each computer by the other.

[2] IBM 7090, 7094, and 7094-II computers will be denoted as a class by "709x"; likewise the IBM 7040 and 7044 will be denoted by "704x".



*709x programming systems are transmitted from the 1301 to the 709x through the Direct Read to Coupled Processor feature of the 7904 Data Channel.

Figure 1. Direct Couple Operating System--General Machine Configuration

(3) at least one module of 1301 disk storage for system residence, input job queue, output queue for printer/card punch, and intermediate blocking of data to/from tapes.[3]

(4) all unit record equipment was attached to a 704x channel.

(5) 8 to 20 tape drives were attached to the 704x's channels.

(6) <u>no</u> i/o devices were attached directly to the 709x. All i/o operations were performed by the 704x and $\overset{\text{data}}{\wedge}$ transmitted main storage-to-main storage via the Direct Couple channel. One exception was the loading of the supervisory program for the 709x (IBSYS) directly from disk to the 709x using the Direct Couple channel without intermediate buffering in the 704x.

It should be emphasized that DCS, like most 709x systems, had a tape-resident data base. Disk was rarely used for permanent storage of data other than programs (system and user).

<u>3. Operating System Organization</u>

"The 709x treats the directly coupled 704x as it would a data channel, and is dependent on the 704x for all i/o and all operator functions. The primary function of the 704x is to service the i/o requirement of the 7090".[4]

---

[3]All tracks within a 1301 cylinder were required to have the same record format. As a result, there was much less flexibility in storing disk data sets than there is with the IBM S/360 disks (2311 and 2314).

[4]IBM Corp., <u>IBM 7090-7040 Direct Couple Operating System-Preliminary Specifications</u>, Form C28-6372, page 8.

The DCS operating system is divided into a supervisory program for the 709x and a supervisory program for the 704x. The 709x runs under a version of IBSYS[5] (DC-IBSYS) modified so that all i/o operations are handled by the 704x. Two modes of operation are available:

1. Direct-Mode - All components of IBJOB operate in this mode and service i/o requests by setting-up a description of the i/o function desired in 709x storage and then interrupting the 704x to request performance of this function. 709x processing continues while the 704x is interpreting the description and initiating actual data transfers along the DC line.
2. Compatibility Mode - Programs that do not set-up i/o request descriptions (mainly non-IBJOB systems) attempt to execute i/o instructions on non-existent 709x channels. An interruption of the 704x results; the 704x system interprets the i/o commands and control words and performs an equivalent i/o operation on the 704x. (This might mean only a transfer of data from the 704x memory via the DC line to the 709x memory.) After executing the i/o instruction, the 709x does no further processing until the 704x issues a restart command for the 709x. Thus, in compatibility mode the 704x acts as a very intelligent channel.

[5]See Appendix A for a description of IBSYS.

The 704x is controlled by a multiprogrammed monitor, DCMUP. All preprocessing and postprocessing functions for jobs and all servicing of 709x i/o requests are performed by DCMUP. Also the execution of 709x jobs is monitored to check run time and amount of output (print lines and cards to be punched) against user specified limits.

4. Job Flow

The processing of a job in DCS can be divided into three logical phases, each consisting of one or more stages.

Preprocessing - (704x)

Input stage - The job is read from the card reader, control cards are analyzed,[6] and the job is entered into the system by

(1) Blocking the input deck into DCS record format and placing it on the disk.

(2) Placing a complete job description on the disk.

(3) Making entries for the job in various in-core tables.

Setup Stage - If the job has input tapes, these must be allocated drives and mounted prior to scheduling the job for execution. A special control card is used to indicate the setup requirements including the option to have the tape blocked into DCS record format and placed on disk. Operators are provided with mount messages.

Remark: It would have been possible to include additional functions in the preprocessing step. For instance, Smith [reference 4, pp. 226-227] considers performing a pre-edit of source

---

[6]DCS control cards consisted of (1) standard IBSYS control cards, and (2) DCS cards containing setup and 704x utilities control information. The format and options available on both types of control cards were relatively simple compared with OS/360 Job Control Language (JCL). As a result, the rate of rejection of jobs for erroneous control cards was lower.

language programs on the 704x prior to compilation on the 709x. This has the advantage of preventing jobs containing simple errors from reaching the 709x. However, unless the 709x compilers were redesigned to make use of text pre-edited on the 704x, there would be a wasteful duplication. Such a redesign of the IBSYS language processors was beyond the scope of the DCS project.

Processing - (709x)

After completion of the preprocessing phase, including a setup stage if required, the job is selected for execution on the 709x. The scheduling of a job is dependent on time of entry into DCS, setup requirements, priority code, and estimated run time and printing volume. After a job is selected by DCMUP on the 704x for execution, the 709x is loaded with a fresh copy of IBSYS and the 709x is started by the 704x. Execution on the 709x is essentially the same as on a normal 709x under IBSYS except all i/o requests are routed to the 704x. DCMUP satisfies these requests using its i/o buffers. As input buffers are transmitted to the 709x, new buffers are filled from the disk. Similarly output data is continually blocked into buffers and written on the disk for eventual printing/punching or transfer to tapes. As mentioned earlier, DCMUP keeps track of the amount of output and checks this against estimated maxima. Likewise, DCMUP stops 709x processing when the runtime estimate is exceeded. If the 709x executes a halt instruction, an automatic interrupt of the 704x occurs so a new job can be started.

Postprocessing (704x)

The postprocessing of a DCS job consists of breaking-down mounted tapes, scheduling the printing and punching of output placed on the disk during execution, and purging of all information about the jobs from DCS after all postprocessing activity is complete.

5. Evaluation and Remarks

DCS significantly reduced turnaround time for 709x jobs as compared with a 709x-1401 configuration. This was particularly true for short jobs (less than 10 minutes) as they were normally given higher priority treatment under DCS. The most important factor in reducing turnaround time was the elimination of batching input jobs and output.

Thruput improvements for the 709x were attributable to reduction of idle time during tape-setup. The effect of blocking tape records onto the disk improved i/o performance but was probably of secondary importance.

It is interesting to conjecture on the effect multiprogramming on the 709x could have had on the development of DCS. First, the 32k words of main storage for an IBSYS-based system is insufficient for meaningful multiprogramming, since most sytems programs were designed for a 32k environment. If more storage had been available, multiprogramming would have allowed overlap (or "padding") of tape setup time. It seems though that a DCS approach would have still had advantages due to the parallel handling of all i/o activity and scheduling of the 709x. We turn now to more recent adaptations of DCS for IBM S/360 that further emphasize these points.

B. Attached Support Processor (ASP)

1. Background

ASP is an outgrowth of the 704x-709x Direct Couple System using IBM S/360 computers. The system has been available for approximately 18 months. More than 30 large-scale S/360 models have been configured into ASP organization. A sizable number of these operate in 709x emulator mode during a portion of the day. (One of the features of ASP is the intermixing of OS/360 and emulator jobs without interruption of processing.) The primary processor may be either an S/360 Model 65 or 75 with or without the 709x emulator feature. The support processor may be either a Model 40 or 50. Version 2 of ASP allows configurations with dual primary processors controlled by a single support processor. ASP can be run on the support processor under any OS/360 option (PCP, MFT, or MVT); likewise the primary processor(s) may run under any OS/360 option.

2. Equipment Configuration

An ASP configuration typically contains:

(1) S/360 Model 65 (often equipped with the 709x Emulator) or Model 75 as primary processor; a model 40 or 50 as the support processor. The larger CPU typically has 500k bytes of main storage, the smaller CPU 250k bytes of main storage. The CPU's rarely share core storage.

(2) A channel-to-channel adapter connects the two CPU's. This is the only interconnection between the primary and support processors, a significant difference from DCS where the support processor was able to directly control the primary processor. All control of the primary processor in ASP is through pseudo operator console assigned to the support processor via the channel-to-channel adapter.

(3) The primary processor has tapes and direct access storage devices (DASD's) directly attached to it. This is a significant change from DCS, where the support processor performed all i/o operations for the system. The attachment of i/o devices to the primary processor requires a larger operating system to service i/o requests and interrupts associated with these devices. Also main CPU time must be used to service this i/o activity; under DCS parallel processing of i/o interrupts and requests was performed on the support processor.

(4) The support processor has disks for storage of SYSIN data and output data to be printed/punched. (Also local card reader/punches and printers are attached.)

(5) Remote terminals are attached to the support processor and full support is provided for remote job entry.

Figure 2 shows a representative configuration.

3. Operating System Organization

The ASP control program operates as a task under control of OS/360. ASP uses many OS/360 supervisory services but independently performs data management functions.

ASP consists of three major components:

. ASP resident programs

. ASP buffer pool

. ASP Dynamic Support Programs (DSP's)

The resident programs and buffer pool provide a multiprogramming monitor and i/o buffers for the DSP's which perform various ASP functions including

(1) Reading the ASP input stream, scanning for ASP and OS/360 control cards, and performing the appropriate ASP functions. (No attempt is made to discover and correct control card errors that are all too common for OS/360 JCL.)

(2) Handling setup and breakdown activity on the primary processor for tapes and disk packs.

(3) Printing/punching local output and transmitting output for remotely-submitted jobs.

(4) Initiating and monitoring primary processor execution. This includes checking job execution time against estimates and likewise printer/punch output against estimates.

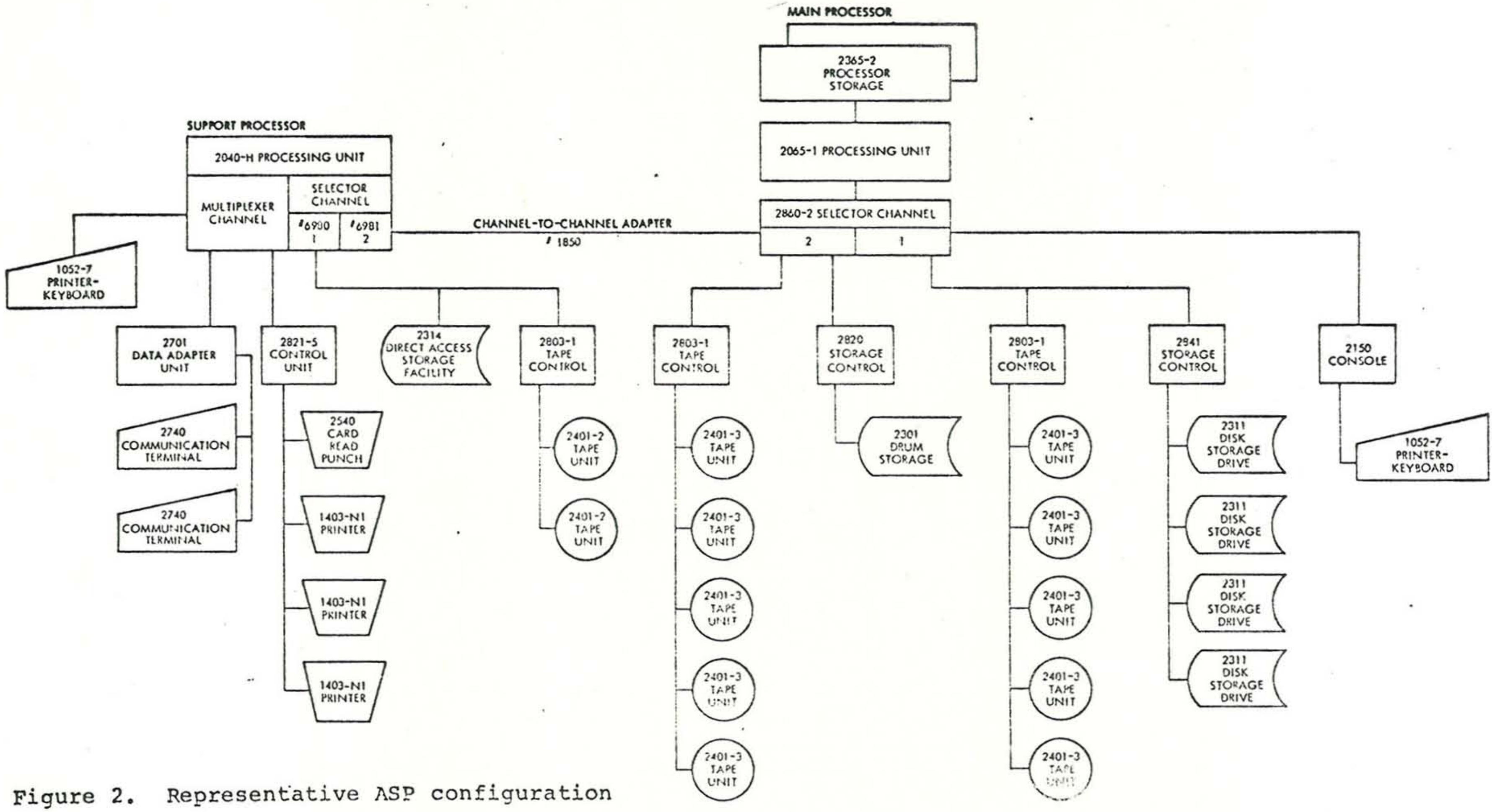


Figure 2. Representative ASP configuration

The operating system for the primary processor is generated with the primary operator console and a maximum of 15 other physical devices assigned to the channel-to-channel adapter. The datasets associated with these devices are simulated as nine-track tapes. The primary processor i/o supervisor translates Execute Channel Program (EXCP) requests for these datasets into channel-to-channel adapter i/o operations. The appropriate data is then transmitted via the channel-to-channel adapter. If the user attempts to read past the end of a dataset, a simulated end-of-file condition is returned to the primary processor.

4. Job Flow

Input queuing and setup under ASP are similar to DCS. During the execution stage ASP serves as a programmed operator for the primary processor. This differs from DCS where the 704x controlled the 709x directly with instructions (e.g., SRC - Start Remote Computer). Since the primary processor in ASP is controlled by some version of OS/360, the use of a pseudo operator console allows the support processor to completely control primary processor operation in a natural (i.e., easily implemented) way.

During execution all primary processor requirements for SYSIN, SYSPRINT, and SYSPUNCH are handled by the support processor using the channel-to-channel adapter.

The breakdown, print, and punch stages for ASP are similar to those of DCS. An additional postprocessing stage has been added to handle teleprocessing requirements. Jobs received from remote terminals are routed to an Output Transmission Stage. (For such jobs the print and punch stages are skipped.) This stage transmits job output to the appropriate terminal for printing and punching. Optionally output may be directed to a terminal different from the job source. A final purge stage operates similar to that of DCS.

ASP also includes facilities for priority processing of jobs and the important feature of automatically advancing the scheduling of a job once it has been setup.

5. Evaluation and Remarks

The management of SYSIN and SYSOUT files is essentially the same for ASP as DCS; the management of tape setup is the same with the exception of direct attachment of i/o devices to the primary processor. The addition of TP support for remote job entry is similar to that for DCS. A minor difference between DCS and ASP is the mechanism for controlling the primary processor - direct control by instruction (DCS) versus simulated operator commands (ASP). The handling of emulation jobs within the normal ASP job stream certainly has operational advantages and indeed ASP is an obvious choice for installations with a significant percentage of emulation jobs.

Remark: Emulation jobs are really a special type of setup job in which the primary processor itself must be setup. Therefore, outside a dual primary processor configuration, it is difficult to schedule their execution without discontinuing all other usage of the CPU (e.g., other tasks in an MFT system).

Heavy use of tapes on the primary processor in ASP will result in significant i/o wait time. A partial solution is to attach tapes only to the support processor, perform full-track blocking onto the disks prior to execution, and provide lookahead buffering on the support processor (as was done in DCS). Attachment of DASD's to the support processor will not eliminate i/o waiting for any dataset organization using random accessing since lookahead buffering is not possible. DASD's may as well be attached directly to the primary processor.

If the primary processor still waits more than 10% for completion of DASD i/o requests, then two possible alternatives seem attractive.

1. Run more than one task at a time on the primary processor overlapping execution of one task with waiting of the other (i.e., OS/360-MVT).
2. Perform some or all of the support processor functions on the primary processor.

The first approach has the same advantages as ordinary ASP for a single-tasking system plus the added advantage of reduction of primary processor idle time. An additional advantage accrues from the preprocessing activity on the support processor - it is possible to completely eliminate jobs with control-card errors, references to non-existent datasets, etc. without their ever wasting primary processor scheduling and initialization time. There are other attractions to this approach but these will suffice for now.

The second alternative, placing support processor functions on the primary processor as additonal tasks under a multi-tasking control program, has been adopted by the HASP and CLASP systems that we shall next discuss.

C. Closely-Linked ASP[7]

CLASP was developed by the Canadian Central Data Processing Service Bureau. It is in essence ASP running in a partition of MFT with a problem program in the other partition. Communication between the two partitions is handled by i/o requests directed to the channel-to-channel adapter. (Of course, there is no channel-to-channel adapter, only a pseudo-device[8] treatment of channel programs directed to it.)

As obvious advantage of CLASP is that primary processor idle time is greatly reduced, but the amount of CPU time available for problem-program execution is also reduced. Assuming that CLASP were able to maintain a thruput equal to an equivalent ASP configuration, then CLASP would be cheaper (by one CPU); however, CLASP has large core storage requirements (150-250k).

---

[7]CLASP is identical in organization to LASP, another ASP variant.

[8]See reference 1 for a discussion of pseudo-devices.

D. Houston Automatic Spooling Priority System (HASP)

HASP attacks some of the OS/360 problems addressed by ASP and CLASP though the techniques chosen are different. HASP operates as a partition of an MFT or MVT system and reduces OS/360 SYSIN / SYSOUT limitations by

(1) Reading the input job stream continuously, blocking it into full-track records on disk.

(2) Maintaining buffers of SYSIN data for satisfying problem-partition demands. (The problem program or language processor reads SYSIN from a pseudo-device that results in activation of the spool partition to deblock a card image.)

(3) Managing SYSPRINT output from the problem partition in a manner similar to (1) and (2) with the queued SYSPRINT tracks being read and printed continuously.

(4) Minimizing access mechanism motion for SYSIN/SYSOUT files by always writing buffers into the "nearest" empty disk track (i.e., if possible choose a track from within the disk cylinder currently under the access mechanism; otherwise write into the cylinder that requires the least motion).

HASP checks the volume of SYSPRINT and SYSPUNCH output against user-specified maxima. Runtime is also compared against the user estimate.

HASP-Version I did not include the advance setup feature of ASP for reels and packs though this has been provided in Version II.

III. Summary

This paper has discussed four approaches to the problems of high-thruput computer configurations. The evolution of a system emphasizing the performance of all i/o and support activity on a support processor (DCS) into an equivalent system that places all support processor activity in a partition of a multitasking single processor system (CLASP) has been detailed.

The ultimate criteria for any computer sytem design-including computer architecture, configuration, and operating system elements- is its cost effectiveness in comparison with other designs. By this criteria DCS was a superior computer system for a high-thruput IBM 709x installation. Whether the ASP or CLASP or HASP approach is as satisfactory for the S/360 large-scale computers, can not yet be clearly determined. However, users of large S/360 computers have overwhelmingly expressed their preference for HASP with roughly 75% of the installations using it. The remaining 25% is divided among ASP, CLASP, OS/360-MFT/MVT, and other special systems (e.g., TUCC).

It is this author's opinion that better designs using S/360 components are yet to come. The idea of a separate support processor "funneling" and "filtering" work into a primary processor may be justifiable on the basis of reliability and availability. (This assumes that if either processor fails the other can perform most system functions.) The exact line of demarcation among support and supervisor services, and problem program activities is unclear and must be judged by each implementor in the light of design objectives.

References and Bibliography

Articles

1. Freeman, David N., "Pseudo-Devices in OS/360", Proceedings of the 1968 Southeastern ACM Regional Conference, June, 1968.

2. Baldwin, F. R., et al., "A Multiprocessing Approach to a Large Computer System", IBM Systems Journal, Vol. 1, September, 1962, pp. 64-76.

3. Hollander, G. L. et al., "The Best Approach to Large Computing Capability - A Debate", AFIPS Conference Proceedings-Spring Joint Computer Conference - 1967, Vol. 30, pp. 463-486.

Computer Systems
Direct Couple System (DCS)

4. Smith, E. C. Jr., "A Directly Coupled Multiprocessing System", IBM Systems Journal, Vol. 2, September-December, 1963, pp. 218-229.

5. IBM Corp., IBM 7090-7040 Direct Couple Operating System-Preliminary Specifications, Form C28-6372.

6. IBM Corp., IBM 7090-7040 Direct Couple Operating System-Operator's Guide, Form C28-6384-2.

7. IBM Corp., IBM 7090-7040 Direct Couple Operating System-Programmer's Guide, Form C28-6382-3.

Attached Support Processor (ASP)

8. IBM Corp., Attached Support Processor System-System Programmer's Manual, Form H20-0323-2.

9. IBM Corp., IBM S/360 ASP System-Version 2-System Description, Form H20-0466-0.

Closely Linked ASP (CLASP)

10. McDorman, E. A., and Russell, J. M., "A Performance Improvement Plan for CDPSB", Central Data Processing Service Bureau (CDPSB), Government of Canada, Ottawa, April, 1967.

11. ---------, Development of the Central Data Processing Service Bureau, CDPSB, Ottawa, July, 1967.

Houston Automatic Spooling Priority System (HASP)

(See reference 1.)

Appendix A
IBSYS - IBM 7090-94 Operating System

IBSYS is the operating system used on the large, second-generation IBM scientific computers. Figure 3 shows the organization of the major elements of IBSYS. IBJOB is the principal subsystem containing language processors such as FORTRAN and COBOL compilers. The illustration also indicates the control cards used to invoke the IBSYS subsystems (e.g., "$EXECUTE IBJOB").

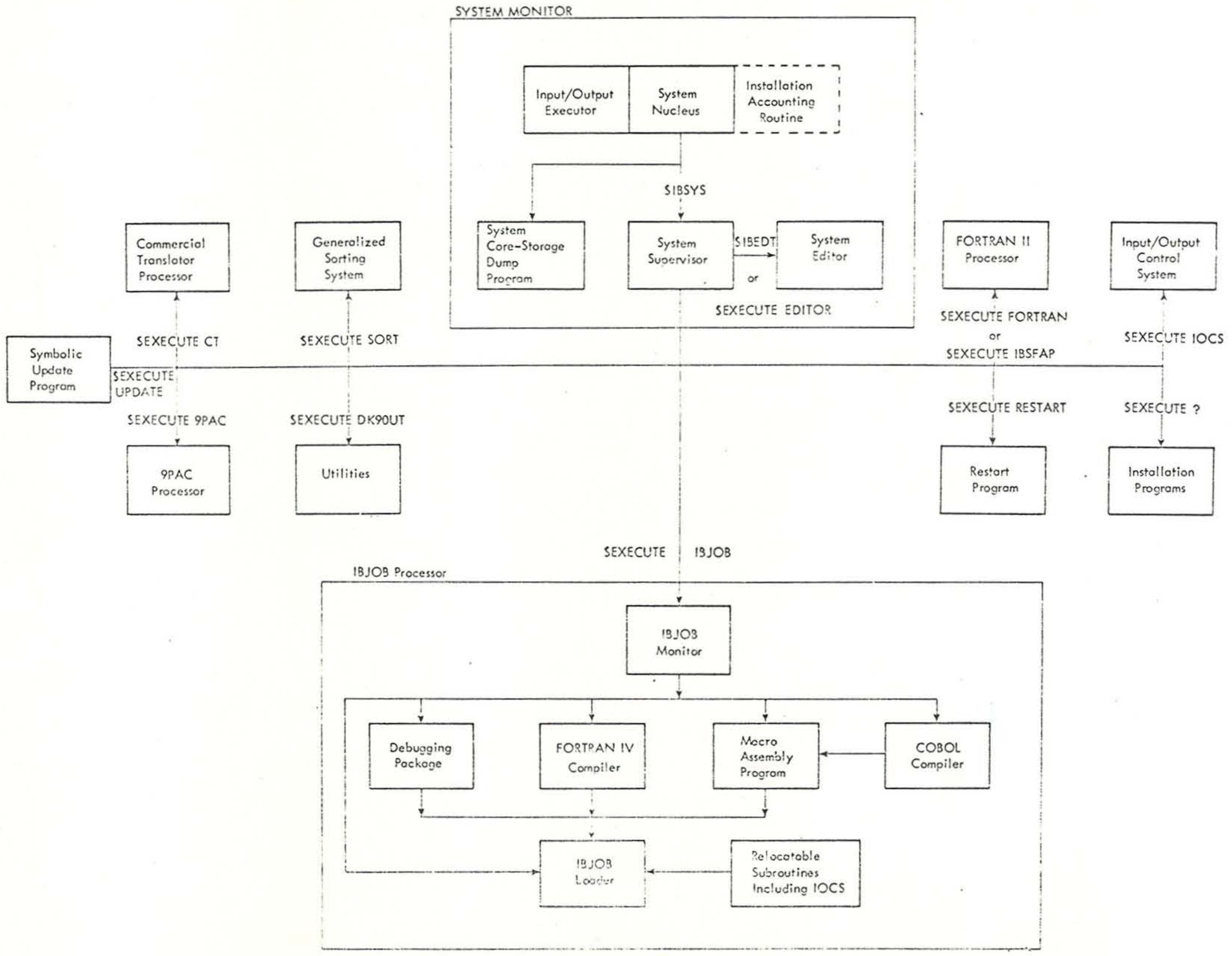


Figure 3. IBM 7090/7094 IBSYS Operating System