System Applications Litton L-304 Microelectronic Computer

MS 2149

LITTON L-304 SYSTEM APPLICATIONS

17 July 1967

Prepared by:

Data Systems Division
Litton Systems, Inc.
8000 Woodley Avenue
Van Nuys, California

LITTON DATA SYSTEMS
REGISTER ENTRY
REGISTER SELECT
PROGRAM LEVEL
PHASE REGISTER
TIME SLOT
ITERATION COUNTER
RUNNING
STOPPED
LITTON DATA SYSTEMS
MEMORY DRAWER
A/C DRAWER No. 3
A/C DRAWER No. 2
DATA SYSTEMS DIVISION OF LITTON INDUSTRIES
POWER SUPPLY DRAWER
PRENTICE-HALL SERIES
Maley and Earle
The Logic Design of Transistor Digital Computers

PREFACE

The L-304 computer is the result of a number of years of extensive research and engineering effort to produce a high-speed, modularized digital computer expressly meeting the rigorous functional and environmental requirements imposed on tactical military data systems. As such, the L-304 incorporates in its design significant organizational features in a combination heretofore not available in general-purpose computers. These features and the reasons for their inclusion in the L-304 design are covered in this document.

A brief L-304 description is contained in Appendix F for the unacquainted reader.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

LIST OF ILLUSTRATIONS

LIST OF TABLES

SECTION 1

INTRODUCTION

The Litton L-304 general-purpose computer was developed through the cumulative efforts of DSD's engineering and technical staff to solve the typical problems encountered in developing large (or small) real-time automated command and control systems. The resultant features provided by the L-304 greatly exceed those of the so-called "conventional" computer, especially in the areas which enhance convenience of program and system development. The purpose of this document is to acquaint the reader with the many beneficial design features of the L-304 Computer, to relate them to the characteristics of more conventional computers, and finally to demonstrate the L-304's strength in a large system application.

The text is presented in three sections: Basic Concepts, Computer Features discussion, and a typical Airborne Tactical Data System environment discussion. Section 2 (Basic Concepts) discusses the evolution of desirable computer features and how they enhance successful system development. Understanding and appreciating the power of multiprogramming and multiprocessing are discussion objectives in this section. Section 3 (Computer Features discussion) explores the design features of the L-304, relating these features to those of conventional computers, where appropriate, to provide a common base for discussion. Section 4 (A Typical ATDS Environment discussion) relates the L-304 directly to a typical large tactical problem implementation (ATDS) and provides practical examples of the L-304's inherent advantages over other, more conventional computers.

In order to appreciate the content of this document, it is first necessary to have a basic understanding of the terminology used. A few of the more important terms are:

- Multiprocessing
- Multiprogramming
- Program Levels
- Interrupts
- Event Dependent
- Time Dependent

Briefly, "multiprocessing" is a term used to describe a computing system consisting of two or more processors that share a common memory. This allows direct, simultaneous access and computation by several processors on a common set of data.

"Multiprogramming" is a term used to describe a nonserial operation of discrete events, i. e., several functions that can, at any time, be in various stages of completion. This allows an optimum sharing of real time (on a priority basis) and is considerably more efficient than "serial programming" where each function must be completed before the next can begin.

"Program levels" is a term uniquely associated with the L-304. It will suffice to say, at this point, that a program level represents the beginning of a discrete set of calculations such as a subroutine or series of subroutines. One can arrange the levels in any priority sequence desired (much as a queueing table) and, with the program-controlled "enable and disable" capability, can accomplish the essence of multiprogramming techniques.

"Interrupt" is true to its dictionary meaning, i. e., disturbing the normal course of events. The normal course of events within a computer is a progressive execution of instructions. An interrupt can occur at any instant (normally due to input/output transmission requests or timing alarm). This, assuming no higher priorities are active, causes control to pass to a preselected series of instructions (input/output channel dependent) which dictate performance of a preestablished action.

"Event dependent" is a term used to describe a sequence-dependent event problem, i. e., Event A must be performed before Event B, Event B before C, etc.

"Time dependent" is used to describe a problem that requires performance of specific tests at predesignated intervals. Most real-time problems are a mix of both event- and time-dependent functions.

The remainder of this discussion demonstrates that the specific programming and hardware techniques chosen to be implemented by Litton succeed in principal and in practice in meeting the goals of programming simplicity as well as the modularity prerequisite to system expansion or program modification. This can best be done by first showing in detail how the hardware of the L-304 merely implements in logic several of the more repetitive and time-consuming tasks inherent in the programming of real-time systems and then illustrating how the software makes use of this logic to assure minimum effort for growth and program modifications.

SECTION 2

BASIC CONCEPTS

MULTIPROGRAMMING

Many of the design features of the L-304 were incorporated to facilitate the use of multiprogramming techniques. Therefore, in the development of this discussion, Litton reviewed the literature to determine the history and extent of the use of multiprogramming.

Historical Evolution

The first reference to the actual use of multiprogramming is found in the 1961 Proceedings of the Eastern Joint Computer Conference. Of the two papers on this technique, the one of greater significance provides a description of the RCA 4100 computer developed for the BMEWS (474L) system. This paper, entitled "The Logic Design of the FC-4100 Data Processing System," describes the implementation of multiprogramming in this early RCA computer. Moreover, multiprogramming systems have since been described at virtually every major computer conference held since that date. Multiprogramming systems are currently incorporated in the IBM system/360 series, the CDC 6600, the GE 634, and the SDS Sigma 7, as well as, in Europe, the Bull Gamma 60 and the Ferranti Atlas.

Development of Multiprogram Logic

Until the mid-1950's, computer programs were serial in nature in the most complete sense. Each action had to await the completion of the preceding action. Even input/output operations required one or more program steps for each word, character, or even bit transferred to or from a peripheral device. Programs were generally required to enter tight timing loops in order to synchronize the data transfer to the rate acceptable by the ancillary device.

By 1957, the first elements of logic that interleaved operation with program interrupt had already become commonplace in the new generation of transistorized computers. The specific features implemented at that time were:

a. A buffered input/output channel operating in an interleaved fashion with program execution.

b. A real-time clock capable of being set and interrogated by the computer.

While this system had the advantage of permitting program and input/output functions to operate in an interleaved manner, each input/output operation was still required to be executed serially and much program time was still lost in periodic testing for completed data transfers and in periodic interrogations of the real-time clock. The next step was to cause an interruption to the running program upon the completion of the input/output operation. This was accomplished by transferring program control to a predetermined memory location in which the starting address of an interrupt source analysis program was generally stored. This analytical program determined why the interruption took place and whether or not to proceed with the previous program or enter a new program.

As long as input/output operations proceeded serially, the task of controlling the system was fairly simple. Then, computers with multiple or time-shared input/output channels were introduced. These machines allowed many input/output units to operate simultaneously and allowed each to transfer data in an interleaved manner with the program. This increased the complexity of control of both the multiple data transfers and the multiple interrupts. At first, the multiple input/output transfers were controlled by independent unit controllers such as magnetic tape controllers, printer controllers, card read-punch controllers, etc. Each contained a set of control registers which kept track of its set of data transfers. Later, as the ratio of computer-to-peripheral-speed increased and as more simultaneous operations could be done in the central processor, the control of peripherals was moved back into the data processor with the majority of systems storing the input/output control words in memory.

The first solution to the interrupt control problem made maximum use of software and minimum use of hardware (probably because the repetitive operations had not as yet been clearly identified and defined). It consisted of the interrupt source analysis program which was still a single program that was entered automatically as a result of the completion of any input/output transfer and which scanned all the possible input/output channels to determine which was causing the interrupt. To ensure that the interrupt source analysis program would run to completion, it generally, as its first step, inhibited all other interrupts.

Of course, during the execution of this time (and memory) consuming interrupt source analysis program, other devices sometimes completed their transfers and, as soon as the interrupt inhibit was removed, caused the computer to once again enter the interrupt source analysis program. This interrupt causes the additional problem (only lightly alluded to until now) that there is now a queue of partially completed programs, each waiting to be executed.

Determination and control of the order in which these programs were to be executed then became the function of another program — the priority determination routine. This program generally worked from a set of tables which listed, in predetermined order, the relative priorities of the various input/output control and processing routines. The difficulty of the programmed maintenance of a rapidly changing queue without some hardware aid became evident quite early in the development of this concept. An early attempt at a partial solution to this problem appeared in the NCR-304 which provided for separate transfer points for each type of input/output interruption which could occur such as completed transfer, parity error, broken tape, no response from unit, etc. A fixed priority for each type of interruption was established so that the most critical events could always be processed first.

A later attempt at solution was in the Burroughs B-5000 which incorporated logic push-down pop-up lists which, among other things, reduced the complexity of maintaining priority queues. Since that time, many other computers have also incorporated hardware aids for interrupt control. A partial list would include the IBM 360-92, CDC 6600, SDS Sigma 7 and the GE 635.

Another aspect of interrupt control which has not as yet been discussed is that of saving the status of the program being interrupted. Specifically, this consists of saving the contents of index registers, accumulator(s), any control flags, any base address or memory extension registers, and the register which has the address of the current or next instruction to be executed. Although time-consuming, this process is normally not complex, provided that the proper interrupt lockout instructions and register access instructions are included in the computer logic.

In the foregoing discussion, it has been implied that the interrupt systems have developed through the years to operate on a priority basis rather than on a time slot or sequence basis. The evolution of real-time and even batch processing programming systems has shown this to be the case. An analysis of the requirements for successful time-slot processing should quickly show why such a scheme is particularly inapplicable to a large scale military data system.

In any system, as long as events proceed in an orderly and predictable manner and the time duration can be predicted within a reasonable tolerance, the time-slot system will work well. However, if variation in event sequence or timing is a possibility or if actions can crowd in on one another (such as all console operators requesting new coordinate computations simultaneously with detection of a test error in the radar preprocessor interface unit and receipt of an "initiate communication" pulse, the time-slot approach quickly falls out of synchronism. Thereafter, some kind of priority sequencing must be established. A complex resynchronizing routine must be executed or events must be delayed, possibly to the point where the system is no longer operating in acceptable real time.

A priority system, on the other hand, when properly implemented frees the programmer from having to maintain a series of complex time relationships (required of the time-slot system) while at the same time assuring the system that the highest priority programs will always be executed before less important or less timely programs.

L-304 Implementation of Multiprogram Logic

The L-304 computer has implemented in logic those repetitive and well-defined functions of a priority interrupt system which had once been performed by program but which are more and more being assigned to hardware. Transferring these functions from programming logic to hardware logic removes from the programmer the burden of designing, coding, and debugging the bookkeeping portion of interrupt analysis and priority control routines. The basic principle behind the execution of these routines is not changed by the method of implementation. For example, the status and mask (program activity) registers of the L-304 are merely a hardware implementation of the priority determination routine and the queue. The key words of the input/output control have taken the place of the registers formerly found in the independent input/output controllers. The termination word has taken the place of the interrupt source analysis routine by directly causing an entry into the specific program level required rather than having to scan the possible interrupting units to determine which should be serviced.

Figure 2-1 illustrates the fundamental sameness of these operations performed on an L-304 and on a computer in which these bookkeeping functions must be programmed. It can be seen that the steps are logically identical but that, in a computer where these operations have not been incorporated in the hardware, the number of instructions and the execution time required become quite extensive. In the L-304 and in several of the other scientific and commercial computers previously mentioned, portions or all of these functions have been built into the hardware. Therefore, the processing time spent in interrupt source analysis, priority determination, and input/output control is reduced to a minimum. There is also a considerable reduction in program complexity and, consequently, in memory used.

There is no feature in the L-304 that has not been previously used in some other computer. Thus, no new, untried technology is proposed. However, no other computer has the particular combination of desirable features required for efficient

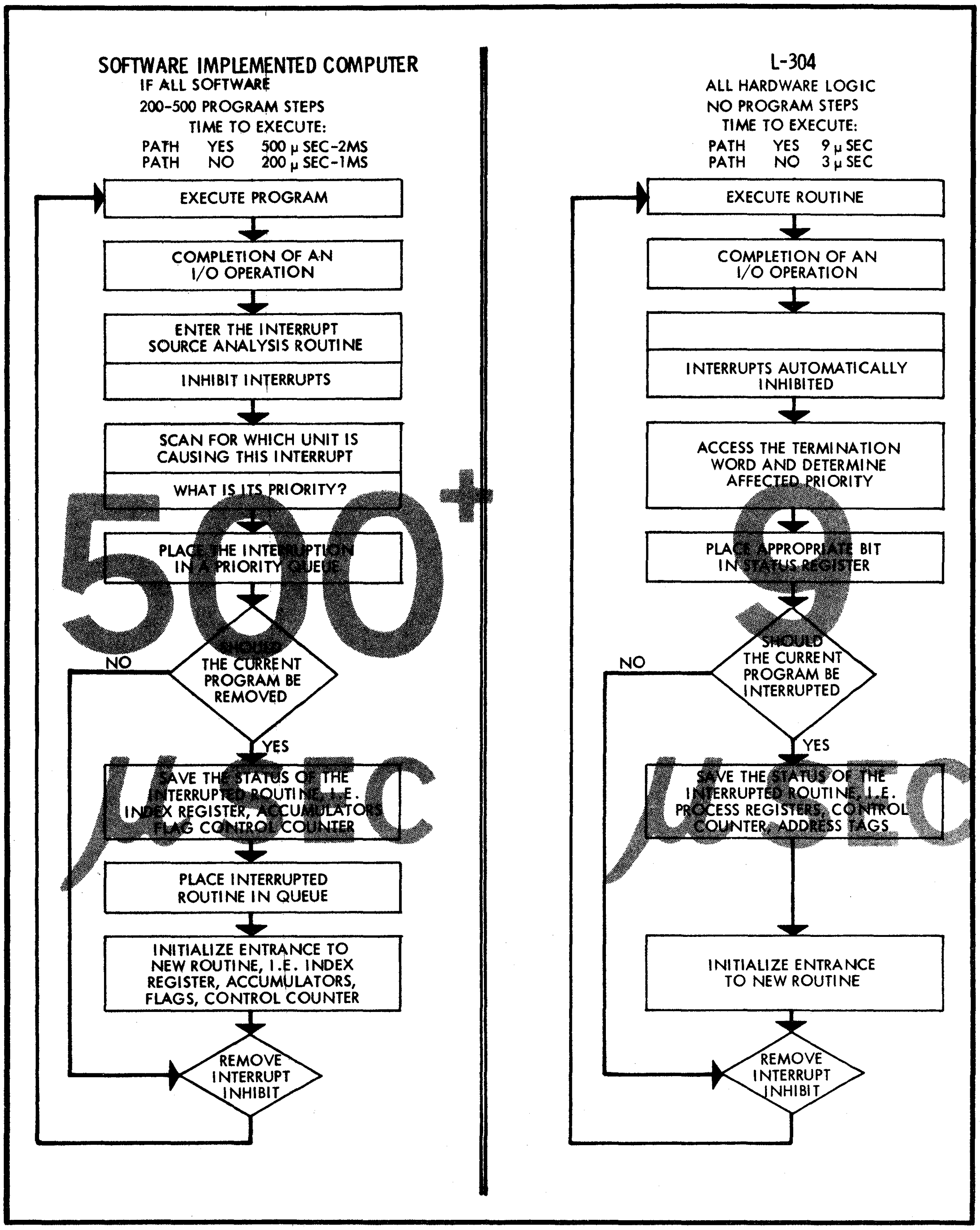

SOFTWARE IMPLEMENTED COMPUTER
IF ALL SOFTWARE
200-500 PROGRAM STEPS
TIME TO EXECUTE:
PATH YES 500 μ SEC-2MS
PATH NO 200 μ SEC-1MS
EXECUTE PROGRAM
COMPLETION OF AN I/O OPERATION
ENTER THE INTERRUPT SOURCE ANALYSIS ROUTINE
INHIBIT INTERRUPTS
SCAN FOR WHICH UNIT IS CAUSING THIS INTERRUPT
WHAT IS ITS PRIORITY?
PLACE THE INTERRUPTION IN A PRIORITY QUEUE
SHOULD THE CURRENT PROGRAM BE REMOVED
NO
YES
SAVE THE STATUS OF THE INTERRUPTED ROUTINE, I.E. INDEX REGISTER, ACCUMULATORS FLAG CONTROL COUNTER
PLACE INTERRUPTED ROUTINE IN QUEUE
INITIALIZE ENTRANCE TO NEW ROUTINE, I.E. INDEX REGISTER, ACCUMULATORS, FLAGS, CONTROL COUNTER
REMOVE INTERRUPT INHIBIT
500+ μSEC
L-304
ALL HARDWARE LOGIC
NO PROGRAM STEPS
TIME TO EXECUTE:
PATH YES 9 μ SEC
PATH NO 3 μ SEC
EXECUTE ROUTINE
COMPLETION OF AN I/O OPERATION
INTERRUPTS AUTOMATICALLY INHIBITED
ACCESS THE TERMINATION WORD AND DETERMINE AFFECTED PRIORITY
PLACE APPROPRIATE BIT IN STATUS REGISTER
SHOULD THE CURRENT PROGRAM BE INTERRUPTED
NO
YES
SAVE THE STATUS OF THE INTERRUPTED ROUTINE, I.E. PROCESS REGISTERS, CONTROL COUNTER, ADDRESS TAGS
INITIALIZE ENTRANCE TO NEW ROUTINE
REMOVE INTERRUPT INHIBIT
9 μSEC

Figure 2-1. Hardware versus Software Implementation of Control Functions

real-time data processing. It is this synthesis (based on Litton's experience with other tactical data processing systems) that is unique and that will serve to permit the rapid, economical programming of the data processing function.

MULTIPROCESSING

The use of multiprocessing techniques represents an equally dramatic progression of system development over the past several years. However, in the case of multiprocessing, the result of this development is more easily discernible. When considering the rapid task completion and mission survivability inherent to a multiprocessor configuration, its effectiveness is an obvious conclusion. As stated in ESD-TDR-64-168 dated January 1965, by D. R. Isreal of the Mitre Corporation:

> "... the modular concept has extended to the central processor itself, and the truly modern machine design includes the capability of employing several processors operating in parallel and sharing the available memory and in-out modules. This permits what has been termed 'multiprocessing' with several processors operating together on a single job (it is to be distinguished from 'multiprogramming' in which one machine works on several different tasks)."

Historical Evolution

A multiprocessor configuration does not impose any significant design alterations on the processors comprising the configuration. For this reason, it is unnecessary to precisely trace the development of this concept; it will suffice to state a few general comments applicable to the document.

Multiprocessing, as indicated by D. R. Isreal, permits several machines to operate on a single job. This implies separate computers having the capability of information exchange. However, to realize the extremely high effective operating speeds typical of multiprocessor systems, the information exchange medium must be carefully considered. The multiprocessor arrangement can achieve maximum effectiveness only as a result of software design which assures the fact of "several processors operating together on a single job." Limiting the software to anything less than a direct memory exchange medium defeats the intent of the multiprocessor system.

Perhaps the most widely known attempt at multiprocessing in a military system is the Naval Tactical Data System. The NTDS employs a parallel operation of two or three computers, depending on the anticipated environment and ship size. (A one-computer level is employed occasionally on smaller ships.) The system subtasks are assigned to the different computers: one computer has tracking; another, intercept control; and the third, display functions, for example.

However, the NTDS does not employ a common memory between the processors; the computers exchange information via input/output channels. This requires a time-consuming data encode and decode program in each computer. Another serious drawback is the machine time wasted for this input/output information exchange which could be devoted to meaningful input/output communication and program operation.

All major computer manufacturers have since realized this limitation and have configured subsequent multiprocessor systems in a noninterference, common memory arrangement. Thus, the higher effective operating speed is attained as well as the much enhanced system maintainability and survivability.

L-304 Implementation of Multiprocessing

As indicated in the previous discussion, a multiprocessor configuration does not generally impose modifications upon the basic computer design. The success of the arrangement is keyed to the software and the surrounding equipment. In recognition of these key inputs, a suggested Litton multiprocessor arrangement has been "designed" accordingly (and is discussed in Section 4, Typical ATDS Environment). This configuration yields system performance in terms of capability, speed and flexibility that is sufficient to not only meet all present requirements of the system, but also to provide a capacity for future system expansion.

SECTION 3

COMPUTER FEATURES

COMPARISON OF COMPUTER FEATURES

The design features of the L-304, how they compare with features of other computers, and why they are preferable for airborne tactical applications are discussed in this section. The L-304 is specifically compared with two other computers – the Navy AN/USQ-20B and the IBM System 4 PI, Model EP (the Model EP is the largest of a group of 4 PI models). The AN/USQ-20B was selected because of the general familiarity among Naval personnel with the AN/USQ-20A used in the NTDS,* as well as its widespread reputation as a good computer (Litton agrees with this contention). The System 4 PI, Model EP was selected because of prevalent familiarity among all computer users with the System 360 computer family and the fact that the System 4 PI, Model EP is functionally near equivalent to selected 360 models. The System 4 PI is presented, rather than the 360 itself, since it is a militarized processor.

The validity of the comparative analysis illustrated in Table III-1 is keyed to Litton's experience with the selected machines. Specifically, the USQ-20B was Government-furnished equipment that was incorporated in the Litton-developed Beach Relay Facility; and the Litton programming section has employed the System 360 extensively in developing the airborne tactical programs. (Approximately 28,000 instructions were programmed on the 360 to support airborne programs developed prior to L-304 availability.)

Features Comparable to Other Computers

In the Table III-1 comparison of the three computers (frequently referenced in this section), the first five items are essentially self-explanatory. These are L-304 features equivalent to the indicated USQ-20B and System 4 PI features. However, Items 2, 4, and 5 indicate some further capability and merit additional comments.

*The AN/USQ-20B and AN/USQ-20A are equivalent for all practical purposes; the primary difference speed.

AN/USQ-20B	SYSTEM 4 PI, MODEL EP	L-304
1. 30-BIT WORD LENGTH CAPABLE OF HALF-WORD ACCESS.	1. 32-BIT WORD LENGTH CAPABLE OF HALF-WORD OR MULTIPLE-WORD ACCESS.	1. 32-BIT WORD LENGTH CAPABLE OF HALF-WORD ACCESS.
2. 2 ACCUMULATORS AND 7 INDEX REGISTERS.	2. 16 REGISTERS EMPLOYABLE AS ACCUMULATORS OR INDEX REGISTERS.	2. 8 REGISTERS PER LEVEL EMPLOYABLE AS ACCUMULATORS OR INDEX REGISTERS.
3. 62-INSTRUCTION REPERTOIRE.	3. 75-INSTRUCTION REPERTOIRE (NOT INCLUDING FLOATING POINT).	3. 65-INSTRUCTION REPERTOIRE.
4. UP TO 16 INPUT/OUTPUT CHANNELS.	4. UP TO 262 INPUT/OUTPUT CHANNELS.	4. UP TO 64 INPUT/OUTPUT CHANNELS.
5. 7-DAY INTERNAL REAL-TIME CLOCK.	5. 15.5-HOUR INTERNAL REAL-TIME CLOCK (CAPABLE OF PROGRAM INTERRUPTION).	5. MANY REAL-TIME CLOCKS (EACH CAPABLE OF PROGRAM INTERRUPTION).
	6. DIRECT MEMORY ACCESSING TO OVER 4 MILLION WORDS.	6. PROGRAM-CONTROLLED MEMORY, EXPANDABLE TO 131K WORDS.
	7. COMMON MEMORY ACCESS BETWEEN PROCESSORS.	7. COMMON MEMORY ACCESS BETWEEN PROCESSORS.
	8. IMPROVED INSTRUCTION REPERTOIRE INCLUDING: A. MOVE B. EXECUTE C. BINARY-TO-DECIMAL OR DECIMAL-TO-BINARY CONVERSION	8. EXPANDED INSTRUCTION REPERTOIRE INCLUDING SUCH IMPROVEMENTS AS: A. MOVE B. EXECUTE C. TEST AND SET BIT D. GATED COMPARISON E. EXCHANGE F. INPUT/OUTPUT TRANSFER DIRECTLY FROM PROCESS REGISTERS G. THOROUGH LITERAL ADDRESSING CAPABILITY TO MEMORY
		9. AUTOMATIC PROGRAM QUEUEING AND TRIGGERING.
		10. TOTAL OF 64 PROGRAM LEVELS, EACH WITH OWN PROCESS REGISTERS AND MEMORY BANK SELECTION (SAVED AUTOMATICALLY WITH EACH LEVEL CHANGE).
		11. INPUT/OUTPUT TRANSMISSION VIA 8-BIT CHARACTERS OR 32-BIT WORDS.

Table III-1. Comparison of Computer Features

Dual-Purpose Registers

As shown in Items 2 and 4, the L-304 has a total of 512 dual-purpose registers, 8 per program level, with up to 64 levels. These are referenced as process registers because they are available for use either as accumulators or index registers. In the AN/USQ-20B, index registers are modified only by special instructions; consequently, convenience of index manipulation is somewhat restricted and the index value must often be transferred to an accumulator to accomplish the manipulation. The dual-purpose feature of the L-304's registers saves considerable instructions.

Fifteen of the sixteen general-purpose registers in the System 4 PI also have this dual capability. Register 0 is available only as an accumulator because a zero entry in the instruction implies no operand modification. In the L-304, the instruction addressing mode is specified in a distinct field; it is not inherent to the index register designation field.

Input/Output Channels

The L-304 programmer may directly access up to 64 input/output devices. Eight primary channels are available, each with eight bidirectional subchannels. Multiplexing units are necessary on each primary channel if more than one subchannel is desired. These multiplexing units need not remember subchannel selection; all input/output transfers are coded directly to the subchannel.

As each service request is sensed in the L-304 input/output control section (IOC), a control word (key word) uniquely associated with that subchannel is accessed. The key word, which is stored in memory, indicates the transfer mode and data memory location as well as the transfer count for that subchannel. The completion of the discrete data transfer on that subchannel includes updating the key word and replacing it in memory. The programmer is unaware of the actual data transfer until the transfer is terminated. At that instant, the program operation is interrupted and the Program Activity Register adjusted (see Program Level Access discussion to follow). The program level specified in the input/output termination word (also uniquely associated with that subchannel) specifies the program level to be activated. This control word contains the termination cause for subsequent program inspection and contains two, program-established program level designation fields, one for a normal termination (transfer count decrements to zero) and one for an "abnormal" termination (device error or intentional interrupt).

The previous paragraph, although an L-304 operation description, nearly details the input/output operation of the AN/USQ-20B as well. The primary difference is, in the AN/USQ-20B, a data transfer termination results in the passing of computer control to a fixed memory location unique to each input/output channel. This location is usually preset by the programmer with a transfer instruction to an input/output servicing program. In the L-304 computer, control is transferred directly (no intermediate steps necessary) to the program level indicated in the input/output termination word.

The input/output termination operation of the System 4 PI is considerably less flexible than the L-304 for the real-time system application. It is readily apparent that, in a typical commercial, multiuser, diverse tasking environment, full computer and input/output control is best left to an omniscient supervisor program. Anything short of this would lead to wasted machine time and thoroughly disgruntled users. However, in a real-time system, fast, precise input/output control is mandatory for maximum system effectiveness.

An input/output termination, in the 4 PI, results in the input/output control section interrupting the CPU to an input/output servicing routine. Unfortunately, this interruption is not unique to a subchannel; thus, all input/output terminations, of any sort, must filter through one servicing routine. In a typical ATDS, a minimum of 300 to 400 input/output terminations (excluding clock interrupts) must be processed each second. The source determination and routing decisions required by the 4 PI for this routine (which are unnecessary in the L-304) are not trivial, as was brought out in Section 2. A comparison of the event sequences of both the System 4 PI and L-304 machines is provided in Appendix C of this volume.

Real-Time Clock

Tactical functions often are operated on a prespecified time base. This base may be arbitrarily selected or may be the result of input/output operations. In either case, it is necessary to activate a subroutine within a minimum tolerance of relative time. In order to satisfy this requirement, a "clock-watching" process is necessary, i.e., some procedure must be established to assure that, at selected intervals, a real-time clock is interrogated and subsequent decisions regarding program routing are made. Traditionally, this requires either a clock monitor subroutine or the performance of clock checks at regular intervals in all programs. Also, various counters representing the desired time bases must be incremented (or decremented) and, if appropriate, the counters be reset and computer control be transferred to some other program "whose time it is to run." (See Appendix E.)

The facility for many real-time clocks, each capable of program interruption, frees the programmer from tedious and time-consuming clock watching duties.* A total of 64 real-time clocks are available in the L-304 since its IOC section merely references a program-established transfer mode to determine if a channel is being employed as a clock input device. If the key word transfer mode (discussed under "Input/Output Channels") for the subchannel currently requesting computer service equals 1, the IOC merely decrements the key word block length count. When the count decrements to zero, the normal input/output termination sequence is performed. This feature is discussed to some detail in Section 4.

Advanced L-304 Features

Items 6 through 8 of Table III-1 indicate those features of the L-304 and System 4 PI not provided by the AN/USQ-20B. Similarly, Items 9 through 11 indicate features unique to the L-304. All of these features, with the exception of Items 6 and 7, were previously available as software techniques. That is, Items 8, 9, 10, and 11 represent features which can be accomplished by programmed instructions. All of these processes are performed by the L-304 with no additional programming required.

Items 7, 9, and 10 are elaborated upon in Section 4. Item 8 is self-explanatory with the following comments provided for amplification.

The System 4 PI computer has optional floating-point-arithmetic logic. However it is not included here since this discussion is concerned primarily with military system applications.

The L-304 assembler was developed on both the System 360 (prior to L-304 availability) and the L-304. Appendix B of this volume contains a comparative summary of the instructions required in both computers for this application. Briefly stated: Appendix B indicates that the expanded repertoire of the L-304 made it possible to implement the same problem on the L-304 while requiring 17 percent fewer instructions than were required with the 360.

The thorough literal addressing capability of the L-304 provides a considerable time savings. In a typical ATDS program approximately 40 percent of all instructions (including transfers) might employ the literal addressing mode. Since this mode allows one less memory access per instruction (the operand is located in the instruction's A-field) than the direct mode, for example, total program operating time would be considerably abbreviated.

*A single clock capable of program interruption alleviates this problem but still requires a control routine to maintain the time base counters and determine program routing.

Memory Accessing

Item 6 lists the L-304 feature of program-controlled memory access which, in a modular fashion, enables memory growth to a maximum configuration of 131,072 32-bit word locations. At any single instruction execution, the L-304 is actively connected to four memory modules, equivalent to the capacity of the AN/USQ-20B. However, in the L-304, the programmer employs a unique instruction to modify the memory bank assignment register, reconfiguring the module selection to access additional locations. This memory selection is unique to each program level, a program on one level being unaffected by module selection of a program on a different level. Also, the selection is automatically saved, in memory, when leaving a level and restored when returning to that level. Appendix G contains a brief description of the memory module selection feature (with reference to expansion to 16 modules).

The L-304's total accessibility to 16 (8192-word) modules falls far short of the System 4 PI memory capacity.* However, the L-304's modular selection scheme allows dynamic system modification in the event of memory failure, a capability which is unavailable in either of the other computers. The L-304 instruction repertoire uniquely includes two memory adjustment instructions which allow the programmer to dynamically adjust the memory bank selection and isolate the failed module from the system. Assuming a spare memory module is included in the system, the L-304 tactical program could easily substitute the spare module for the failed one, reload only the substitute module from magnetic tape if necessary, and continue the complete tactical operation.

Input/Output Data Transfer

The L-304 IOC section enables the user to carry out data transfer either with full 32-bit words or 8-bit characters. When using the character mode, the data word packing and unpacking is performed with no program intervention necessary. In an ATDS environment, significant time savings are realized with this feature when real-time data extraction is desired. Since no unpacking is required, as would be in the AN/USQ-20B, all data areas can be directly output to the 9-channel (8-bit characters plus parity bit) magnetic tape unit with no intermediate formatting.

*The business data processing environment which the System 360 is accustomed to, generally imposes far greater memory capacity demands than most tactical systems.

In the System 4 PI, all data transfer occurs in 8-bit characters.* Although data packing and unpacking into 32-bit words is inherent to the scheme, additional logic would be required in the majority of military system input/output interface units to encode or decode the data into these characters. Also, the input/output servicing time is quadrupled because four input/output service requests are necessary for each single request to the L-304.**

PROGRAM ACTIVITY REGISTER OPERATION

Perhaps the most interesting and least understood features of the L-304 are automatic program queueing and triggering. Subsequent paragraphs elaborate on the use of the queueing and triggering mechanisms in the L-304. This subsection attempts to briefly explain these procedures to the unacquainted observer. A firm understanding of these is desirable in order to fully grasp and appreciate the forthcoming discussions.

Program Level Access

The Program Activity Register, depicted in Figure 3-1, occupies four memory locations and has two memory bits for each program level – an enable bit and a status bit. The enable bit is a program mask (only modified by instructions) controlling level activation and the status bit (set by either instruction or I/O termination) indicates current level activity. Whenever a particular instruction is executed or whenever an I/O-initiated store cycle into the program activity register is detected, a search of the PAR is performed to determine the highest priority active program level. This search consists of a logical comparison of the bits in the enable section with the bits in the status section.

The L-304 automatically "searches" the PAR by accessing each PAR memory word in turn (the enable and status bits for Levels 60-77 are in the most significant word, Levels 40-57 in the second word, and so on) and logically ANDing the enable

*In the 360 commercial applications, the primary input/output media are magnetic tape units and discs each of which store data bytes (8-bit character).

**This time is not always subtracted from instruction execution time since the System 4 PI input/output section can be independent from the CPU. In any event, in a full multiplex operation with several input/output devices, the IOC must ensure rapid response to all requests or risk data loss.

	PROGRAM LEVEL ENABLE BITS																PROGRAM LEVEL STATUS BITS																
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	BIT NUMBER
MEMORY LOCATION																																	
	77	76	75	74	73	72	71	70	67	66	65	64	63	62	61	60	77	76	75	74	73	72	71	70	67	66	65	64	63	62	61	60	OCTAL LEVEL NUMBER
1606	1			1										1	1																1		
	57	56	55	54	53	52	51	50	47	46	45	44	43	42	41	40	57	56	55	54	53	52	51	50	47	46	45	44	43	42	41	40	
1604		1	1		1	1	1		1		1	1	1		1												1						
	37	36	35	34	33	32	31	30	27	28	25	24	23	22	21	20	37	36	35	34	33	32	31	30	27	26	25	24	23	22	21	20	
1602	1	1	1	1	1				1		1		1		1					1													
	17	16	15	14	13	12	11	10	7	6	5	4	3	2	1	0	17	16	15	14	13	12	11	10	7	6	5	4	3	2	1	0	
1600	1		1		1		1		1		1						1						1		1		1						

Figure 3-1. Typical ATDS Program Activity Register Configuration

1601-3

bits with the status bits. If a logical "1" is detected after the AND operation, that level is activated. If a "1" does not exist, the next word is accessed and the operation is repeated on that word. The search continues until a "1" is detected or until all 64 levels have been checked. Level 0 is activated if no other level satisfies the activation criteria.

The search is always performed from the most significant enable and status bits to the least significant. This results in the current highest priority program level assuming computer control. The total operation takes only 2.5 to 10 microseconds, depending on the number of memory words that must be accessed.

For example, assume the following PAR configuration (in this example assume only six program levels):

Program Activity Register		
5 4 3 2 1 0	←— Level Number —→	5 4 3 2 1 0
0 1 0 1 ① 1		0 0 0 0 ① 0
Enable (Mask) Section		Status (Activity) Section

With this configuration, program Level 1 would be active because this level is the highest with a logic "1" common to both its enable bit and status bit.

If the program operating on Level 1 wished to activate Level 4, it would merely set the status bit for Level 4:

5 4 3 2 1 0	5 4 3 2 1 0
0 1 0 1 1 1	0 ① 0 0 1 0

Level 4 would be activated because it is the highest priority "active" level. If Level 1's program had set the Level 0 status bit, a search would be performed that would result in control returning to Level 1 because Level 0 is a lower priority level.

Similarly, if an I/O interrupt resulted in Level 2's status bit being set,

5 4 3 2 1 0	5 4 3 2 1 0
0 1 0 1 1 1	0 0 0 ① 1 0

Level 2 would be activated.

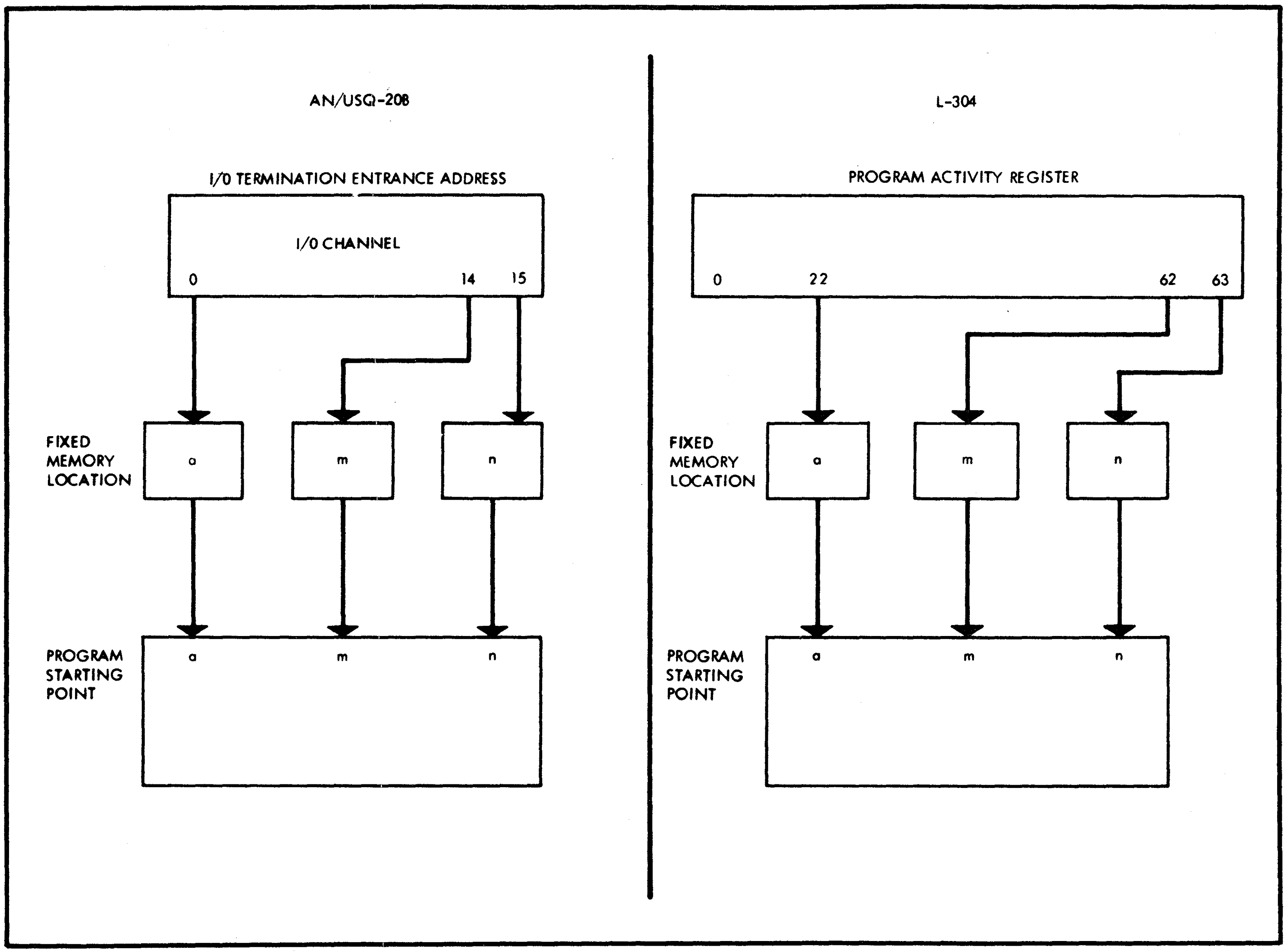


Figure 3-2. Program Queueing Comparison

However, if the interrupt results in a selection for Level 3,

5 4 3 2 1 0	5 4 3 2 1 0
0 1 0 1 1 1	0 0 $\textcircled{1}$ 0 1 0

Level 1 remains active because Level 3 does not have a logic "1" in both its enable bit and status bit.

Program Triggering

The procedure for program level activation, once the PAR search is completed, is similar to the I/O servicing routine triggering sequence in the USQ-20B. Figure 3-2 illustrates this similarity.

As the figure indicates, the L-304 process of activating a program level is very similar to the USQ-20B process of activating a subroutine after an external interrupt or I/O transfer termination. When an external interrupt or I/O data transfer termination is sensed in the USQ-20B, control is directed to a fixed memory location uniquely associated with the interrupting I/O channel. This location is generally preset with a transfer instruction to an I/O servicing routine.

In the L-304, when a program level is to be activated, the computer's instruction location register is set with the contents of a fixed memory cell associated with that program level and the computer begins executing instructions at that location on the new program level. When a level is exited, the instruction location register contents are stored in the unique location for that level. This enables the program to resume operation where it left off when the level is reentered.

SECTION 4

TYPICAL ATDS ENVIRONMENT

GENERAL

The L-304 computer system in a typical airborne tactical data system could utilize the many L-304 advanced design features to enhance system operation. These features have been discussed in Section 3. The purpose of this section is to elaborate upon this discussion with examples of a typical ATDS program design.

MULTIPROCESSOR CONFIGURATION

An ATDS multiprocessor configuration, as presented in Figure 4-1, might consist of: (1) two processors, (2) five 8K word memory modules, each accessible from both processors, and (3) a control I/O interface unit which houses interprocessor communication logic and real-time clock, magnetic tape unit, IFPM panel, and computer control panel interface logic.

This multiprocessor configuration would allow for over 160 percent system functional operation expansion, based on a worst-case system load, as well as greatly increasing total system reliability in several forms.

System Expansion Capability

In earlier tactical systems, the operating capacities of the systems were generally exceeded with little effort expended toward improvements on the original design. Since functional improvements are inevitable, a design which allows more than 100 percent additional real time entirely for system modification and expansion is clearly preferable over one which does not. The three- and four-computer configurations now used within other tactical systems are examples of unanticipated expansion. The time-consuming communication necessary in a multicomputer configuration (all via I/O) results in the requirement for three separate computers to do the job of two having common memory access.

Considering only a typical AEW mission, the total maximum system load can easily be handled in a single L-304. These percentages indicate that a

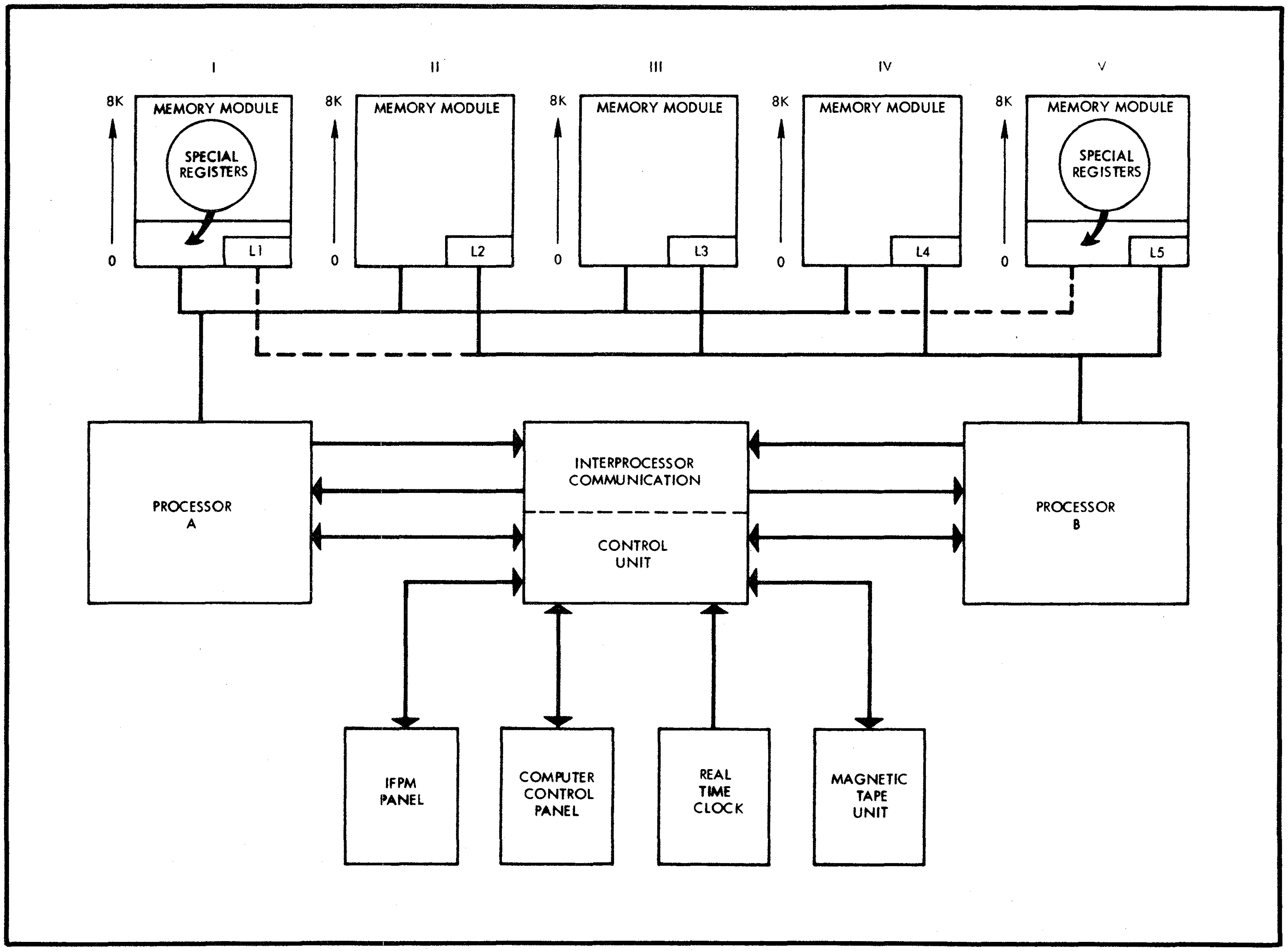


Figure 4-1. Typical ATDS Multiprocessor Configuration 1601-5

second processor would be totally available for expanded functional responsibilities. The additional responsibilities may easily encompass an improved version of the tactical program, as well as additional assignments such as ECM or ASW processing.

System Reliability Enhancement

In a typical "ATDS", each processor would have the ability to communicate with each memory module (up to 10 modules) and with all of the I/O interface assemblies. This communication capability allows both processors to perform dynamic system corrective action in the event that failures occur that require system degrading. A small amount of logic in each I/O unit is uniquely associated with each processor. If the failure occurs in this area, the I/O unit will maintain a meaningful data transfer path when connected to the other processor. In a typical tactical program design, if either processor detects an error in an I/O interface assembly (periodic, program-controlled data-transfer tests are performed on each converter), the failed assembly will be automatically (no operator action necessary) switched over to operate from the other processor. This relocation will be performed in such a manner that the operator and functional programs will not be aware of the switch. (Note: This is not a degraded operation.) However, if the failure occurs in the logic common to both processors, it will be detected in both processors, the I/O will be idled if necessary, and the operator will be notified of the error.

Processor Verification

Perhaps even more valuable is the processor verification feature provided by the multiprocessor configuration. Since it is possible to maintain full system operation in a single processor and to shift the tactical load between the processors, one processor can assume the responsibilities of the other (failed) processor as well as maintaining its own. Also, the good processor can disable the bad processor to prevent unanticipated harmful actions in the bad processor. More specifically, in the retrofit system, when a processor detects a General Machine Test error in the other processor (the GMT runs once per scan in each processor and the result is verified by the other processor respectively), it will assume the full system operation and will disable the other processor.[*]

[*]This powerful capability must be treated carefully but is nevertheless valuable. Software traps are set up such that a failed computer cannot disable a functioning computer.

In future expanded operations where one processor is performing tactical operations and the other ECM, for example, if a processor fails, the system operator will be able to select what operation or even what subfunction mix he wishes to maintain. The good processor will automatically assume those responsibilities regardless of its prior function.

PROGRAM PRIORITY STRUCTURE

In a conventional computer, the function of program sequencing and triggering falls upon an executive routine. This "master caller" traditionally performs a series of decisions, often employing a queueing file, to determine the order of programs it is to run in the normal sequence of events (see Appendix D). A clock-watching routine usually is included in this category also, its function being to monitor the system real-time clock and to provide the master caller (or do the calling itself) with supplementary information as to the order of events (see Appendix E).

This effect is precisely what is provided via the L-304 Program Activity Register with the exception that the queueing file interrogation is performed automatically. After a System Initialization routine has preestablished the Program Activity Register the dynamic sequencing of events will continue without necessity for program control.

An ATDS Program Priority Structure

A typical priority structure for an ATDS Retrofit Program including the program level number, program name, and procedure by which that level/program would be accessed. The access media column containing another level number indicates the program at that other level is directly setting the PAR status bit for this program level. Activations caused by clock terminations indicate that a prespecified number has been counted down to zero by continuous one-kilocycle clock pulses on the associated I/O channel (see Section 3). The I/O terminations occur when a data transfer is complete or an external interrupt is received into the L-304.

Program Sequencing

It has already been pointed out that the program priority determination and triggering in a conventional computer is accomplished with a "master caller" subroutine and some scheme of periodically monitoring the real-time clock. This approach generally allows for a fixed sequence of processing functions and a higher priority, immediate-service set of I/O control routines. The servicing priority of the latter routines is always determined by the most recent interrupt to the computer.

In future expanded operations where one processor is performing tactical operations and the other ECM, for example, if a processor fails, the system operator will be able to select what operation or even what subfunction mix he wishes to maintain. The good processor will automatically assume those responsibilities regardless of its prior function.

PROGRAM PRIORITY STRUCTURE

In a conventional computer, the function of program sequencing and triggering falls upon an executive routine. This "master caller" traditionally performs a series of decisions, often employing a queueing file, to determine the order of programs it is to run in the normal sequence of events (see Appendix D). A clock-watching routine usually is included in this category also, its function being to monitor the system real-time clock and to provide the master caller (or do the calling itself) with supplementary information as to the order of events (see Appendix E).

This effect is precisely what is provided via the L-304 Program Activity Register with the exception that the queueing file interrogation is performed automatically. After a System Initialization routine has preestablished the Program Activity Register the dynamic sequencing of events will continue without necessity for program control.

ATDS Program Priority Structure

Table IV-3 depicts the priority structure for the ATDS Retrofit Program including the program level number, program name, and procedure by which that level/program is accessed. The access media column containing another level number indicates the program at that other level is directly setting the PAR status bit for this program level. Activations caused by clock terminations indicate that a prespecified number has been counted down to zero by continuous one-kilocycle clock pulses on the associated I/O channel (see Section 3). The I/O terminations occur when a data transfer is complete or an external interrupt is received into the L-304.

Program Sequencing

It has already been pointed out that the program priority determination and triggering in a conventional computer is accomplished with a "master caller" subroutine and some scheme of periodically monitoring the real-time clock. This approach generally allows for a fixed sequence of processing functions and a higher priority, immediate-service set of I/O control routines. The servicing priority of the latter routines is always determined by the most recent interrupt to the computer.

LEVEL NUMBER	PROGRAM	ACCESS MEDIA
77	PROCESSOR RESET	COMPUTER RESET
75	TRACER	PAR STORE CYCLE
74	MEMORY CHECK	LEVEL 7
62	IFPM AND SYSTEM CONTROL	LEVEL 61
61	IFPM AND SYSTEM CONTROL	CLOCK, LEVEL 27, OR I/O
56	SPARE CLOCK CHANNEL CONTROL	CLOCK
55	MISCELLANEOUS I/O ERRORS	I/O
53	CD BUFFER TERMINATION	I/O
52	ADD TRACK TO SYSTEM	LEVELS 27, 21, 15, 13
51	LINK-11 I/O ERROR OR OUTPUT INTERRUPT	I/O OR LEVEL 47
47	LINK-11 BUFFER TERMINATION	I/O OR LEVEL 25
45	DISPLAY BUFFER TERMINATION	I/O OR CLOCK
44	DISPLAY NIXIE CONTROL	I/O OR LEVEL 27
43	LINK-4 INPUT PROCESSOR	I/O
41	LINK-4 OUTPUT PROCESSOR	I/O
37	ERROR CONTROL	I/O
36	INITIATION CONTROL	CLOCK OR I/O
35	DISPLAY EQUATIONS	LEVEL 34
34	CRITICAL LOOP EQUATIONS	LEVEL 33
33	I/O CONTROL	CLOCK OR I/O
27	DISPLAY INPUT PROCESSOR	LEVEL 45
25	LINK-11 OUTPUT PROCESSOR	LEVELS 51, 47
23	CD INPUT PROCESSOR	LEVEL 27
21	GENERAL BOOKKEEPING (INCLUDING DISPLAY OUTPUT FILE UPDATING)	LEVEL 27
17	INTERCEPTION CALCULATIONS	LEVELS 11, 5
15	CORRELATION AND TRACKING	LEVEL 23
13	LINK-11 INPUT PROCESSOR	CLOCK
11	INTERCEPTION EXECUTIVE	LEVEL 15
7	GENERAL MACHINE TEST	LEVEL 61
5	INTERCEPTION TEWA EXECUTIVE	

Table IV-3. ATDS Program Priority Structure

The L-304 allows for the same orderly sequence of processing functions while not requiring any "master caller" routine. The program operation time saved, which is traditionally wasted in the "master caller's" intricate decision paths,* can be considerable (5 to 10 percent). In the L-304, this time becomes available for more meaningful functional operation. Also program efficiency and modularity is enhanced since all programs are no longer dependent on some master program and, similarly, never have to determine if it is time to return control to the "master caller." The same I/O servicing procedure of the conventional computer is permissible in the L-304 with the added feature of realistic priority determination among these I/O controllers. The last I/O interrupt need not be the first processed; if it is a lower priority than a previous interrupt, it will await its turn.

Program Priority Modification

The priority arrangement in Table IV-3, which takes into consideration running time, relative priorities, and the required system response time of each program, is by no means inflexible. An alteration can easily be incorporated; only a program assembly is necessary. All tactical programs have been coded irrespective of their associated program level. A permanent priority adjustment would entail only the changing of a few cards in the common data communications pool (compool) and reassembling the tactical program.

Futhermore, immediate dynamic adjustment of the system operating sequence is facilitated in the L-304. It is readily appreciated that, in a conventional computer, the dynamic adjustment concept would require additional decision paths in the "master caller" routine. For example, flags are necessary to indicate and remember a system failure which dictates a change in the normal program sequencing. After determining that it is program X's turn to operate, the "master caller" must then inspect the flag for that program to ascertain if any sytem failures had occurred such that program X should not be called at this time. This decision logic involves several instructions and is exercised frequently.

The program activity register concept greatly facilitates in-flight program flow modification to suit the current system environment. The following two examples of dynamic priority adjustment are provided to illustrate this point.

*These paths may not be lengthy, but frequent operation equals significant time delay.

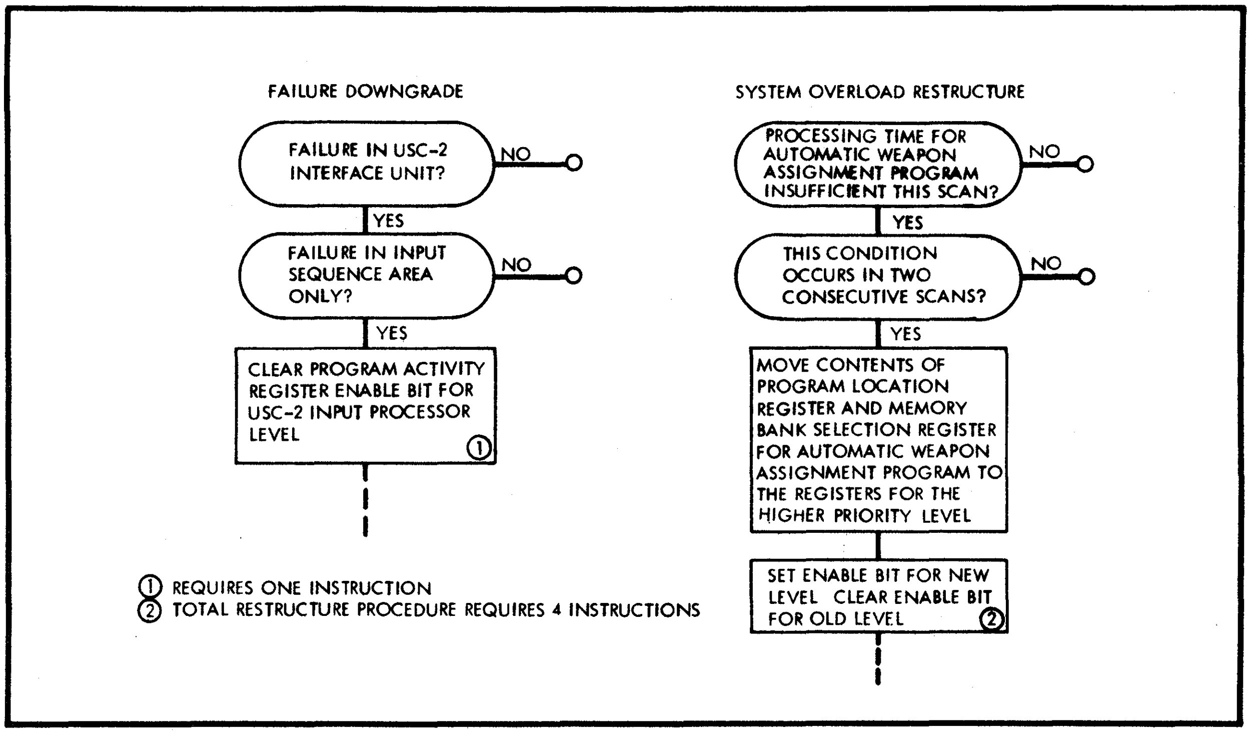

FAILURE DOWNGRADE
FAILURE IN USC-2 INTERFACE UNIT?
NO
YES
FAILURE IN INPUT SEQUENCE AREA ONLY?
NO
YES
CLEAR PROGRAM ACTIVITY REGISTER ENABLE BIT FOR USC-2 INPUT PROCESSOR LEVEL ①
① REQUIRES ONE INSTRUCTION
② TOTAL RESTRUCTURE PROCEDURE REQUIRES 4 INSTRUCTIONS
SYSTEM OVERLOAD RESTRUCTURE
PROCESSING TIME FOR AUTOMATIC WEAPON ASSIGNMENT PROGRAM INSUFFICIENT THIS SCAN?
NO
YES
THIS CONDITION OCCURS IN TWO CONSECUTIVE SCANS?
NO
YES
MOVE CONTENTS OF PROGRAM LOCATION REGISTER AND MEMORY BANK SELECTION REGISTER FOR AUTOMATIC WEAPON ASSIGNMENT PROGRAM TO THE REGISTERS FOR THE HIGHER PRIORITY LEVEL
SET ENABLE BIT FOR NEW LEVEL CLEAR ENABLE BIT FOR OLD LEVEL ②

Figure 4-2. Program Activity Register Usage

Failure Downgrade

A typical system failure downgrade procedure within the L-304 is shown in the left half of Figure 4-2. In a typical system, various I/O unit failures may dictate a system downgrade. Specifically, a particular test failure in the AN/USC-2 Data Terminal I/O interface unit indicates the interceptor reply data is likely to be garbled. Hence, it is desirable to avoid processing that information. When the test failure occurs, the IFPM and System Control Program will reset the PAR enable bit for the program level assigned to AN/USC-2 input processing. As long as the enable bit is reset, the AN/USC-2 program will never be accessed. The L-304 required one instruction, in this instance, to accomplish the same effect as a conventional computer with several instructions, repeatedly executed.

Overload Degrade

The right half of Figure 4-2 depicts a hypothetical priority modification based on a system overload. It is presented as further evidence of the ease with which dynamic priority adjustment is accomplished in the L-304.

The postulated situation consists of elevating the automatic weapon assignment function to a higher operating priority when the system load becomes sufficiently high that insufficient operating time is allotted to this program. This capability, although not included in the present program design, requires the inclusion of only a few instructions in the IFPM and System Control Program to note a system overload condition. These infrequently executed instructions would verify the consistency of the environment and alter the program priority structure accordingly.

This priority restructure has the effect of allocating more time for the automatic weapon assignment operation. As is evident from Figure 4-2, the rearrangement is complete with the execution of only four instructions. From that point on, the automatic weapon assignment program, which is unaware of the modification, will operate on a new program level of higher priority; thus, more real time is made available to this function. The same sequence of four instructions will reconfigure the system to its original priority arrangement when the system load returns to normal.

REAL-TIME CLOCK

The ideal real-time system design should be one in which a minimum of time is spent making decisions on program sequencing as related to real-time control. This is the sole function of a clock-watching sequence. A scheme which automates this approach, thereby making this time (traditionally spent in the "who's next loop") available for useful processing is clearly preferable.

This concept is precisely what is provided in the L-304. Each of the listed programs would employ its own real-time clock and would be activated automatically when the associated clock-channel block length decrements to zero. Thus, each function would operate independently, on its own time base, irrespective of other functions. Once again, program efficiency and modularity have been increased.

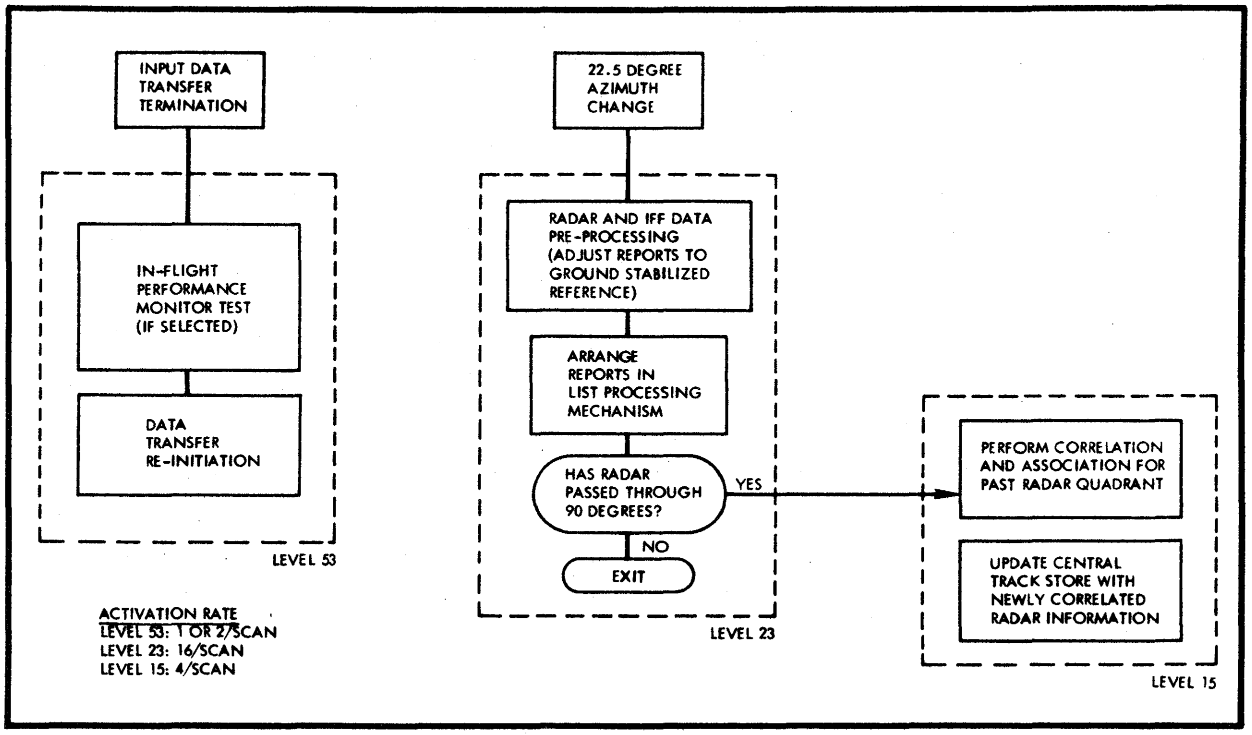

INPUT DATA TRANSFER TERMINATION
IN-FLIGHT PERFORMANCE MONITOR TEST (IF SELECTED)
DATA TRANSFER RE-INITIATION
LEVEL 53
ACTIVATION RATE
LEVEL 53: 1 OR 2/SCAN
LEVEL 23: 16/SCAN
LEVEL 15: 4/SCAN
22.5 DEGREE AZIMUTH CHANGE
RADAR AND IFF DATA PRE-PROCESSING (ADJUST REPORTS TO GROUND STABILIZED REFERENCE)
ARRANGE REPORTS IN LIST PROCESSING MECHANISM
HAS RADAR PASSED THROUGH 90 DEGREES?
YES
NO
EXIT
LEVEL 23
PERFORM CORRELATION AND ASSOCIATION FOR PAST RADAR QUADRANT
UPDATE CENTRAL TRACK STORE WITH NEWLY CORRELATED RADAR INFORMATION
LEVEL 15

Figure 4-3. Multilevel Usage

MULTILEVEL USAGE

The L-304's program level concept allows the system designer to modularize the functional areas of the tactical program to increase program efficiency. In a typical system, the tactical program might be subdivided even further; the components within a functional area would be assigned to unique program levels wherever it is operationally expedient.

Figure 4-3 illustrates this point by depicting the level assignments of three components of a Correlation and Association Program. This program is naturally grouped into three distinct phases:

(1) Input data transfer reinitiation is performed on a high priority program level. This includes no data processing; only an IFPM test, if selected, and reinitiation of the data transfer.

(2) The computer-detector input data is preprocessed several times per radar scan. This preprocessing consists of ground stabilizing the radar and IFF reports and inserting them in the list processing linkages used in the correlation process.

(3) The main correlation program operates four times each radar scan. Immediately following its operation, the central track store is updated with the newly correlated radar information.

These phases of the correlation function are divided over three program levels to increase program efficiency. Each phase operates on a unique time base, independent of the other phases. Also, the operating time of these components varies from very short to extremely long. Employing three levels provides a quick response time, when necessary, relative operating priority with the other tactical functions, and the use of distinct processor registers for each independent subfunction. The same total effect is obtainable in other computers at the expense of extra instructions and decreased program modularity.

Processor Register Preservation

Multilevel allocation is also useful when considering the traditional process register storing and restoring necessary when independent subroutines are accessed. In a conventional computer, the first step of an I/O servicing routine, for example, is to save the present contents of the process registers (so as not to affect the operation of the interrupted program). Similarly, when leaving the routine, the registers are restored (see Appendix C). This operation requires several instructions and extra memory space (a set of locations for each I/O servicing routine).

In the L-304, the same number of memory locations is necessary because the process registers are actually memory locations. However, the sequence of storing and restoring the registers is performed automatically, thus saving considerable time. A recent study (see Appendix A) indicated that the sequence of register preservation in the USQ-20B required 30 times the computer operating time of the L-304. With 17 I/O channels and frequent I/O transfer terminations (several hundred per second), the time saved by the L-304 would be very significant.

The discussion thus far has been confined to the procedures inherent to real-time I/O servicing. Actually, the same procedure is necessary when any subroutine is accessed from multiple sources. Figure 4-4 illustrates a typical use of the L-304 automatic process register storage feature. The example selected is not an I/O servicing routine, but rather is a subroutine which controls the process of locating available positions for new system tracks in the central track store. As is evident from the example, by employing a unique program level for this subroutine, a considerable time savings is realized. The effect is identical to a conventional scheme but the L-304 performs the selected task with only two additional instructions while 18 instructions are necessary in the conventional computer.

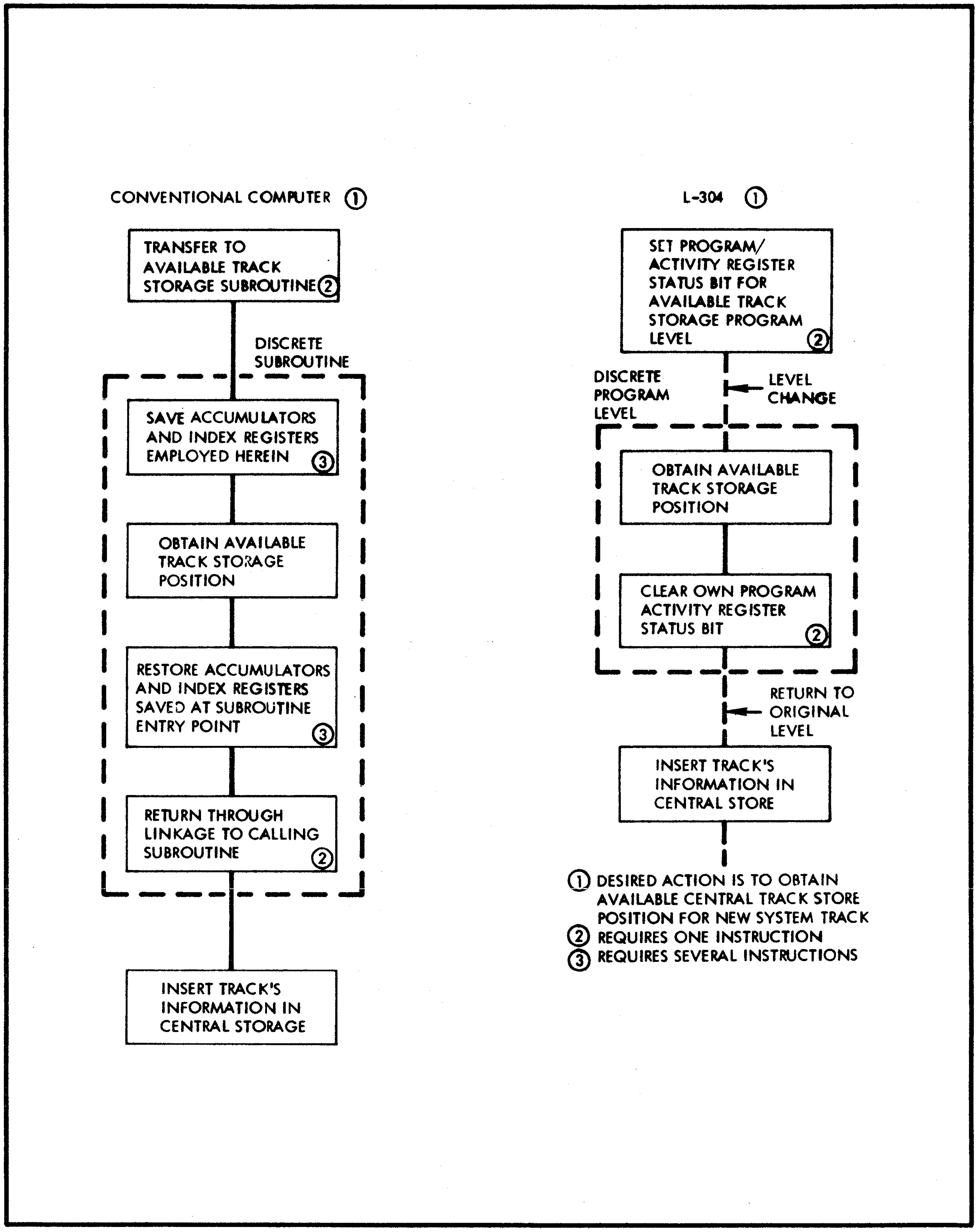

CONVENTIONAL COMPUTER ①
TRANSFER TO AVAILABLE TRACK STORAGE SUBROUTINE ②
DISCRETE SUBROUTINE
SAVE ACCUMULATORS AND INDEX REGISTERS EMPLOYED HEREIN ③
OBTAIN AVAILABLE TRACK STORAGE POSITION
RESTORE ACCUMULATORS AND INDEX REGISTERS SAVED AT SUBROUTINE ENTRY POINT ③
RETURN THROUGH LINKAGE TO CALLING SUBROUTINE ②
INSERT TRACK'S INFORMATION IN CENTRAL STORAGE
L-304 ①
SET PROGRAM/ ACTIVITY REGISTER STATUS BIT FOR AVAILABLE TRACK STORAGE PROGRAM LEVEL ②
DISCRETE PROGRAM LEVEL
LEVEL CHANGE
OBTAIN AVAILABLE TRACK STORAGE POSITION
CLEAR OWN PROGRAM ACTIVITY REGISTER STATUS BIT ②
RETURN TO ORIGINAL LEVEL
INSERT TRACK'S INFORMATION IN CENTRAL STORE
① DESIRED ACTION IS TO OBTAIN AVAILABLE CENTRAL TRACK STORE POSITION FOR NEW SYSTEM TRACK
② REQUIRES ONE INSTRUCTION
③ REQUIRES SEVERAL INSTRUCTIONS

Figure 4-4. Process Register Usage

1601-11

SECTION 5

SUMMARY

The intent of the foregoing sections has been to point out the applicability of the L-304's design features to real time military systems. If some of the features appear unconventional, it is only because they are uniquely organized. These features represent the culmination of a computer development characterized by the ever-increasing inclusion in computer hardware of those well defined and repetitive processes formerly accomplished partly by hardware and partly by software. The objectives of this unique synthesis of features are the minimization of programming effort and minimization of tactical real time requirements. Ultimately, these features allow the system designer to enhance his design by utilizing the inherent power and efficiency of the L-304 computer.

Litton believes that these features make the L-304 computer superior to any other computer for military real-time computer systems, and, in general, for almost any application.

APPENDIX A

COMPARISON OF REGISTER PRESERVATION REQUIREMENTS

In the L-304, 256 32-bit words are available for use as process registers. In a conventional computer, approximately the same amount of storage is required for bookkeeping associated with storing and restoring registers. A breakdown for the USQ-20B computer is as follows:

	Words
Register storage (7 index registers, 2 arithmetic registers) assuming 17 I/O channels (as in ATDS) and 6 words of storage for registers per channel: 6 x 17 = 102	102
General subroutine to store and restore registers (store all registers on entry, restore all registers on exit)	20
Linkage to "register store" and "restore" subroutines (assuming equivalent of 64 L-304 levels), 2 transfer and links per function	128
	250

Based on this comparison, the L-304 uses 6 extra storage locations for register handling. The L-304 real-time advantage for register processing is about 30 to 1 for each I/O interrupt; that is, 6 microseconds for the L-304 versus 176 microseconds for the USQ-20B (22 instructions[*] at 8 microseconds per instruction).

[*]Two transfer and link instructions and 20 instructions to store and restore the registers.

APPENDIX B

COMPARISON OF L-304 ASSEMBLER ON L-304 AND ON IBM 360

NATURE OF PROGRAM

The L-304 assembler is a two-pass assembler with a floating field operand. It utilizes several alphabetized tables containing a compool (a communications pool), internal symbols, external symbols, and subroutine entry points. Data may be defined symbolically, as a decimal number, octal number, hexidecimal number, or as any combination of these. The programs compared perform identical functions.

The 360 program was written under 8KBOS and the L-304 program was written using L-304 assembly language and was assembled by the 360 assembler.

INSTRUCTION AND WORKING STORAGE COMPARISON

Module	L-304	360
First Pass	737	2093
Second Pass	1177	2058
Operand Interpreter	1285	1040
Compool Storage	549	0
TOTAL	3548 32-bit words	5191 32-bit words

DISCUSSION

The 360 version is divided into three separate modules (assembly decks); the L-304 program consists of twenty modules. The reason for the larger number of modules in the L-304 (in addition to the resulting update and debugging economies) is the comparative ease of linkage between modules. The relative difficulty of linkage in the 360 (caused by the base registers) causes common routines to be included in more than one module. If the length of these routines is subtracted from the 360 program, the storage comparison would look as follows:

Module	L-304	360
First Pass	737	1842
Second Pass	1177	1991
Operand Interpreter	1285	981
Compool Storage	549	0
TOTAL	3548 32-bit words	4804 32-bit words

The removal of the duplicate routines increases the complexity of using them and restricts the modularity of the program (causes both passes to be in core at all times). Much storage is saved by the L-304 addressing capabilities and ease of establishing and using a common data pool.

The program evolution in this instance should be mentioned. The original assembler was written for the IBM 7094; this program was then moved to the IBM 360. The move included program improvements because it was the second coding. Then, the same basic program was again moved to the L-304; additional improvements, although fewer in number, were inherent to this move.

Input and output control sections have not been included in the foregoing totals because so much of this area is performed by the BOS Supervisor in the 360. It was felt that including the total Supervisor in the I/O control figures would constitute an unfair evaluation.

CONCLUSIONS

The results of this comparison are extremely interesting when considering the nature of the program. One would perhaps expect the L-304 to require fewer instructions in a tactical system program because the L-304 was specifically designed to meet that type of requirement. Likewise, one would expect the 360 to excel in its design environment of primarily commercial applications. Considering this fact, Litton would have been pleased had the L-304 equaled the number of instructions required for the 360 on this assembly program, a type of program typically considered to be the 360's forte.

This is not the case. The L-304 improved upon the 360 requirements by more than 48 percent! Even when the duplicate subroutines in the 360 are eliminated, the L-304 excels by 35 percent. This is true even though the totals in the latter case do not include the extra instructions required in the 360 program to overcome the increased program complexity indicated.

APPENDIX C

COMPARISON OF I/O SERVICING REQUIREMENTS

SEQUENCE OF EVENTS UPON RECEPTION OF AN I/O INTERRUPT

System 4PI, Model EP	L-304
1. Transfer control to a processing subroutine regardless of the I/O channel (and inhibit further interrupts).	1. Transfer control directly to a processing subroutine, depending on the I/O channel.
2. Save process register contents.	2. Automatic
3. Construct a new Program Status Word for subsequent interrupts.	3. Status of interrupted program already saved and is not effected by subsequent interrupts.
4. Determine the interrupt source.	4. Inherent to Number 1.
5. Release interrupt lockout.	5. Unnecessary
6. Decide if an I/O servicing routine is to be called.	6. Inherent to Number 1.
7. Do processing (in I/O servicing routine).	7. Do processing.
8. Restore process registers.	8. Automatic
9. Return to interrupted program.	9. Return to interrupted program.

DISCUSSION

I/O servicing procedures are similar regardless of the computer under consideration. The process of immediately transferring computer control to a subroutine which can perform required actions has become a standard computer component. The design of L-304, however, goes beyond this standard component to accomplish automatically that which previously required programmed bookkeeping. Specifically, by employing unique program levels for I/O servicing, the process registers in use at the time of the interrupt are automatically "saved" and, when returning to the interrupted routine, "restored." Additionally, the L-304 allows the system designer, if he so desires, to establish priorities among the I/O servicing subroutines. This feature, not easily obtainable in a conventional computer's last-entered/first-processed operation, provides added system capability in the event highly critical I/O reinitiation times exist.

APPENDIX D

COMPARISON OF FUNCTIONAL PROCESSING REQUIREMENTS

SEQUENCE OF EVENTS FOR NORMAL FUNCTIONAL PROCESSING (OTHER THAN DIRECT I/O SERVICING)

Conventional Computer (All control begins with "master caller" subroutine)	L-304
1. Calling subroutine determines next subroutine to be called (calls various subroutines in a predetermined fashion) and transfers control to that subroutine.	1. Automatic*
2. Called subroutine is normally completed and control returns to the "master caller."	2. "Automatically" called subroutine relinquishes control to next highest priority subroutine.
3. "Master caller" determines next subroutine to be called and calls it.	3. Automatic
4. Repeat of Step 2.	4. Repeat of Step 2.
5. Repeat of Step 3, and so on.	5. Repeat of Step 3.
6. Whole cycle is repeated.	6. Dynamic sequencing continues automatically ad infinitum.

DISCUSSION

Once the program activity register in the L-304 is initially established, the priority determination and sequencing of programs proceeds automatically. Thus, the L-304 allows the system designer to eliminate the "master caller" subroutine. It is readily apparent that a system could be designed to operate on one program level within the L-304, thus creating the need for the traditional "master caller." However, with multilevel program organization, the features inherent to the L-304 can be employed to accomplish the very same operation automatically (no program control necessary) while increasing program efficiency and modularity.

*Requires initial (one time) setup of queueing file (program activity registers).

APPENDIX E

COMPARISON OF REAL-TIME CONTROL REQUIREMENTS

SEQUENCE OF EVENTS FOR REAL-TIME CONTROL

Conventional Computer	L-304
1. Inspect clock periodically, i. e., design program with clock-inspection sequence throughout coding (or pass through clock-watching sequence periodically).	1. Automatic with clock interrupt.
2. Update all clock counters; check for appropriate action.	2. Automatic (can have many independent clocks).
3. Take appropriate action and clear related clock counter if required.	3. Take appropriate action and reset related clock counter.
4. Return control to inspection point if action had been taken.	4. Automatic.

DISCUSSION

The feature of many real-time clocks, each capable of program interruption, saves many programmed instructions. As indicated in the above sequence the L-304 accomplishes automatically that which required many instructions in a conventional computer. It also represents an improvement over more recent computer's capability of a single interruptable clock. In the above sequence only step number one can be eliminated with a single clock.

APPENDIX F

L-304 COMPUTER DESCRIPTION

GENERAL MACHINE DESCRIPTION

The Litton L-304 is a highly reliable, fully microelectronic, high-speed general-purpose computer designed specifically for operation in airborne, shipborne, and ground-mobile environments. Its small size and high reliability are achieved by the exclusive use of integrated circuits and multilaminate intercircuit wiring compatible with its random-access, coincident-current microferrite core memory.

Maximum flexibility and expandability have been achieved by designing the L-304 computer logic to permit memory expansion (8192-word increments) from 8192 to 131, 072 32-bit words without logic modification. Expansion from single to multiprocessor operation is economical from the standpoint of cost and size. Table F-I presents comparisons of the power requirements and approximate* volumes and weights of representative processor configurations.

The basic characteristics of the L-304 computer are:

(1) Parallel binary operation

(2) 32-bit instruction word

(3) 16- or 32-bit data word, including sign

(4) 2.2-microsecond memory read/write cycle

(5) Two's complement arithmetic

(6) Memory expandable in 8192-word modules, 32 bits per word

(7) 65 basic instructions

(8) 3 addressing modes

(9) 64 program levels

(10) 8 multipurpose process registers per program level (512 total, usable as index registers or accumulators)

(11) Up to 64 input/output (I/O) channels each with program-initiated but independently operating data transfers of up to 32 bits in parallel

*Minor variations in volume and weight will result from characteristics of specific installations.

PROCESSOR	MEMORY (32-BIT WORDS)	POWER (WATTS)	VOLUME (CUBIC FEET)	WEIGHT (POUNDS)
SINGLE	8, 192	300	0.94	67
SINGLE	16,384	390	1.32	87
SINGLE	32,768	490	2.19	148
DUAL	32,768	780	2.44	164
DUAL	65,536	980	4.19	272
DUAL	131,072	1180	7.69	390

Table F-I. Comparison of Power Requirements, Volume, and Weight of Representative L-304 Configurations

(12) High I/O transfer rates, up to 432, 000 32-bit words per second on one channel or 227, 000, 32-bit words per second when several channels are operating simultaneously

(13) Automatic priority and high-speed multiprogram switching

(14) Coincident current, random access memory using wide temperature cores

The L-304 computing system is divided into four modular parts: the central processor, micromemory, input/output, and power supply. These are shown in block diagram form on Figure F-1.

As indicated in this diagram, the number of flip-flop registers is minimized and multiple use is made of each of the registers in the computer. This organization reduces size, weight, and power consumption, and lowers cost by decreasing the number of components.

THE CENTRAL PROCESSOR

The central processor contains five functional sections. These include the Arithmetic, Instruction Control, Memory Control, Program Level Control, and Input/Output Control Sections.

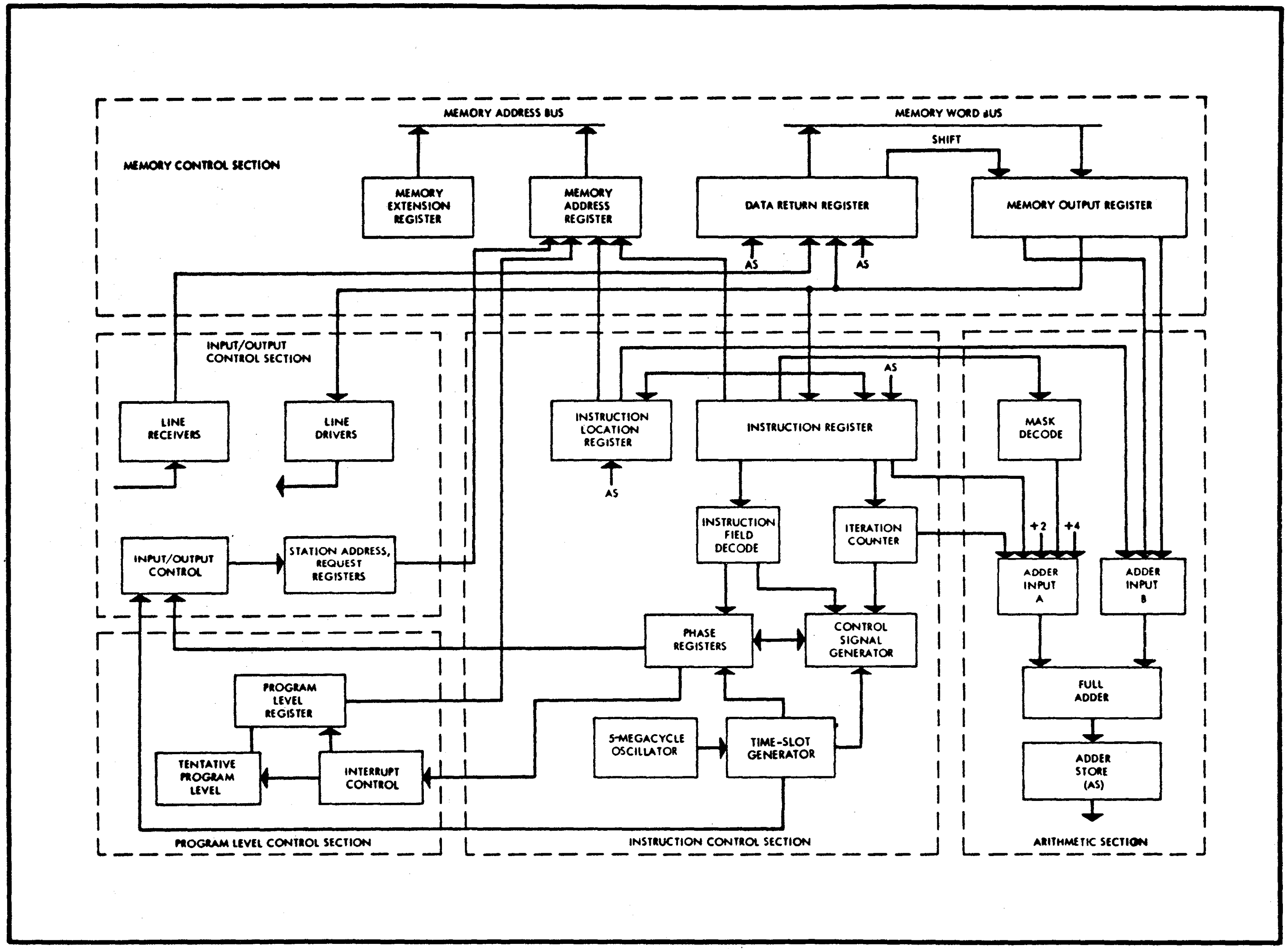


Figure F-1. L-304 Computer Block Diagram

The Arithmetic Section

The arithmetic section contains a parallel, full adder of 16 bits. This adder is used in the execution of most instructions. Carry bypass and look-ahead logic is used for a fast add time of less than 210 nanoseconds. Several logic controlled input sources to the adder are provided and selective output from the adder yields logic functions, sums, differences, decrements, increments, and zero detection.

The Instruction Control Section

The instruction control section contains a 32-bit instruction holding register and the timing control state counters. Instruction operation and mode decoding gates provide control of all processor operations. This section also contains the computer clock and control state counters.

The Memory Control Section

The memory control section is that part of the processor which controls memory address extension. It contains a memory address extension register which allows direct memory accesses to up to 16 modules (131, 072 full words or 262, 144 half words).

The Program Level Control Section

The program level control section determines which of up to 64 different programs are active on a priority demand basis. A program level register selects the active set of eight process registers and one of 64 program location registers from fixed memory locations to be used by the instruction control section. A program activity register of 128 bits is also stored in a fixed memory location. This register is used to reflect computer status with 64 active bits and 64 enable bits for programmed control of interrupts.

The Input/ Output Control (IOC)

The input/ output control (IOC) is that part of the processor that controls the transfer of data between the processor's memory and the transmitting or receiving I/O device. It controls the servicing of I/O requests for data transmission on a priority demand scheme. It provides for communication and all synchronization between the processor and each interface unit. Much of the processor's control unit and arithmetic unit are time-shared by the IOC in the execution of its functions.

MEMORY

The memory section for the L-304 computer consists of basic modules of 8192 words of 32 bits each, organized for coincident-current operation, as shown in Figure F-2. Access time for the 8192-word memory is 800 nanoseconds and the total read/write cycle time is 2.2 microseconds. In addition to the use of wide-temperature, low-noise elements in the stack, current compensation is used in the drive circuits to provide stable operation over the full temperature range of operation.

The memory drawer is in two sections. The stack section contains the core stack, steering diode matrix, and sense amplifiers and the logic and driver section has the memory logic, current drivers, and switches.

The arithmetic and control module and the memory modules are designed so that the total memory for any system can be directly expanded to 131,072 words. Adding memory modules is accomplished by increasing chassis height and adding the necessary signal harness. The address register and the output register for the memory are located in the memory drawer to allow for memory extension and multiple computer operation. Up to two computers or computer-like devices can communicate with each memory module via two independent buses.

POWER SUPPLY

Power is provided by a number (as required by the system size) of microelectronic power supplies. Each module is six cubic inches in volume and supplies regulated voltage up to 8 amperes of current. These units were developed under a Navy contract and operate from either 400-cycle, 115-vac power or from +28 vdc prime power as specified in MIL-STD-704. The power supply efficiency is 70 percent when operating from an ac source and 80 percent when operating from +28 vdc. Each power supply is self-contained and provides full protection against prime power voltage variation. In addition, overvoltage and overcurrent protection of the power supply output is provided. Any fault condition automatically shuts off the power supply.

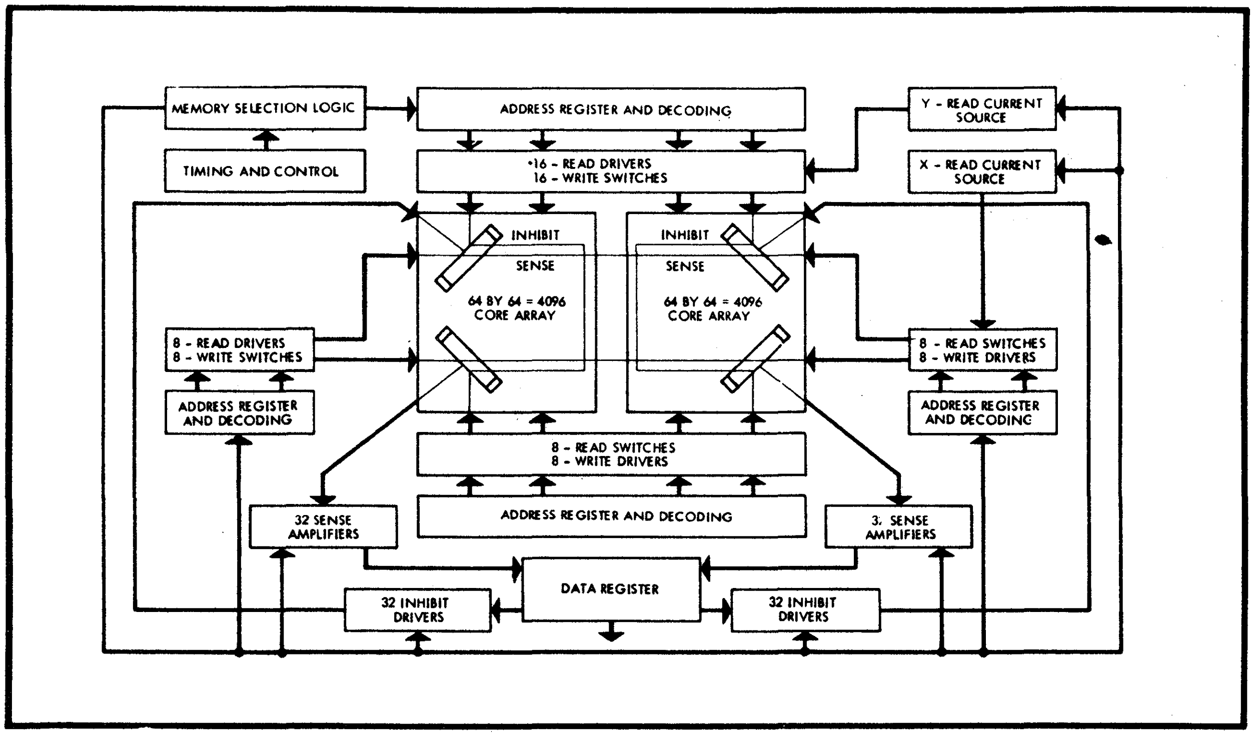


Figure F-2. The 8192-Word Memory Block Diagram

INPUT/OUTPUT SECTION

Extensive, versatile input/output (I/O) capabilities are provided in the L-304 which allow communication with up to 64 I/O devices. A block diagram showing the input/output interface is contained in Figure F-3. Once a transfer of data is initiated, transmission between the computer memory and the I/O device is accomplished independently of the current program.

Information Transfer

In the L-304, simultaneous information communication with up to 64 I/O devices is permissible on a multiplexed or time-shared basis. This type of communication is achieved by an automatic multiplexing feature in which peripheral device requests for information transfers are scanned within the computer and serviced on an assigned priority basis. The computer servicing of a scanned request requires an automatic program interruption of two memory cycles in duration. The program is permitted to resume execution after the I/O servicing is completed provided no I/O device service requests are present at the time. In this type of operation, the maximum transfer rate is 227, 000 words (32 bits) or characters (8 bits) per second.

A second type of I/O operation occurs when the program has been interrupted to service a device that requires continual servicing for a given period. This operation is defined as the burst mode. The computer continually services the device request for information transfer as long as the request is present and remains the highest priority. This type of operation results in a maximum transfer rate of 435,000 words or characters per second.

Input/Output System Organization

The L-304 input/output system consists of three parts: the input/output control (IOC) section of the L-304 processor, the input/output interface (I/O interface), and interface units with their associated input/output devices (I/O devices).

The L-304 processor has provision for communicating with up to 64 input/output devices under program control. The I/O devices must be connected to the I/O interface via an interface unit that will respond to the control and meet the requirements of the IOC. The design of each interface unit is special-purpose to meet the system control requirements and characteristics of the I/O devices it will control. These units may be designed to control one or several devices on a time-shared basis.

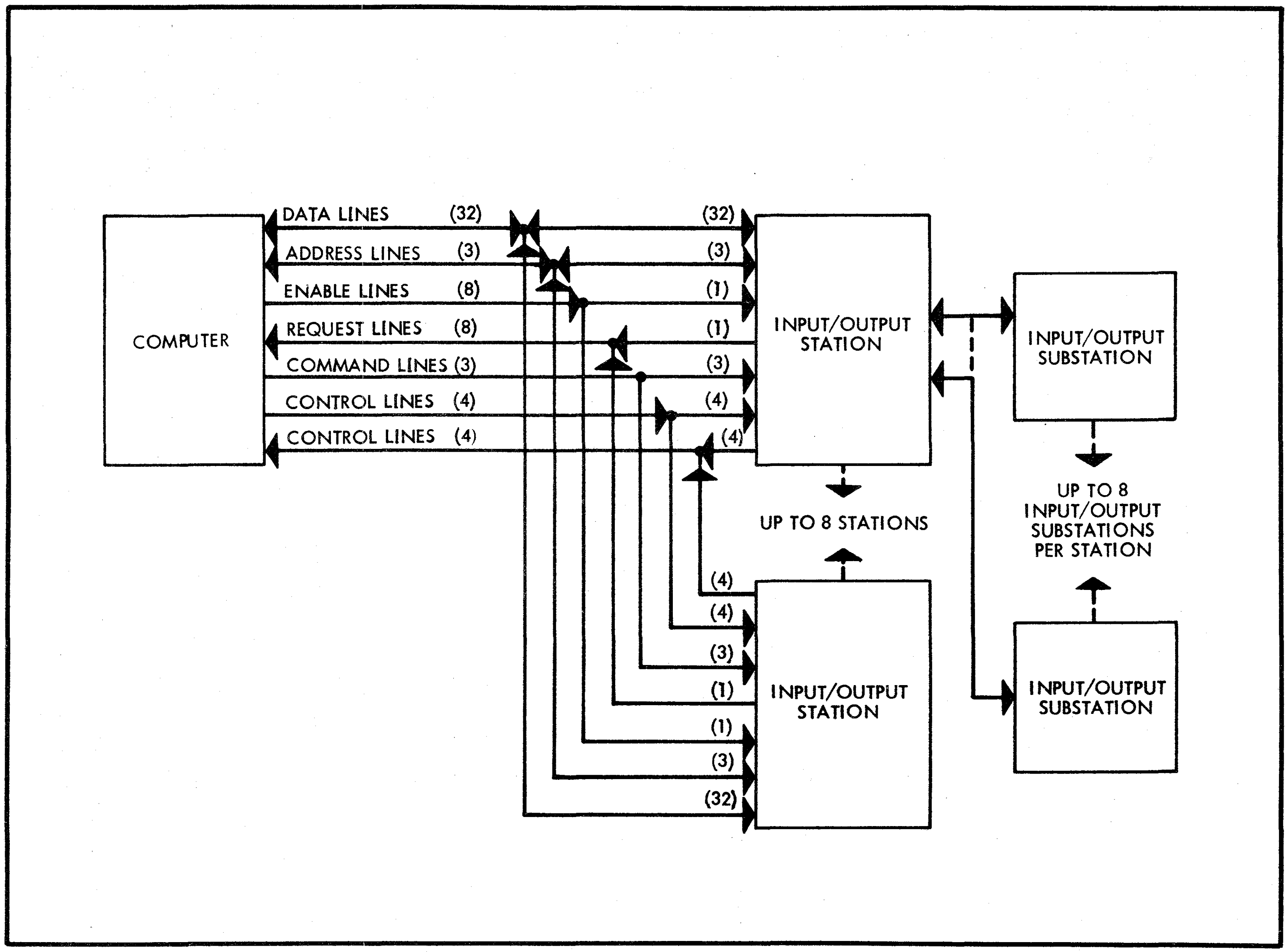


Figure F-3. L-304 Computer Input/Output Organization

The processor, under control of the IOC, is designed to directly communicate with up to eight terminals. The terminals are designated as I/O stations. Each I/O station may in turn allow communication with eight other terminals which are designated I/O substations. An interface unit may be connected at a station or substation depending upon the special design of the unit which is tailored to the system requirements. Addressing or recognition of I/O stations is performed by the IOC, whereas addressing or identification of the I/O substations is performed by the interface units via the three address lines of the I/O interface.

Communication between the computer input/output section and each channel is handled via three subchannels: (1) a data subchannel, (2) a control subchannel, and (3) a request subchannel. Data and command information occurs on the two-way data subchannel connecting the computer input/output section to each of the eight possible I/O stations. A common control subchannel between the input/output control and the I/O stations provides lines for: station code select, enable, indicator, and stop. Input/output stations request operations via a request subchannel to the IOC on separate request lines.

I/O Control Words

Data transfers are controlled by words stored in the computer memory. Two 32-bit words (a key word and a terminate word) are set up by the program for each I/O substation (up to a total of 64 devices). Each set of control words contains the following information: (1) the starting address of the transfer data; (2) the number of words (32-bit format) or characters (8-bit format) to be transmitted; (3) the mode of transmission: (4) a set of termination bits, which indicate the reason for the termination; (5) the number of the program level to be activated if a normal termination occurs; and (6) the number of the program level to be activated if an abnormal termination occurs. Figure F-3 contains these I/O control word formats.

COMPUTER ORGANIZATION

A significant feature of the L-304 is the availability of 64 program interrupt levels. Each level may contain an independent program or an integrated portion of a larger program. Each level has eight process registers and time-shares the arithmetic section and memory output register with the other 63 levels. Each level has an assigned priority, but only operates if the level is the highest priority "active" level. An interrupt level may be designated "active" either by instruction or by interrupts from the input/output section. Interrupts may thus be processed immediately, without program provision for temporary storage of the process register contents.

Control over which programs are running or waiting to run is maintained in the activity and mask registers. The activity register contains 64 bits, where each position represents a program level. The mask register is also 64 bits in length and may be used to inhibit a program from running. Thus, the program running at any given time is the highest priority program with coincident one bits in both the activity and mask registers. When a given program is running, the set of process registers accessed is that which corresponds to the active program level.

Special Registers

There are eight addressable 16-bit process registers for each of the 64 program interrupt levels (512 total). These registers are contained in Locations 0000-0777_8 of the core memory. Figure F-4 is a graphic representation of the location occupied by each level and the registers within each level. The eight process registers are multipurpose registers which may be used as accumulators, or as index registers. Each instruction will designate the registers it will use.

The Program Location (PL) register is used to contain the interrupt start address for its program level. That is, when processing is transferred to a given program level, the PL register designates the memory address of the first instruction to be executed. The memory extension register is used in conjunction with this address to get the needed memory module. It contains the numbers of those memory modules to be used during execution of a program assigned to a given program level. (See Figure F-4.)

Data Word Formats

Although the L-304 memory word consists of 32 bits, the computer treats this word as two consecutive 16-bit locations in memory; the first, even-numbered, the second, odd-numbered.

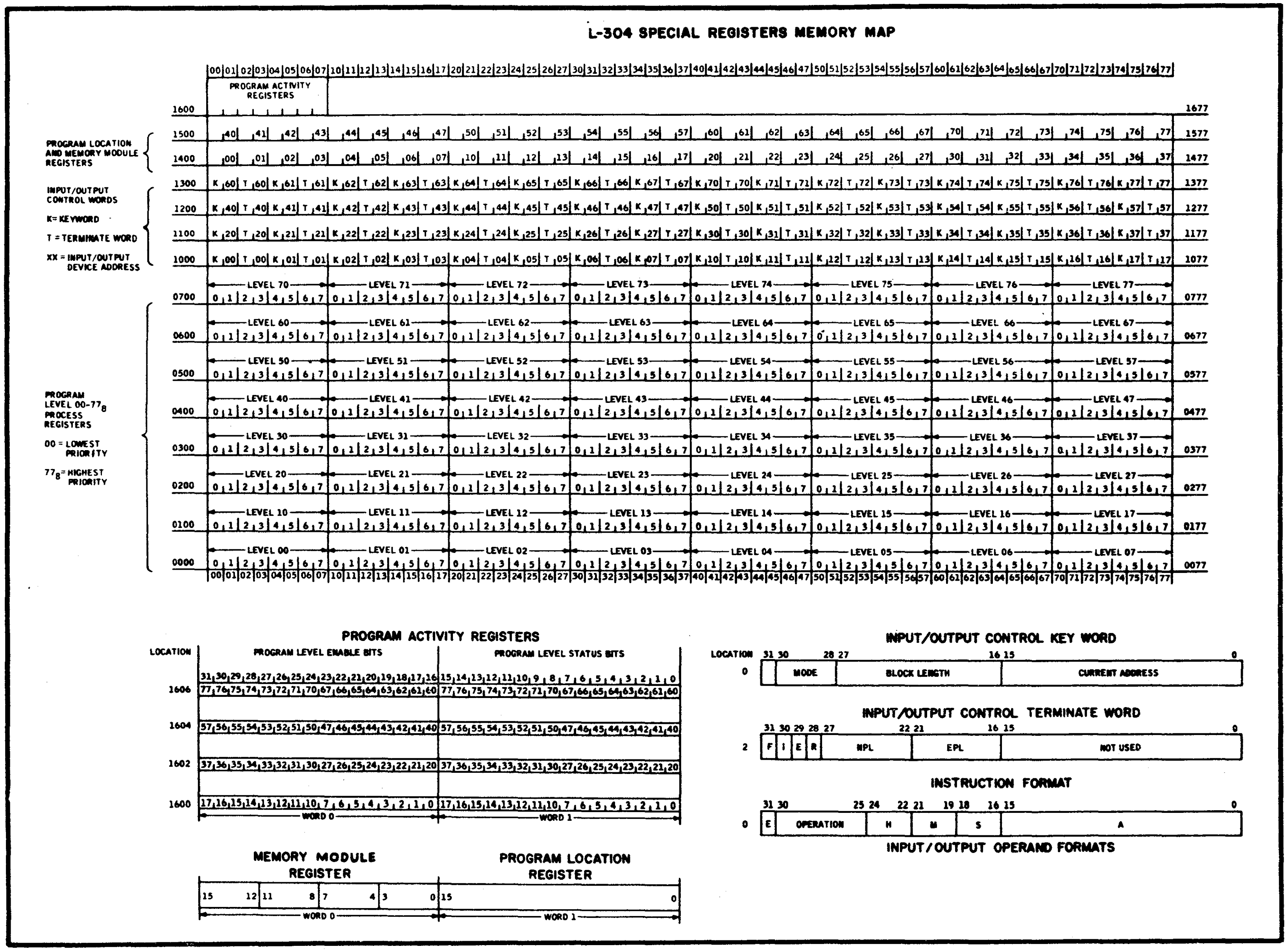

L-304 SPECIAL REGISTERS MEMORY MAP
PROGRAM ACTIVITY REGISTERS
PROGRAM LOCATION AND MEMORY MODULE REGISTERS
INPUT/OUTPUT CONTROL WORDS
K = KEYWORD
T = TERMINATE WORD
XX = INPUT/OUTPUT DEVICE ADDRESS
PROGRAM LEVEL 00-77₈ PROCESS REGISTERS
00 = LOWEST PRIORITY
77₈ = HIGHEST PRIORITY
1600
1500
1400
1300
1200
1100
1000
0700
0600
0500
0400
0300
0200
0100
0000
1677
1577
1477
1377
1277
1177
1077
0777
0677
0577
0477
0377
0277
0177
0077
PROGRAM ACTIVITY REGISTERS
LOCATION
PROGRAM LEVEL ENABLE BITS
PROGRAM LEVEL STATUS BITS
1606
1604
1602
1600
WORD 0
WORD 1
MEMORY MODULE REGISTER
PROGRAM LOCATION REGISTER
INPUT/OUTPUT CONTROL KEY WORD
MODE
BLOCK LENGTH
CURRENT ADDRESS
INPUT/OUTPUT CONTROL TERMINATE WORD
F
I
E
R
NPL
EPL
NOT USED
INSTRUCTION FORMAT
E
OPERATION
H
M
S
A
INPUT/OUTPUT OPERAND FORMATS

Figure F-4. L-304 Special Registers Memory Map

The data word normally contains 16 bits. These bits are numbered from 0 (least significant) through 15, beginning from the right, as shown in Figure F-5. The number system is binary, 2's complement. A number in the normal data word consists of 15 bits and a sign. For convenience, it can be stated as five octal digits and a sign. A binary 0 indicates a positive number and a binary 1 denotes a negative number (in 2's complement form).

Some instructions involve double-length operands, i. e., 32 bits of data, including sign. In these instances, registers or memory words are considered as pairs with the most significant 16 bits of the pair always the even-numbered, and the least significant always the odd-numbered register or memory word. The sign of a double-length number appears as the most significant bit in both registers or words of the pair.

INSTRUCTIONS

The L-304 computer utilizes a single address instruction that provides several operand address options. A total of 65 unique instructions are provided. These instructions include all of the normal general-purpose commands along with some very useful and powerful commands that simplify programming and save execution time and memory space. Table F-II is a list of the L-304 instructions.

INSTRUCTION WORD FORMAT

The L-304 computer uses a 32-bit instruction word. The format of the instruction word is shown in Figure F-6.

E Field

This single bit is used for instruction operation options. On those instructions which could cause an arithmetic overflow, it provides the programmer the option to skip the next instruction in sequence if an overflow does not occur. If the E bit is a zero, the next instruction in sequence is executed and overflow is ignored.

On five special instructions, the E bit is used to modify the operation called for by the operation field, F, in a related manner. This is described in the description of each of these instructions.

F Field

This six-bit field is the instruction operation code. In subsequent descriptions this code is represented by a two digit octal number.

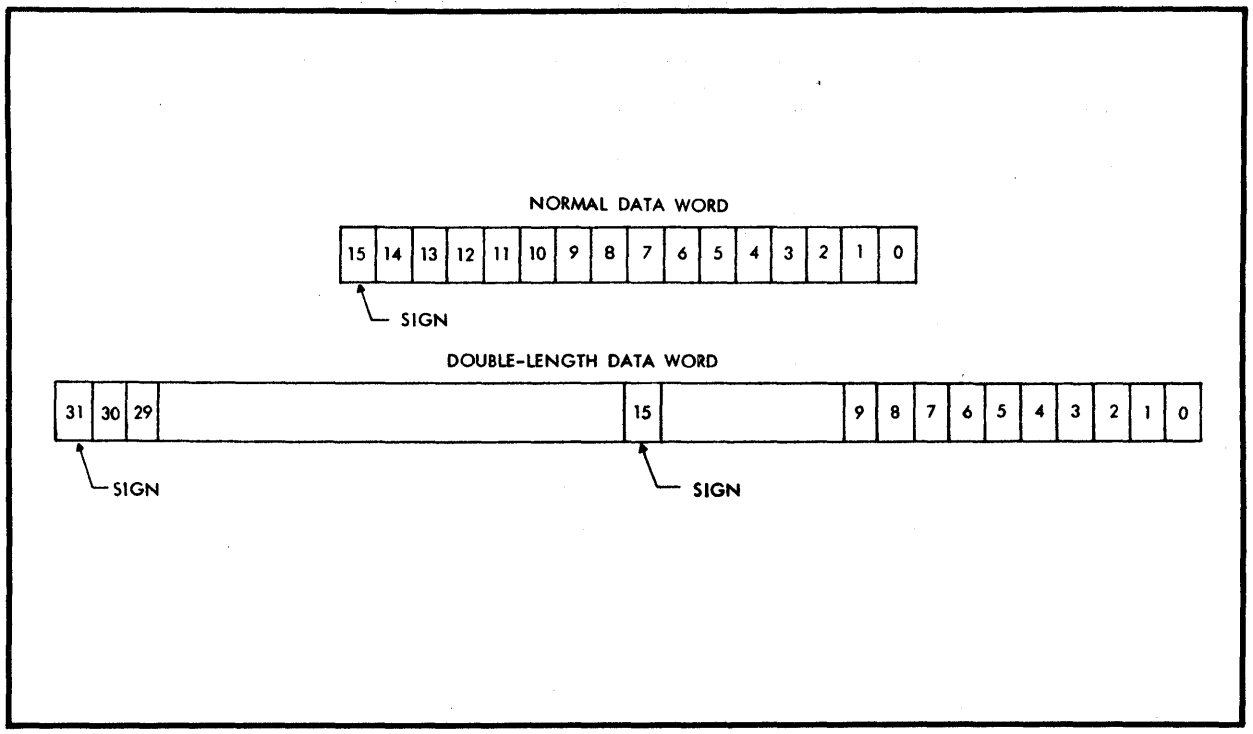


Figure F-5. Data Word Formats

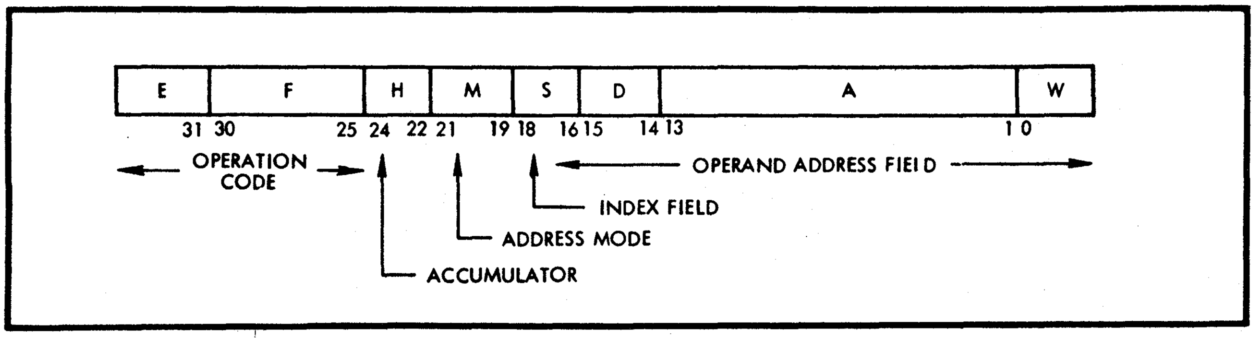


Figure F-6. L-304 Instruction-Word Format

H Field

The H field is a three-bit binary number that selects one of eight process registers which is to be used as the accumulator by the instruction. Process registers are addressed by H = 0, 1, 2, . . . , 7 on all program levels.

M Field

The M field is a three-bit code that provides up to eight instruction address options as follows:

M = 0, Direct Address

M = 1, Direct Address with indexing

M = 2, Literal

M = 3, Literal with indexing

M = 4, Indirect

M = 5, Indexed, Indirect

M = 6, Indirect, Indexed

M = 7, Relative

S Field

The S field is a three-bit field that selects one of the eight process registers to be used as an index register on Modes M = 1, 3, 5 and 6 and one of 7 process registers if M = 7 (S = 0, M = 7 implies no indexing). The S field addresses the same set of process registers as the H field on a given program level. A different set of eight process registers is provided for each of the 64 program levels.

FUNCTION CODE	MNEMONIC CODE	INSTRUCTION	SYMBOLIC OPERATION	EXECUTION TIME (μSEC)†
A. DATA TRANSMISSION INSTRUCTIONS				
1. 04	LDR	LOAD RH	$Y = (A) \rightarrow RH$	6.6
2. 05	STR	STORE RH	$Y = (RH) \rightarrow (A)$	6.6
3. 06	LDD	LOAD DOUBLE	$Y = (A, A+1) \rightarrow RH$	6.4
4. 17	LDC	LOAD TWO'S COMPLEMENT	$(Y)_{\bar{2}} \rightarrow RH$	7.2
5. 16	LDA	LOAD ABSOLUTE	$\lvert (Y) \rvert \rightarrow RH$	7.2
6. 07	STD	STORE DOUBLE	$Y = (RH, RH+1) \rightarrow (A, A+1)$	6.4
7. 71	MVI	MOVE AND INSERT	$(F_{RH}) \rightarrow (F_{RS})$	7.4 + 0.8 n*
8. 70	MVZ	MOVE AND ZERO	$(F_{RH}) \rightarrow (F_{RS}),\ 0 \rightarrow (\bar{F}_{RS})$	6.6 + 0.8 n*
9. 02	EXC	EXCHANGE	$(A) \rightarrow RH,\ (RH) \rightarrow (A)$	8.8
10. 03	EXD	EXCHANGE DOUBLE	$\left\{\begin{array}{l}(A+1, A) \rightarrow RH+1, RH; \\ (RH+1, RH) \rightarrow (A+1, A)\end{array}\right.$	9.4
B. ARITHMETIC INSTRUCTIONS				
11. 10	ADD	ADD	$Y = (A) + RH \rightarrow RH$	7.2
12. 11	SUB	SUBTRACT	$(RH) - Y \rightarrow RH$	7.2
13. 12	RAD	REPLACE-ADD	$Y = (A) + (RH) \rightarrow (A)$	7.4
14. 13	RUB	REPLACE SUBTRACT	$Y = (A) - (RH) \rightarrow (A)$	7.4
15. 14	ADA	ADD ABSOLUTE	$(RH) + \lvert Y \rvert \rightarrow RH$	7.2
16. 15	SBA	SUBTRACT ABSOLUTE	$(RH) - \lvert Y \rvert \rightarrow RH$	7.2
17. 30	MPY	MULTIPLY	$(RH) \times Y \rightarrow RH, RH+1$	18.8 + 0.8 n**
18. 31	DIV	DIVIDE	$(RH, RH+1) \div Y \rightarrow RH+1;$ REMAINDER $\rightarrow RH$	33.2
C. LOGIC INSTRUCTIONS				
19. 20	EOR	EXCLUSIVE OR	$(RH) \veebar Y \rightarrow RH$	7.2
20. 21	IOR	INCLUSIVE OR	$(RH) \vee Y \rightarrow RH$	7.2
21. 22	AND	LOGICAL AND	$(RH) \wedge Y \rightarrow RH$	7.2
22. 24	RER	REPLACE EXCLUSIVE OR	$(A) = Y \veebar (RH) \rightarrow (A)$	7.4
23. 25	RIR	REPLACE INCLUSIVE OR	$(A) = Y \vee (RH) \rightarrow (A)$	7.4
24. 26	RAN	REPLACE LOGICAL AND	$(A) = Y \wedge (RH) \rightarrow (A)$	7.4
D. SHIFT INSTRUCTIONS				
25. 44	SLL	SHIFT LONG LEFT		7.6 + 0.8***
26. 57	SRA	LONG SHIFT RIGHT, ALGEBRAIC		7.6 + 0.8***
27. 56	SRL	SHIFT LONG RIGHT, LOGICAL		7.6 + 0.8***
28. 45	NLL	NORMALIZE LONG LEFT		10.6 + 0.8***
29. 46	SNC	SHIFT AND COUNT		8.4 + 0.8***
30. 47	RFT	REFLECT		7.6 + 0.8***

*n = NUMBER OF SHIFTS REQUIRED TO ALIGN THE FIELD

**n = NUMBER OF "1" BITS IN MULTIPLIER

***n = NUMBER OF PLACES SHIFTED

Table F-II. Instruction Repertoire (Sheet 1 of 2)

FUNCTION CODE	MNEMONIC CODE	INSTRUCTION	SYMBOLIC OPERATION	EXECUTIVE TIME (SEC)[†]
E. TRANSFER INSTRUCTIONS				
31. 32	ITX (E BIT - 1)	MODIFY RH BY TWO; TRANSFER		7.2
	DTX (E BIT - 0)	IF RH $\neq$ 0		
32. 33	IOX (E BIT - 1)	MODIFY RH BY ONE; TRANSFER		7.2
	DOX (E BIT - 0)	IF RH $\neq$ 0		
33. 34	XFR	TRANSFER UNCONDITIONAL		4.4
34. 35	XLK	TRANSFER UNCOND. AND STORE LINK		6.4
35. 36	XSW	TRANSFER ON CONSOLE X SWITCH		4.4
36. 40	XEZ	TRANSFER IF RH = 0		6.4
37. 41	VNZ	TRANSFER IF RH $\neq$ 0		6.4
38. 42	XNG	TRANSFER IF RH IS NEGATIVE		6.4
39. 43	XPS	TRANSFER IF RH IS POSITIVE		6.4
F. JUMP INSTRUCTIONS				
40. 37	JTW	JUMP THREE WAY		7.4
41. 50	CJL	COMPARE, JUMP IF LESS		7.2
42. 51	CJE	COMPARE, JUMP IF EQUAL		7.2
43. 52	CJU	COMPARE, JUMP IF UNEQUAL		7.2
44. 53	CJG	COMPARE, JUMP IF GREATER		7.2
45. 54	GCI	GATED COMPARISON, JUMP IF INSIDE		9.4
46. 55	GCO	GATED COMPARISON, JUMP IF OUTSIDE		9.4
47. 64	TLZ	TEST LOWER BIT, JUMP IF 0		5.2
48. 65	TUZ	TEST UPPER BIT, JUMP IF 0		5.2
49. 66	TLF	TEST LOWER BIT, JUMP IF 1		5.2
50. 67	TUF	TEST UPPER BIT, JUMP IF 1		5.2
G. MISCELLANEOUS INSTRUCTIONS				
51. 77	NOP	NO OPERATION		2.2
52. 01	EXE	EXECUTE		2.2
53. 60	SBL	SET LOWER BIT		5.2
54. 61	SBU	SET UPPER BIT		5.2
55. 62	RBL	RESET LOWER BIT		5.2
56. 63	RBU	RESET UPPER BIT		5.2
57. 72	STZ	STORE ALL ZEROS		5.2
58. 00	HLT	HALT		2.2
H. INPUT/OUTPUT INSTRUCTIONS				
59. 27	MBA	MEMORY BANK ASSIGNMENT		4.4
60. 75	FIP	INPUT TO REGISTER		7.2
61. 76	FOP	OUTPUT FROM REGISTER		7.2
62. 74	EDC	EXTERNAL DEVICE COMMAND		7.6
63. 23	MBD	MEMORY BANK DESIGNATOR		4.4

[†]ASSUMING NO MEMORY OVERLAP. IF MEMORY OVERLAP OCCURS, SUBTRACT 0.6 MICROSECOND FROM THESE TIMES.

Table F-II. Instruction Repertoire (Sheet 2 of 2)

D Field

The D field is a two-bit field that selects one of four registers whose content is used for address extension. The selected register contains four binary bits which are appended to the most significant end of the instruction's A field. This effectively yields a 17-bit operand address 131,072 32-bit words to be addressed. The set of four four-bit registers is called the Memory Extension Register. A different memory extension register is provided for each of the 64 program levels.

A Field

The A field is a 13-bit address field. The 13-bit address will select one of 8192 32-bit words within a memory module. A memory module is selected by the four-bit field of the Memory Extension Register.

On some instructions the A field (with the D and W fields) is used as a logic operation (mask) or it is used to specify the number of bits to be shifted.

W Field

This one-bit field specifies left or right half of the 32-bit operand which will be used as a 16-bit operand. If W is a one the right half of the word is used. If W is a zero, the left half is used. The A and W fields together represent a 14-bit half-word address.

HARDWARE CHARACTERISTICS

The overall design of the L-304 computer takes full advantage of the latest microintegrated-circuit technology, multilayer laminate interconnection techniques, and unique packaging and heat removal techniques. The circuit characteristics unique to monolithic integrated circuits and the circuit characteristics unique to the Litton multilayer board construction are utilized to improve the characteristics of the final product, the L-304. Figure F-7 represents a typical multilayer board assembly. Continuous heat conducting metal cooling paths are built into the multilayer boards to achieve the goal of low temperature rise of components. Similarly, low-impedance power transmission lines (0. 1 ohm dc to 100 megacycles) and ground planes are integrated into the multilayer boards along with electrostatic shielding techniques to achieve minimum noise while providing a source of high-frequency current transients necessary for high-speed operation without resorting to "brute force" discrete local filtering.

Circuitry

The circuitry used in the L-304 consists of semiconductor integrated circuits, in the form of NAND (an inverted logical AND) gates, interconnected by multilayer laminated boards. The Litton integrated NAND circuits (LINCs) are of two types: an eight-input NAND gate, and a dual, four-input NAND gate. Both LINCs measure 0.250 by 0. 175 by 0. 065 inch, excluding leads, and 0. 325 by 0. 375 by 0. 065 inch including leads.

Significant features of a LINC include:

(1) A high degree of noise immunity is provided when compared with other microelectronic circuits. This is especially important when many LINCs are used together in a system the size of an L-304 computer.

(2) A large amount of gating current is available as needed at the output of each LINC for either direction of logic swing, thereby permitting very-high-speed operation.

(3) Standby power between switching operations is quite low, since, in the high output-voltage state, the LINC is "looking" into a back-biased emitter; and in the low output-voltage state, the load current is limited to the base current of the multiemitter input transistor of the following LINC.

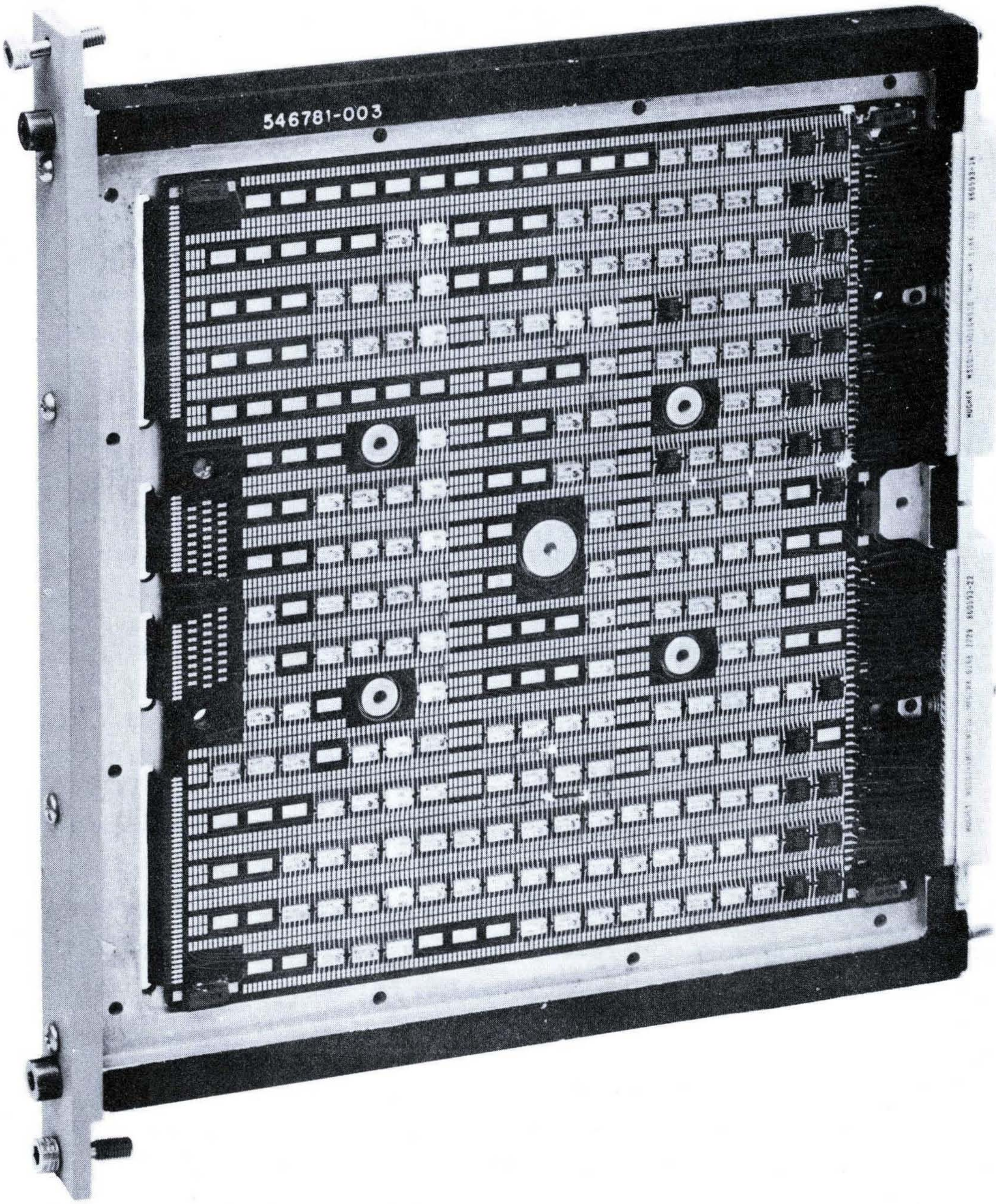


Figure F-7. A & C Drawer Module Assembly

The micromemory is optimally designed around a wide-temperature coincident-current, 22-mil outer diameter lithium-ferrite core stack to provide minimum size and complexity and to improve reliability. The drivers use thin-film substrates, with remaining memory circuits being monolithic integrated semiconductor circuits. To increase operating margins even beyond those available with wide-temperature ferrite the stack temperature is sensed, and the power supply for the current drivers is varied linearly with temperature. The temperature tracking of the power supply has the advantage of minimizing power dissipation at elevated temperatures.

Packaging

The L-304 computer reflects the extensive developmental effort which Litton has conducted in the field of microelectronic packaging and interconnection techniques. The design meets the requirements of MIL-E-5400G, Class II equipment. Some of the prime considerations governing the packaging of the computer electronics include those outlined below.

(1) Compliance with the environmental specifications

(2) Accessibility and ease of maintenance

(3) Relative positioning of electronic modules in order to minimize interconnection complexity and to achieve maximum thermal efficiency

(4) Low weight and volume in relation to total computing power

(5) Ease of manufacture

(6) Lowest cost consistent with expected performance

Automated fabrication techniques are used to significantly reduce construction cost, and system adaptability and expandability are easily achieved. The L-304 computer is designed to accommodate varying quantities and combinations of modules as system requirements change. Increased memory capacity or conversion from a single to a multicomputer is accomplished by adding modules.

Each of the modular subassemblies is designed internally for specific functional performance. Except for thickness, they appear similar externally (see Figure F-8). The thicknesses are approximately 0. 7 inch for integrated circuit drawer modules, 1. 5 inches for power supply drawer modules, and 3. 25 inches for the memory module. The computer frame housing is of riveted sheet metal construction, reinforced with wrought extrusion in areas of high stress. The top, bottom, and side walls (the entry and exit plenums) are rigid box sections on four peripheral sides. These are further stiffened by perpendicular cross bracing which supports and guides the three decks of electronic drawers.

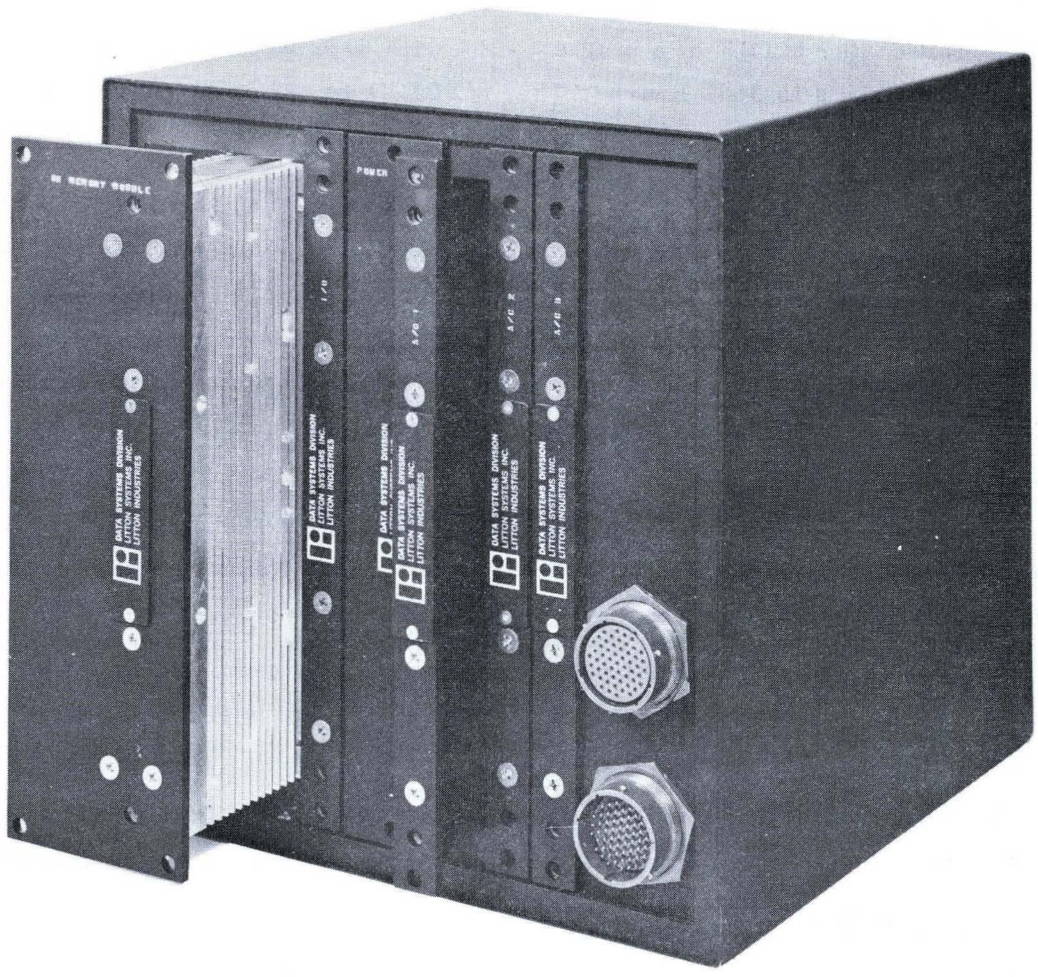


Figure F-8. Typical L-304 Configuration

A connector panel located at the rear of the drawers supports the receptacle connectors and harness for both internal and external communications. This panel is made an integral part of the structure. Harness and connectors form a removable assembly.

RELIABILITY

The predicted MTBF of a single L-304 with 8192 words of memory and associated power supplies is 3660 hours. This MTBF is based on an extremely conservative application of failure rate data. The assumptions and criteria which were employed may be summarized as follows:

(1) Applicable failure rates used were derived from recognized or acceptable military sources such as the MIL-HDBK-217, dated 8 August 1962; Bureau of Naval Weapons Failure Rate Data Handbook (FARADA), etc. However, where failure rates obtained from these sources were determined to be obsolete due to the current advancement of the state of the art on component parts, such failure rates were adjusted to consistent values.

(2) Other applicable failure rates that cannot be found directly from the sources in Item 1, were derived from the failure rates of similar items reported from the field and from reliable vendor reports.

(3) Failure rates for most parts are derived at a temperature of $60^{\circ}C$, which exceeds safely the expected temperature that the equipment will be subjected to in actual application.

(4) The prediction method is based on the assumption that a part failure rate is a reflection of system failure, which is typical of a series-connected system.

MAINTAINABILITY AND AVAILABILITY

At organizational level of maintenance, the predicted mean time to repair (MTTR) of the 8K memory computer is 0.162 hour or 9.72 minutes, resulting in an availability of 99.994 percent. The low maintenance task times are obtained with the aid of the computer's built-in maintainability design features. A standard 3/16 Allen wrench is all that is needed in getting access to the defective item to effect its replacement. No special tools are necessary in this level of maintenance. A test message entered into the computer's memory is, however, a prerequisite to the isolation of a fault.

APPENDIX G

CONVENIENCE OF L-304 MEMORY MODULE ACCESS

Understanding the L-304 process for accessing up to 16 8K memory modules is relatively simple assuming one has a previous acquaintance with the L-304 level concept. At any instant, a maximum of four 8K memory modules may be active. This allows all programs to be assembled and executed between 0 and 32K thus requiring only a 16-bit instruction address field.

There are essentially two methods to manipulate which four of the 16 memories are active at any one time; one is by direct command, the other is by a level change. The "direct command" approach requires one command to change the memory configuration; the level change approach, once it is established, automatically provides a new (if desired) memory configuration with each change to a new level. When loading a program, the memory module designation must agree with its assembled location, i. e. , a program assembled between 8K and 16K must be loaded and executed within the 8K and 16K memory slot (see Figure G-1) regardless of module number designation. Once a memory module is loaded and properly designated, the contained program cannot be relocated without reloading (as is the case in any other computer). A simple analogy to illustrate the versatility of loading and executing a program is shown in Figure G-1.

Note that any combination of four can be set in the readout windows with the pseudo-thumbwheel selection shown in Figure G-1. (In actuality, these selections are made by the operator at load time via a control routine.) However, no two may have the same number when loading programs (each module can only be loaded once for a given exercise). Assume a program were loaded with the "0, 2, 12, 5" MER setting shown in Figure G-1. This simply causes those particular 8K modules to be loaded consecutively with a program assembled between 0 and 32K. The modules would be referenced in terms of those indicated address locations during execution, therefore their memory slot assignment, when loading, must be consistent with the assignment when executing.*

*Areas reserved for data storage can deviate from this rule. This added versatility will not be discussed at this time.

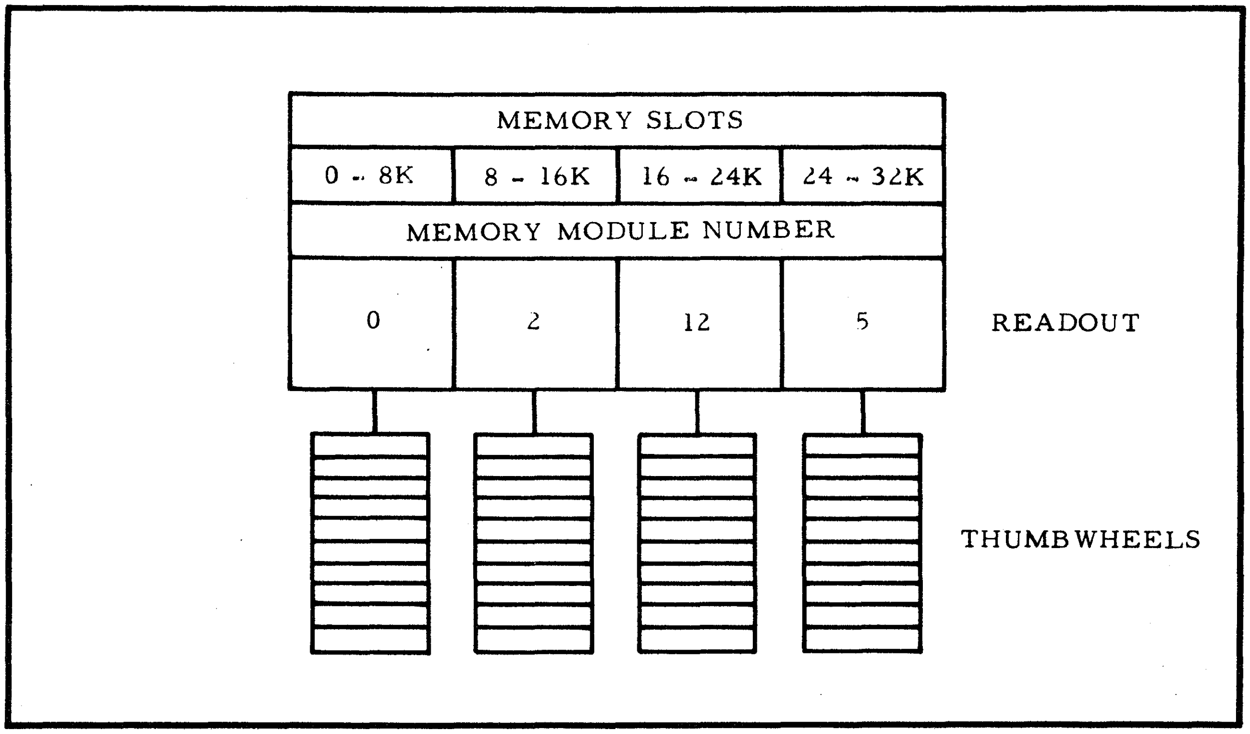
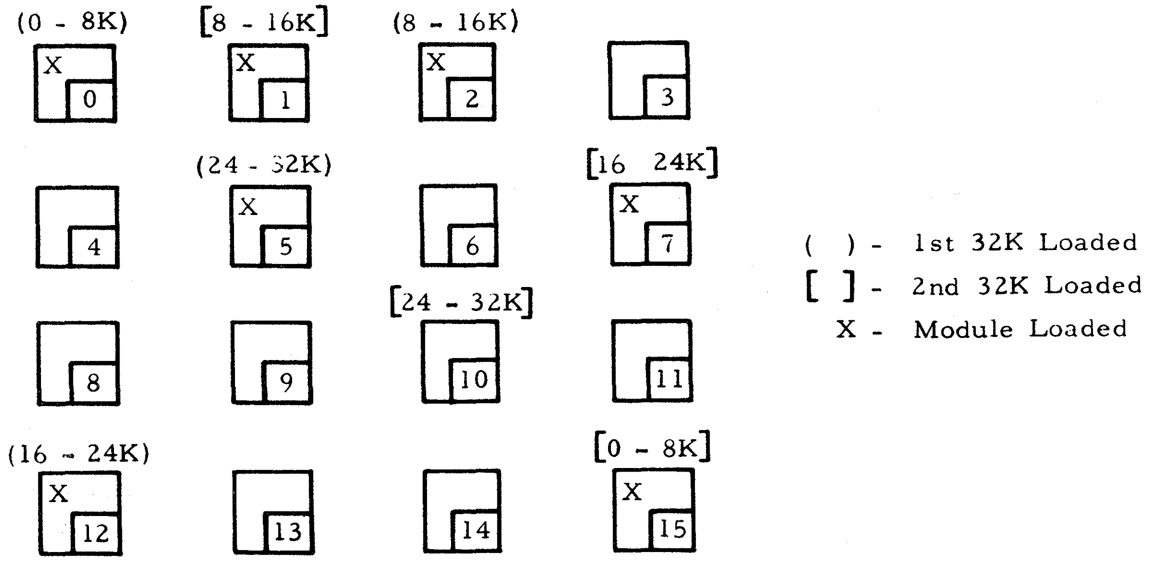


Figure G-1. Memory Extension Register (MER) Analogy

Assume one desired to load another 32K program (or another portion of the same program). The setting could be changed to say:

15	1	7	10

and cause these modules to be filled. Memory would now appear as follows:



If during execution one desired to communicate between any two programs, it would simply require resetting the numbers in the window to any combination needed. This is conveniently accomplished in an actual program by preassigning desired module combinations to program levels, then communicating by changing levels (by setting another level's status bit) which automatically activates the associated modules (each level has an assigned set of modules). If, for instance, the program in module number 15 desired to communicate with the program in module number 0, it could simply set a status bit for a level which included module 0 in its MER. The called program would be executed, it would clear its own status bit causing control to return to the calling program.

In a dual processor configuration the module assignments for each level in one processor likewise have no effect on the other processor's selections.

As can be determined by this discussion, the modes of versatility with respect to module selection and configuration control are quite adequate to provide any communications necessary to execute a program, regardless of size, within the 16-module configuration.

1026 64660

DATA SYSTEMS DIVISION, LITTON SYSTEMS, INC., 8000 WOODLEY AVENUE, VAN NUYS, CALIF.