1. CONTENTS

2. INTRODUCTION

The UNIVAC® 1108 System – the logical, program compatible successor to the UNIVAC 1107 – is an integration of system-oriented hardware design, imaginative development and programming technology. The result is a system that effectively knows no application boundaries. It is equally at home in real-time, scientific or data processing environments, and capable of adjusting dynamically to any one or a mixture of these environments.

All system operations are coordinated and controlled by a versatile executive system having full real-time, multiprogramming and multiprocessing capabilities, but possessing the classic simplicity of a monitor system.

The following description will serve as an introduction to the UNIVAC 1108 Multi-Processor System organization, hardware components and programming system.

3. SYSTEMS DESCRIPTION

The 1108 System is a general purpose, high performance Unit and Multi-Processor system featuring the latest advances in computer design, systems organization and programming technology. Its modular structure enables a precise blend of systems components to fulfill the exact speed and capacity requirements for applications ranging from the most basic job shop system to the comprehensive Public Utility Computing complex. Among the principal features of the 1108 System are:

- Common resource systems organization
- Multiple 1108 Central Processors
- Multiple Input/Output Controllers
- Large, modular, parity checked high speed main storage
- Overlapped and interleaved main storage access
- Redundancy between systems components
- Program address relocation
- Storage protection
- Partial word addressability – 6, 12, 18, 36, 72 bit bytes
- High speed, random access auxiliary storage
- Equality among multiple Central Processors
- Privileged Mode For Executive System
- Guard Mode for User Programs

SYSTEMS COMPONENTS

The 1108 System is organized to allow multiple processors to simultaneously perform many different tasks under the direction of a common Executive control system. A multi-processor system requires much more modularity in organization than does a unit processor system. It must be broken down into individual logical components where:

- *Each system component must have more than one access path*
- *Priority logic is required to resolve possible access conflicts*
- *The failure of any individual component must not prohibit continued operation of the system*
- *System components must be logically removable for servicing without disabling the system*

The 1108 System is constructed of six individual component classifications:

- 1108 CENTRAL PROCESSORS
- INPUT/OUTPUT CONTROLLERS
- MAIN STORAGE MODULES
- AUXILIARY STORAGE SUBSYSTEMS
- SYSTEM INTERCONNECTION COMPONENTS
- PERIPHERAL SUBSYSTEMS

UNIVAC 1108 Central Processors

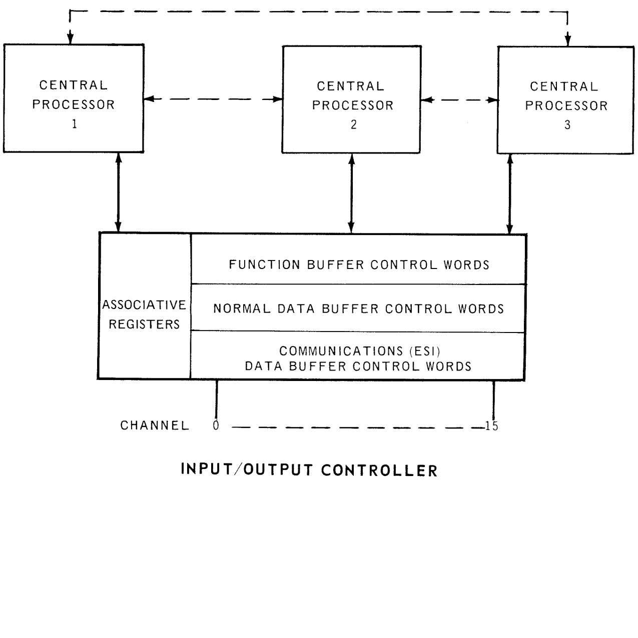
Each processor is equipped with all functions for the execution of instructions including arithmetic, input/output, and Executive Control. In a multiprocessor configuration, all processors have equality – the test of a true multi-processor system. Included in each processor is its own set of 125 nanosecond integrated circuit control registers providing multiple accumulators, index registers, input/output access control registers, and special use registers.

Input/Output Controller

The 1108 Input/Output Controller is an independent wired logic processor utilized in multi-processor systems providing:

- Up to sixteen high speed (1,500 KC) data channels
- Independent access to main storage
- Data chaining
- Contingency checking
- Sixteen Associative Registers
- Sixteen External Function Buffer Control Words
- Forty-eight Normal Input/Output Buffer Control Words
- 192 Communications Buffer Control Words
- An optional 256 additional Communications Buffer Control Words

I/O Controller Schematic


CENTRAL PROCESSOR 1
CENTRAL PROCESSOR 2
CENTRAL PROCESSOR 3
ASSOCIATIVE REGISTERS
FUNCTION BUFFER CONTROL WORDS
NORMAL DATA BUFFER CONTROL WORDS
COMMUNICATIONS (ESI) DATA BUFFER CONTROL WORDS
CHANNEL 0
15

INPUT/OUTPUT CONTROLLER

Main Storage Modules

The 1108 System provides a main storage expandable in 65,536 word increments up to a maximum of 262,144 thirty-six bit words. The main storage read/restore cycle time is 750 nanoseconds. Up to four logical banks for instruction/data reference overlapping provide an effective cycle time of 375 nanoseconds. In addition, up to eight way interleave of storage modules is provided to further reduce the probability of access conflicts.

Auxiliary Storage Subsystems

The Auxiliary Magnetic Drum Storage Subsystems are an integral part of each 1108 System. Up to eight FH-432 and/or FH-1782 magnetic drums may be attached to a single Control Unit/Synchronizer. Both drums transfer data at 1,440,000 characters per second, with the average access of the FH-432 at 4.3 milliseconds and the FH-1782 at 17 milliseconds.

System Interconnection Components

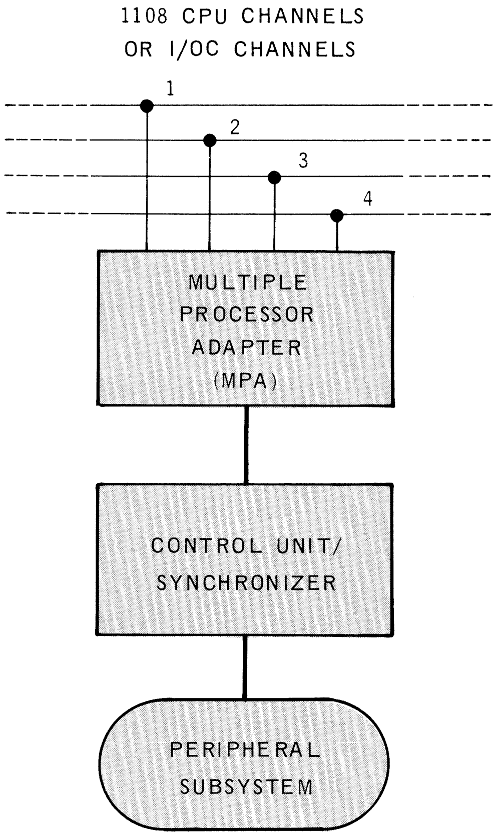
The design of a multi-processor system requires both established and reactive priorities between shared components of the system. The 1108 system provides shared storage and peripheral components through the Multiple Module Access Unit (MMA) and the Multiple Processor Adapter (MPA). Both the Multiple Module Access Unit and Multiple Processor Adapter are integral parts of multiprocessor configurations. The MPA may also be utilized to share peripheral subsystems between two independent 1108 Systems.

The *Multiple Processor Adapter* (MPA) enables up to 4 Processors or Input/Output Controller channels to access the units in a shared peripheral subsystem.

Access to shared peripheral subsystems is determined primarily by time of request. If two requests are made simultaneously, the Processor or Input/Output Controller on the lowest numbered MPA Input/Output channel position will receive priority. The Executive will be informed if a Processor or I/O is unable to access a subsystem because of a busy condition, and the Executive will automatically stack the request until the MPA channel is available.

Each Input/Output channel has a Function Word holding register in the MPA so that first level queuing is handled in the MPA itself. Longer queues are stacked by the Executive which keeps track of the number of function requests outstanding.

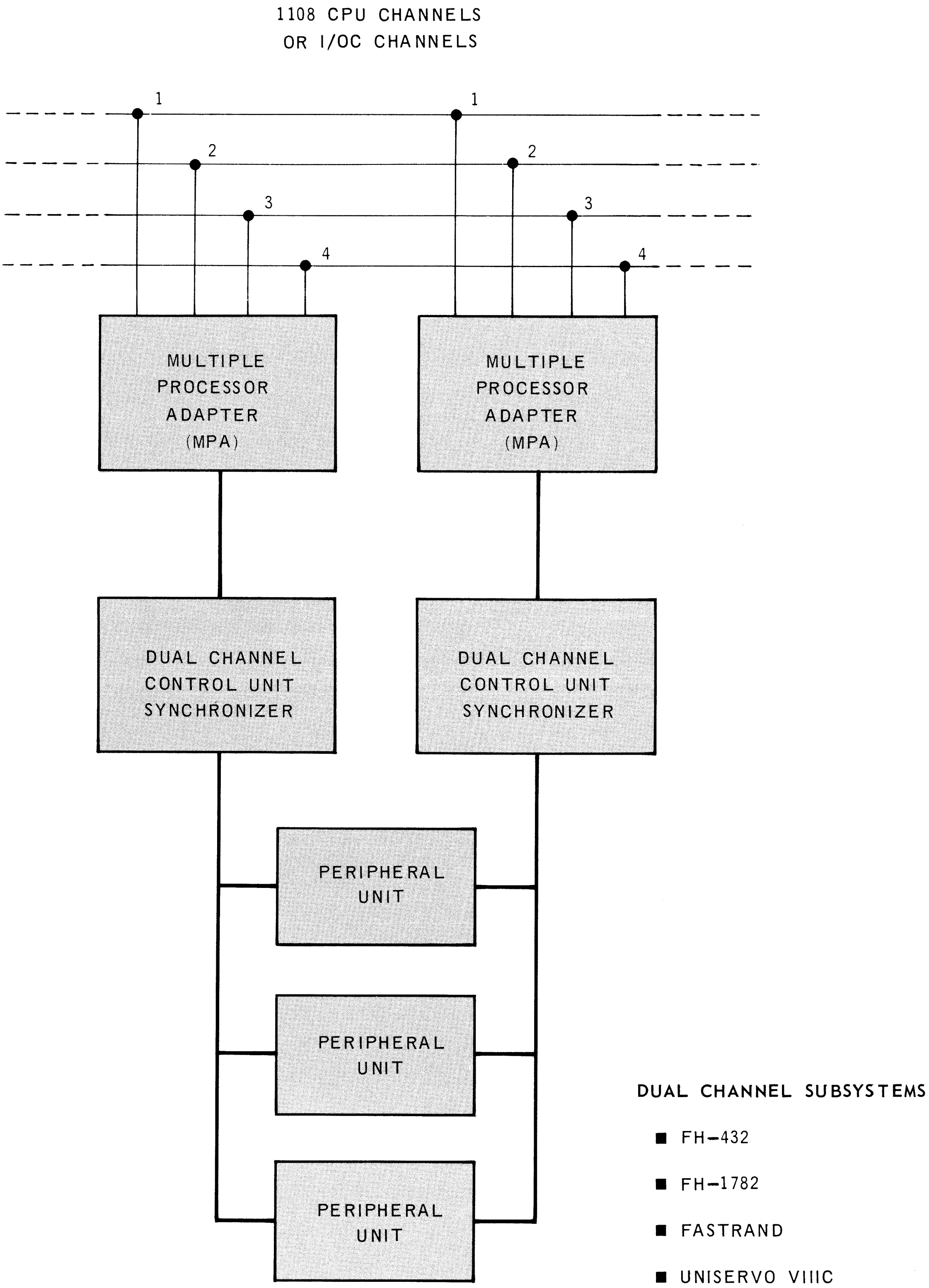
Processors are connected to a dual-path subsystem through dual MPA Units. Two Input/Output channels from each Processor or Input/Output Controller take separate paths. The failure of an MPA or one of the pair of Control Unit/Synchronizers will only affect one of the two paths to a peripheral subsystem, and all units will still be accessable via the second MPA and Control Unit/Synchronizer.



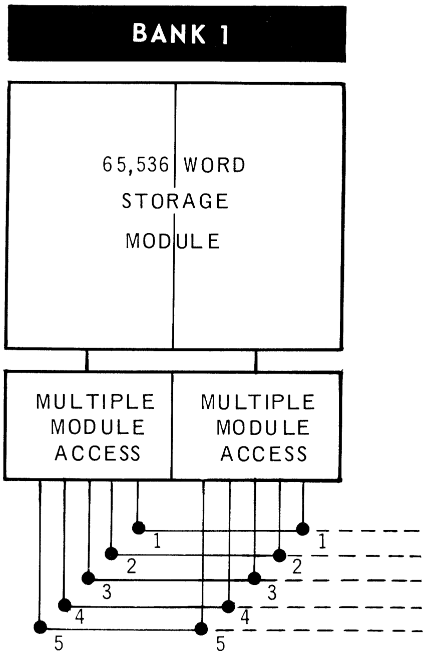
SINGLE CHANNEL SUBSYSTEMS

- FH–432
- FH–1782
- FASTRAND
- UNISERVO VIC
- UNISERVO VIIIC
- CARD READ/PUNCH
- PRINTERS
- PAPER TAPE
- STANDARD COMMUNICATIONS SUBSYSTEM
- WORD TERMINAL SYNCHRONOUS
- 1004

MULTIPLE PROCESSOR ADAPTER (MPA) – SINGLE CHANNEL SUBSYSTEM



MULTIPLE PROCESSOR ADAPTER – DUAL CHANNEL SUBSYSTEM



MULTIPLE MODULE ACCESS UNIT (MMA)

The *Multiple Module Access* Units (MMA) allow the sharing of individual storage modules by up to five Central Processors and/or Input/Output Controllers on a fixed priority basis.

The MMA recognizes the Storage Access Requests on a priority basis with the highest number processor interface retaining the highest priority. Input/Output Controllers which require highest priority will, therefore, be connected to the highest numbered interface. Upon recognition of a Storage Access Request the MMA will gate the address lines, central processor data lines, and write control signals to the storage module. The MMA will route the Storage Acknowledge back to the recognized processor and provide the drive necessary to display the storage data to the processor interfaces. The utilization of MMA's adds 125 nanoseconds to the basic storage cycle.

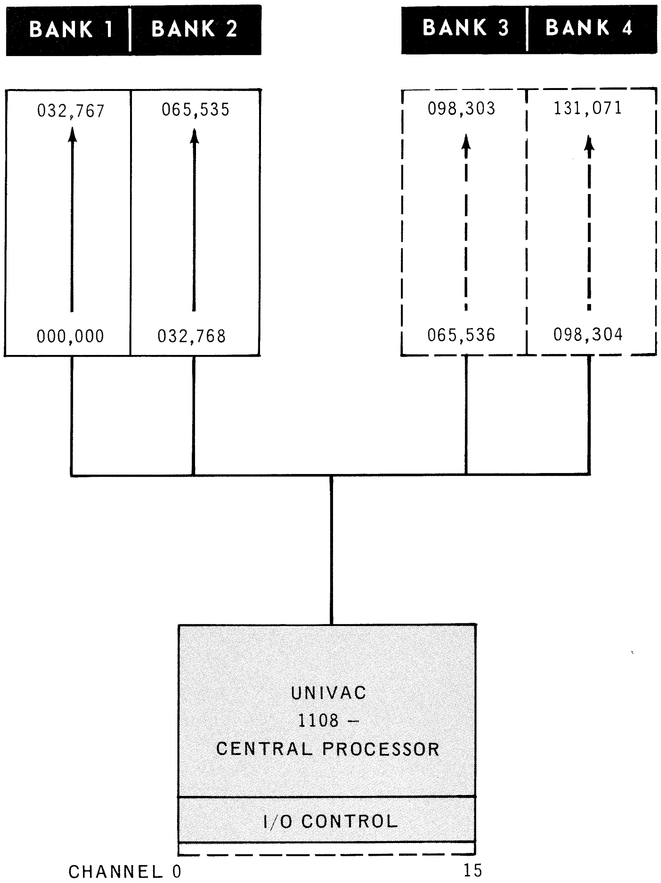
In the event of a storage module failure such as recurrent parity errors, individual storage modules may be taken off-line for maintenance purposes through the Availability Control Panel (ACP). This small system control panel allows individual storage modules to be made unavailable to the system while either Preventive Maintenance or Emergency Maintenance is performed on these individual components of the total 1108 system.

CONFIGURATIONS

The introduction of the 1108 System with its many systems components provides the most flexible system available today. The many ways that the individual components can be configured are almost limitless.

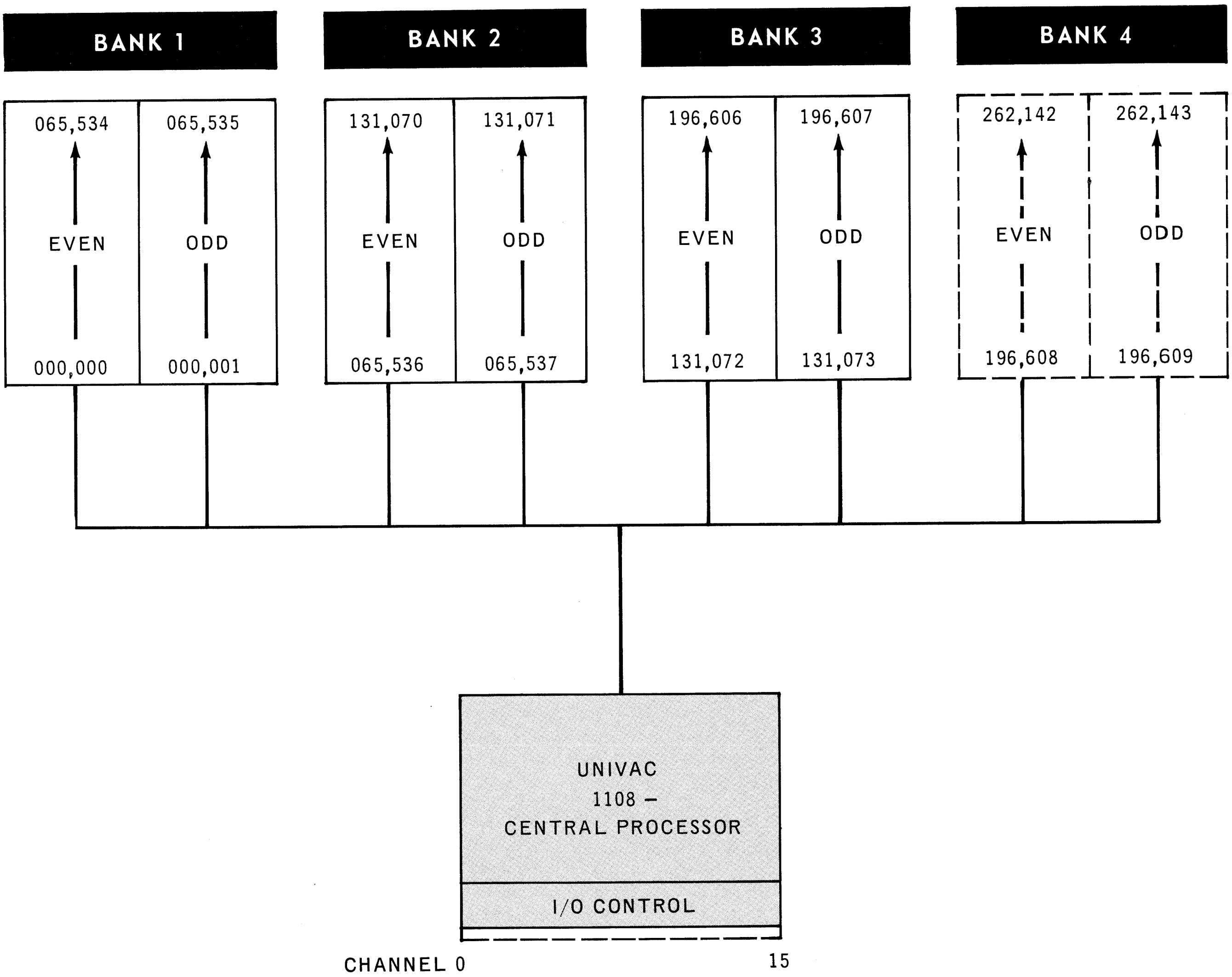
The following illustrations show:

- A Unit Processor system with 65,536 or 131,072 words of main storage
- A Unit Processor system with 196,608 or 262,144 word main storage
- A Unit Processor system with Input/Output Controller
- A Multi-processor system with three Central Processors and two Input/Output Controllers

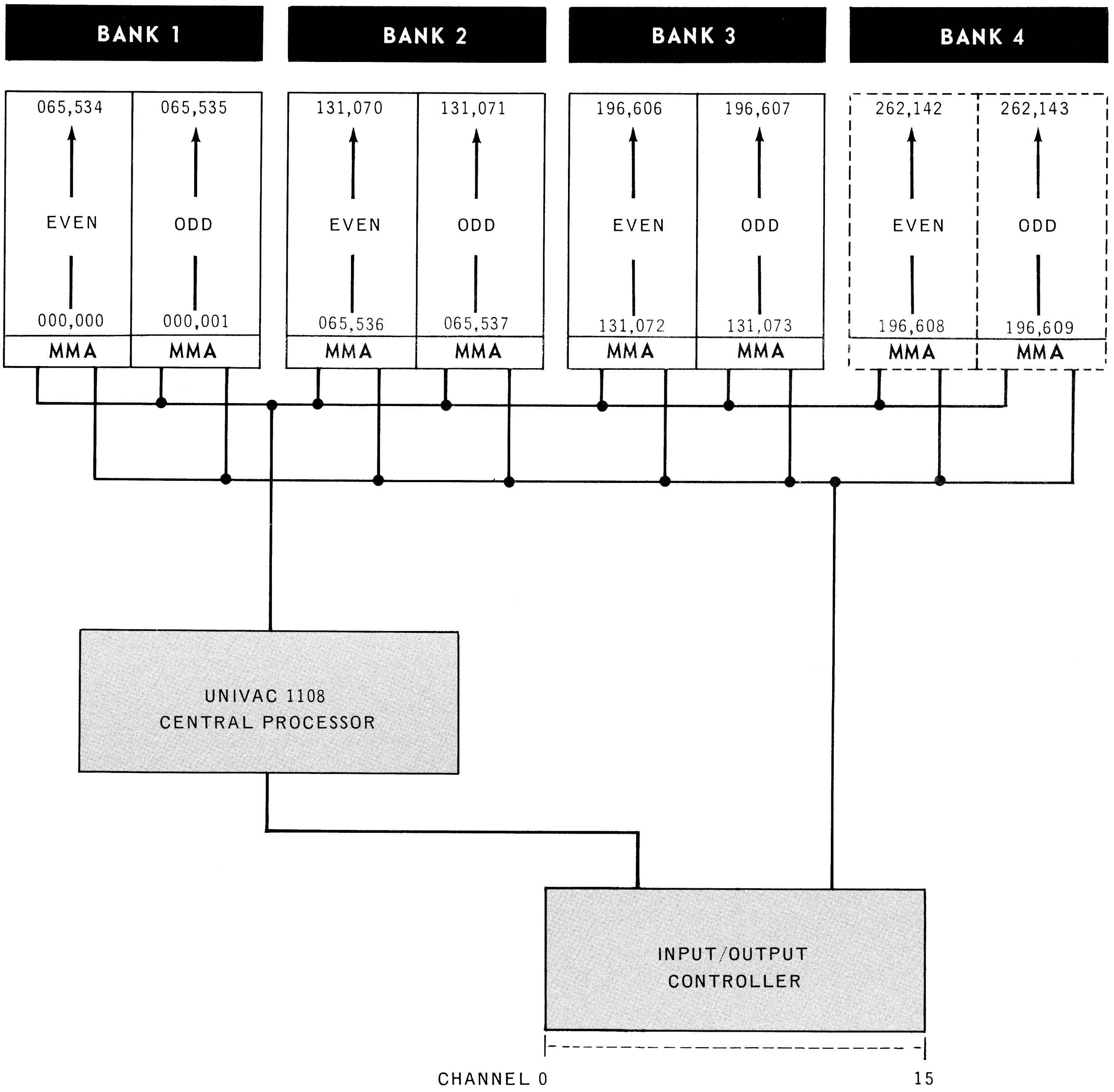


UNIVAC 1108 – UNIT PROCESSOR SYSTEM

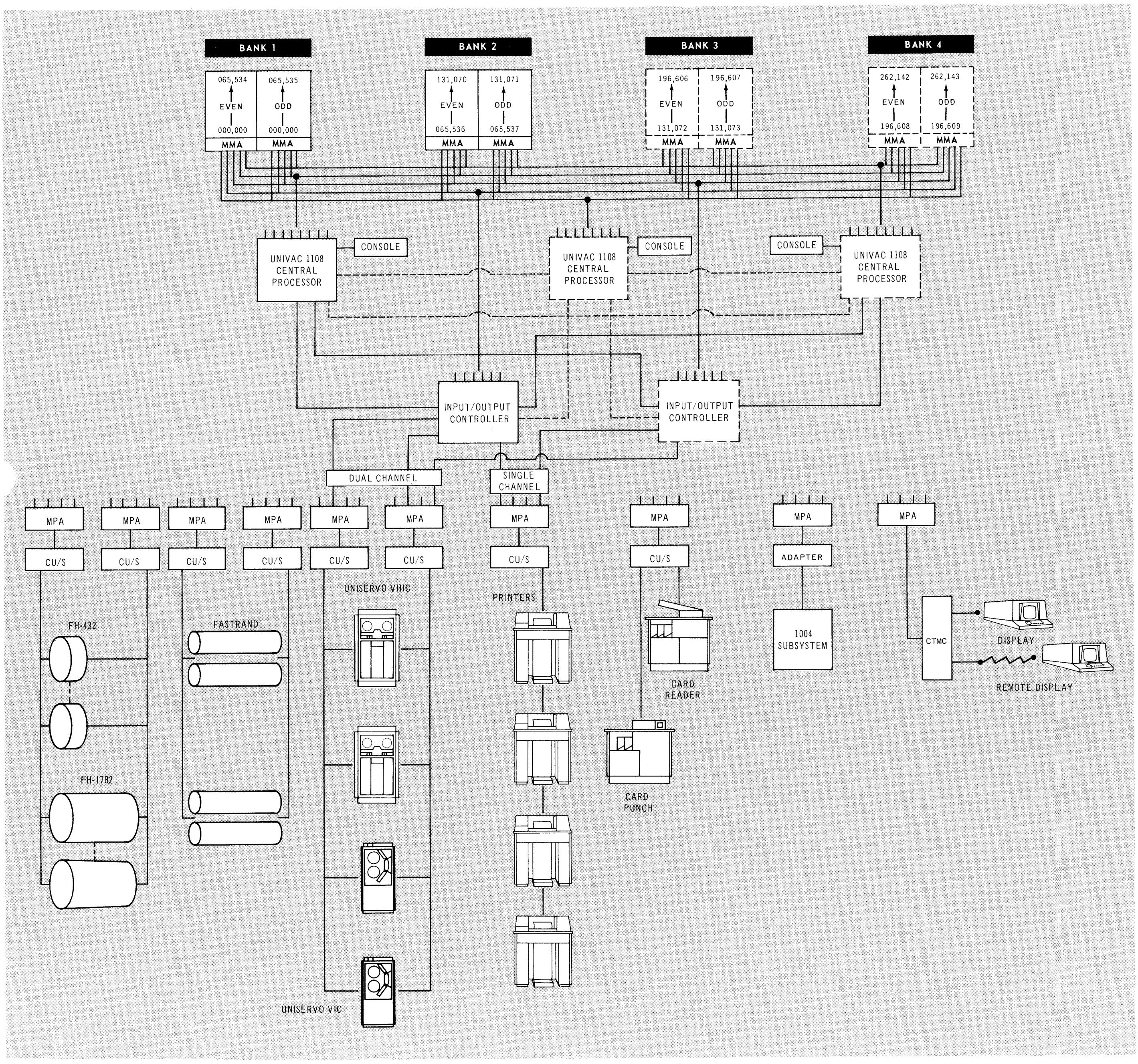
65,536 or 131,072 Word Main Storage



UNIVAC 1108 – UNIT PROCESSOR SYSTEM
196,608 or 262,144 Word Main Storage



UNIVAC 1108 UNIT PROCESSOR SYSTEM WITH I/O CONTROLLER



UNIVAC 1108 – MULTI-PROCESSOR SYSTEM
131,072; 198,608; or 262,144 Word Main Storage

4. MAIN STORAGE

Main Storage of the UNIVAC 1108 System provides a high performance, immediate access repository for instruction, data, and input/output communications areas. Its design fully supports the concepts of multiprogramming, multiprocessing, modularity, reliability, and persistence, around which the entire 1108 System is constructed. Among its featured characteristics are:

- 750 nanosecond read/restore cycle time
- 65,536 to 262,144 36 bit words
- Parity checking on all storage references
- Eight-way simultaneous access by up to five processors
- Modular expansion – two, four, six or eight 32,768 word modules
- Hardware storage protection – Lockout boundaries establishable in 512 word increments
- Relative addressing and dynamic program relocatability through program basing registers
- On-Line serviceability – modules may be removed for servicing without stopping the entire system.
- Overlapping/Interleaving to boost processor performance and minimize access conflicts between processors.

While these features are all discussed generally as storage features, many of them such as relative addressing, storage protection, overlapping/interleaving are actually functions of each processor. With proper multiprocessor system organization, the main storage then becomes a series of allocatable components of the system in the same manner as peripheral devices. In realizing this objective, some departures from the traditional close integration of the processor and the storage element have taken place:

– *The Main Storage is composed of independently accessible modules, yet it presents a continuous addressing structure to the processor(s).*

- *In order to service more than one processor, a method of establishing relative priority between processors at each module. is provided in case two or more processors attempt to reference the same module simultaneously.*

- *To assure that a processor will "wait" for storage access, communication between the processor and the module is done on request/acknowledge basis.*

With these considerations in mind, the Storage modules become passive components with the ability to:

1. *Grant storage access to a number of processors on a fixed priority basis.*
2. *Accept an address from the processor.*
3. *Store or retrieve a word at that address.*
4. *Issue an acknowledge signal signifying that a storage reference has been completed.*
5. *Check parity on each access and deliver an interrupt signal to the attached processor should a parity error occur.*

This type of Processor/Module relationship presents significant advantages for the immediate as well as the future needs of the system.

- Expansion in terms of additional processors or storage modules is simplified.
- The addition of processors or storage elements with advanced performance characteristics can be accomplished on the same modular basis as technology is advanced.

BASIC STORAGE MODULE

The basic storage module is composed of 32,768 words of ferrite core array. Each word is 36 bits in length, and carries two additional parity bits in non-addressable levels, one bit for each half word. The physical components of each module are:

- A 15 bit address register
- A 36 bit read/restore register
- Parity checking circuits
- Request/acknowledge circuits
- Maintenance switches allowing the module to be removed logically for servicing or test.

The 15 bit address register of each storage module provides a continuous addressing structure of 32,768 words. Since an 18 bit address is generated within the processor at each storage reference, three bits are available for selection of one of the eight possible storage modules. It is not necessary to transmit these bits to the module.

MODULE PARITY ERROR

Parity is checked (reading) or calculated (writing) on each storage access. If a parity error is detected the module will issue a parity interrupt signal to the processor to which it is currently attached, and rewrite the word in its incorrect form to assure subsequent errors when the word is again referenced.

By running random and "worst case" patterns, the storage diagnostic function of the Executive can determine whether the failure is purely transient or associated with a marginal or complete outage of the module.

MULTIPLE MODULE ACCESS (MMA)

In a multi-processor system, an MMA unit is positioned between each Storage Module and the several processors which may reference it to resolve potential storage access conflicts. This unit furnishes five priority ordered processor connection paths. Should an access conflict occur between processors the MMA will grant storage access to the processor having the relative highest priority attachment to the MMA, then the next, and so on. Communications between processors and a single storage module can, therefore, be conducted on an asychronous basis – if the storage module is "busy" servicing one processor a passive wait cycle is induced in others of lower priority that may be referencing it.

Because a delay in honoring an input/output transfer can result in an undesirable "go-around" on drum, reread or rewrite on tape; or actual loss of data in the case of real time input, I/O controllers are ordinarily attached to the highest priority inputs of the MMA, followed by 1108 Processors, which have built-in precedence of I/O over computational activities.

PACKAGING

Two 32,768 word Storage Modules, as described above, are packaged in a Storage Cabinet. Each Cabinet contains power, cooling, and cabling interface for five inputs to each module. Storage expansion, then, can be accomplished by the installation of additional cabinets as conditions require.

STORAGE CONFIGURATIONS

Available storage configurations range from 65,536 words to the system maximum of 262,144 words. Growth increments are in "Cabinets" of 65,536 words. The resulting storage capacities are as follows:

A. 65,536 words (two modules)

B. 131,072 words (four modules) – Minimum for multiprocessor system

C. 196,608 words (six modules)

D. 262,144 words (eight modules)

These basic options must be in balance with the number of processing elements in the system; therefore, minimum storage for the multiprocessor system is greater than that for the Unit Processor, primarily due to the increased need for access paths.

ADDRESSING TECHNIQUES

In referencing the main storage modules, two special techniques are employed by the processors to increase processor performance and minimize multiprocessor access conflicts. The first, called overlapping, enables the 1108 processor to retrieve the current operand and the next instruction simultaneously; the second, called interleaving, enables two processors to access a group of modules with minimum access conflicts.

OVERLAPPING

The 1108 processor is capable of determining whether its current operand and next instruction lie in different storage modules. It is also capable, if this situation is present, of retrieving these two words in parallel, at an effective 100% performance increase. Since the I/O controller is not required to reference instructions in main storage, except on command transfers, it does not have, nor does it need, the overlapping feature.

The overlapping feature permits the separation of the instruction and data of a program into separate physical banks. Furthermore, the basing register of the 1108 processor allows either the instruction or data area of a program to be relocated independently – a significant advantage in core compacting to overcome fragmentation.

INTERLEAVING

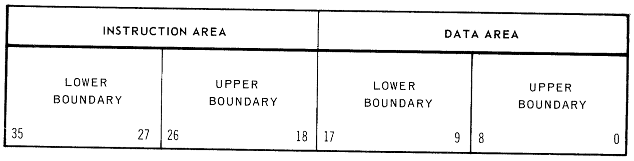
Interleaving is a special process of addressing adjacent storage modules in an even/odd fashion. It significantly reduces storage access conflicts in a multiprocessor system, and thereby increases overall system performance. With interleaving, the modules are divided into even and odd locations (although the addressing structure within the modules themselves remains unchanged). Thus, in a fully-expanded eight module system, modules 0, 2, 4, 6 are referenced for even addresses while modules 1, 3, 5, 7 are referenced for odd.

For a practical example, substitute the letters A, B, C, D for the modules contained in two cabinets, and assume data are being stored sequentially by a program. (The same assumption may be made for instructions being executed sequentially by the same program). With the overlapping feature, assume processor number 1 starts executing instructions and retrieving data, with the instruction area in cabinet 1 and the data area in cabinet 2. For simplicity, assume the starting instruction and data addresses are at even locations. The processor will then reference module A-B-A-B... for sequential instructions, and C-D-C-D... for sequential data locations. In any single storage interval, either modules A-C or B-D will be "busy" while their alternates will be idle. If another processor starts an identical process, only referencing odd addresses to begin with, both processors may run concurrently without one impeding the operation of the other.

Assuming both processors in the above example started at even addresses, the processor with lesser priority will passively "wait" one storage cycle after which the two are again in synchronization and may operate simultaneously.

STORAGE PROTECTION

To prevent inadvertant program reference to out-of-range storage addresses, the 1108 processor provides a hardware storage protection feature called the Storage Limits Register (SLR). A diagram of this register is presented below.



The SLR can be loaded by the Executive System to establish allowable operating areas for each program. These areas are termed the program I (Instruction) and D (Data) areas. When control is given to a particular program, the Executive loads the SLR with the appropriate I and D boundaries.

Before each storage reference, the 1108 processor performs a limits check on the address, comparing against either the I or D field of the SLR. An out-of-limits address will interrupt the processor through a guard mode interrupt allowing the Executive to regain control and take appropriate action.

STORAGE PROTECTION MODES

Through control fields in the Processor State Register, the Executive System may establish various modes of storage protection. The Executive normally operates in "open mode"; that is, the Storage Limits Register may be loaded but the Processor State Register is set to disregard its contents – allowing the Executive to freely reference the entire 262,144 word addressing structure of the system.

PRIVILEGED MODE

Another mode can be established in the PSR for privileged programs. This privileged mode protects against out-of-bounds writes. However, programs (such as Real-Time programs or Executive-controlled subroutines) may enter non-alterable (re-entrant) subroutines, which are part of the Executive. While privileged programs are assumed to be at a high level of checkout, the system is still fully protected against unexpected occurrences since Guard Mode and write protection is still in effect.

JOB PROGRAM MODE

In the job program mode, both read/write and jump storage protection is in effect. Therefore, job programs are limited entirely to those areas assigned by the Executive. If the job program reads, writes, or jumps to an out-of-limits address, an interrupt will return control to the Executive for remedial action.

Read/jump protection allows the Executive to stop the program at the point of error, terminate it, and provide diagnostic information to the programmer, thereby minimizing wasted time and smoothing the checkout process.

A particular advantage of read/jump protection is that programs of a classified nature can be confidently run together; they are fully protected from audit (inadvertant or otherwise) by other programs.

RELATIVE ADDRESSING

Relative addressing is a feature of great significance in multiprogramming, time-sharing, and real time operations, for it allows storage assignments to be changed dynamically to provide contiguous storage for operation of another program, and permits programs to dynamically request additional main storage according to processing needs. An additional advantage is that systems programs stored on mass storage may be brought in for operation in any available area without complicated relocation algorithms.

Relative addressing is provided for through basing registers contained within the 1108 processor. A separate register controls the basing of the program instruction and data bank, and a third register controls the selection of the appropriate basing register.

5. 1108 PROCESSOR

The UNIVAC 1108 Processor is the principal component of the 1108 system and, generally, the one by which the entire 1108 system is identified. Its functions provide for both Executive and worker program modes of control, arithmetic and logical operations, and control of up to 16 input/output channels.

The processor is logically divided into six interacting sections. For a clearer understanding of the composite function of these sections, each is identified and briefly described below.

Control Section

Provides the basic phasing and logic for instruction decoding and execution. It includes the Program Address Counter (P) used for the sequencial accession of instructions; the Program Control Register (PCR) in which instructions are staticized for execution; and the Processor State Register (PSR), which determines various processor operating modes. The Control Section also services interrupts.

Arithmetic Section

Contains the adder registers and control circuits necessary for performance of fixed and floating point arithmetic; partial word selection; shifting; logical operations, and tests. Arithmetic operations overlap with input/output operations.

Input/Output Section

Controls and multiplexes data flow between main storage and 16 input/output channels. Provides interrupt priority network and exchange paths to peripheral subsystems for both functions and data.

Indexing Section

Contains parallel index adders and threshold test circuitry. Used generally for processor control functions, operand address development, program relocation, and input/output transfer control.

Storage Class Control Section

The Storage Class Control section receives the final operand address from the index adder and, based on installed storage capacity and configuration, establishes address and data paths to one of 8 possible storage modules. Storage Class Control also determines whether a final address refers to control registers.

Control Registers

The 1108 Processor has 128 program-addressable control registers used for arithmetic operation, indexing, and input/output buffer control.

CONTROL REGISTERS

The 128 program addressable control registers are grouped to provide multiple index registers, accumulators, channel control registers, and special registers. Processor state modes determine which of two sets of these registers are referenced in programmed operations. One of the sets is reserved for the Executive system.

The control registers consist of 128 program-addressable integrated-circuit registers, each having a word size of 36 bits and a basic cycle time of 125 nanoseconds. Two parity bits are present with each control register.

Control registers are divided logically into groups of index registers, accumulators, input and output access control registers and R-Registers.

Effective use of multiple accumulators and index registers for the development and use of constants, index values, and operands substantially improves performance. UNIVAC 1108 compilers, for example, derive significant performance advantages through multiple register usage, and are able to produce highly optimized and efficient code.

In the following descriptions only the programmable registers will be mentioned. The Executive has exclusive use of the duplicate set of A, B and R Registers as well as the Access Control Registers indicated by the shaded areas in Diagram 5-1.

Index Registers

Control register locations 00-15 are grouped as Index Registers.

Δ	Q
35 18	17 0

Δ *(18 bits) – increment or decrement*
Q (18 bits) – modifier

The *Q* portion of the index register is an 18 bit modifier to be added to the base operand address of the instruction. The Δ portion of the index word updates the *Q* portion, *after* base operand modification.

Index register modification is specified by the *h* bit in the instruction designator, while indexing itself is specified by a non-zero value in the instruction x field.

Both functions are accomplished within the basic instruction execution cycle. For $x = 0$, both indexing and index modification are suppressed.

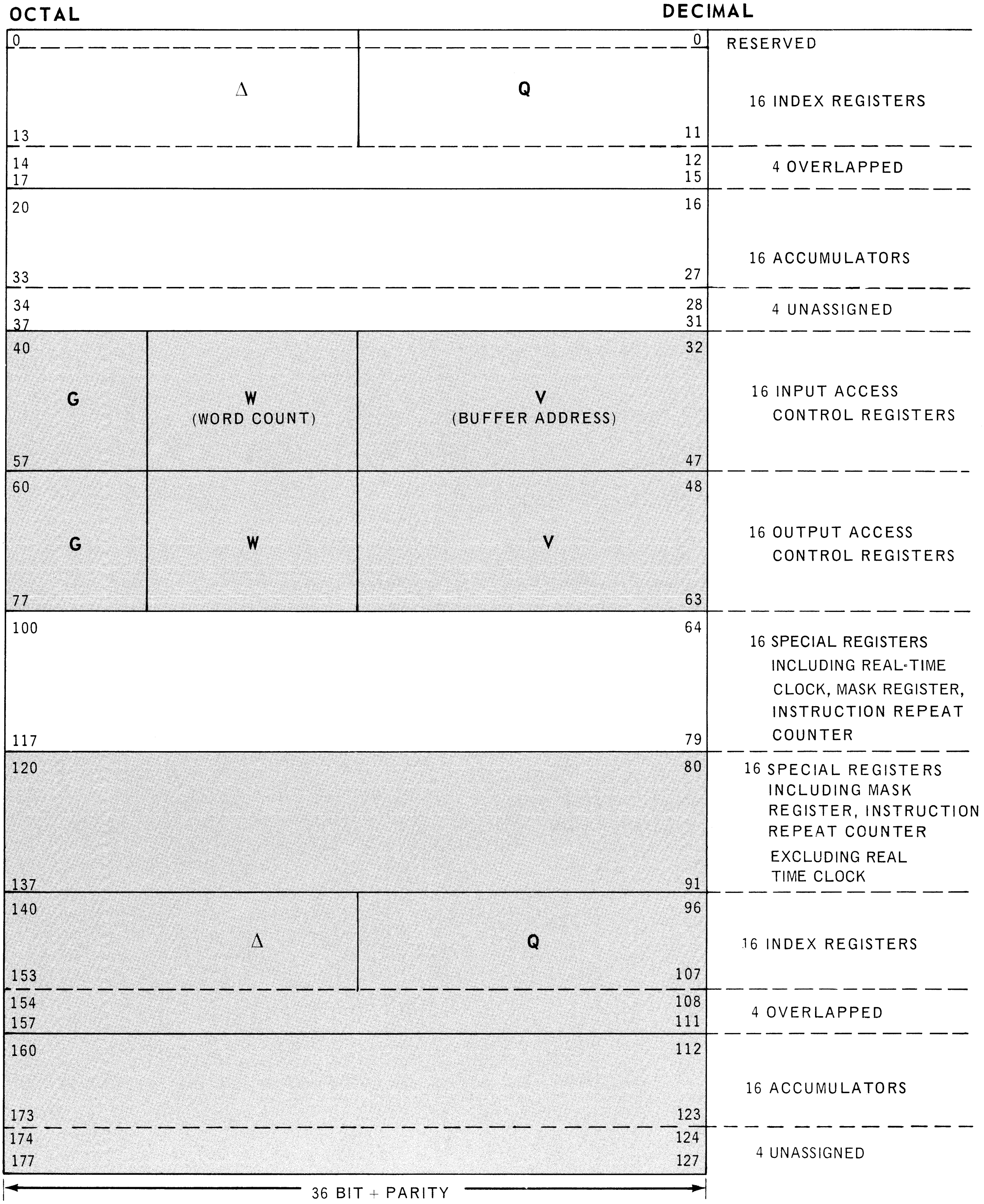
When cascaded indirect addressing is used in a programmed operation, full indexing capabilities are provided at each level. Indirect addressing replaces the *x, h, i,* and *u* portions of the program control register, beginning with a new indexing cycle for each cascaded sequence. This process continues until $i = 0$.

Index Register 00, while program addressable, is used as a recipient for the Processor State Register (PSR) upon occurrence of an interrupt. Since its contents are overwritten at each interrupt it is not generally useful for programming purposes.

Arithmetic Accumulators

Control register locations 12-27 are grouped as arithmetic accumulators, for programmed storage of arithmetic operands and results. Actual computation is performed in other registers internal to the arithmetic section.

Depending upon the instruction, usage of the accumulators results in a variety of word formats. Double precision instructions and a number of logical instructions reference two contiguously located accumulators, i.e., A and A + 1. In arithmetic operations, A + 1 always holds the least significant value of an operand or result. Some instructions, as in single precision floating point operations, call on a one word operand from memory but produce a two word result in the specified A and A + 1.



OCTAL				DECIMAL	
0				0	RESERVED
13	Δ		Q	11	16 INDEX REGISTERS
14 17				12 15	4 OVERLAPPED
20 33				16 27	16 ACCUMULATORS
34 37				28 31	4 UNASSIGNED
40 57	G	W (WORD COUNT)	V (BUFFER ADDRESS)	32 47	16 INPUT ACCESS CONTROL REGISTERS
60 77	G	W	V	48 63	16 OUTPUT ACCESS CONTROL REGISTERS
100 117				64 79	16 SPECIAL REGISTERS INCLUDING REAL-TIME CLOCK, MASK REGISTER, INSTRUCTION REPEAT COUNTER
120 137				80 91	16 SPECIAL REGISTERS INCLUDING MASK REGISTER, INSTRUCTION REPEAT COUNTER EXCLUDING REAL TIME CLOCK
140 153	Δ		Q	96 107	16 INDEX REGISTERS
154 157				108 111	4 OVERLAPPED
160 173				112 123	16 ACCUMULATORS
174 177				124 127	4 UNASSIGNED

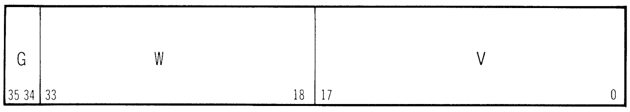
36 BIT + PARITY

▨ = EXECUTIVE

CONTROL REGISTER ADDRESS DIAGRAM

Access Control Registers

Control register locations 32-63 are grouped as Input and Output Access Control Registers (ACR's). They are guard mode protected and referenced only by the Executive.



G = *the incrementation designator*
W = *the number of words to be transferred*
V = *the initial storage address to or from which data will be transferred. Specifies incremental, decremental or single-word buffer operations.*

G	OPERATION	NEXT ADDRESS
00	Increment V Address	V + 1
01	Inhibit Increment	V
10	Decrement V Address	V − 1
11	Inhibit Decrement	V

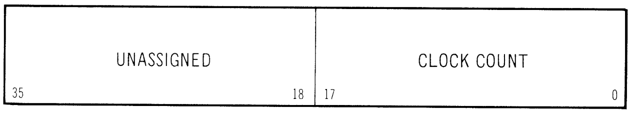
The word by word transmission of data over an I/O channel is governed by the ACR's. Two ACR's, one for input and one for output, are assigned to each of the sixteen channels. Input ACR's (locations 32-47) control input data transfers while output ACR's (locations 48-63) govern the transmission of output Data and Function Words.

In initiating an input/output operation, the programmed ACR word is loaded in the ACR corresponding to the channel associated with the specified peripheral unit.

Special Purpose R-Registers

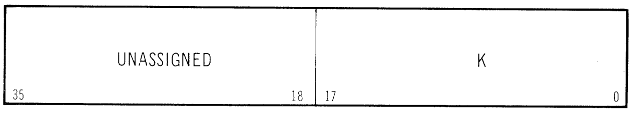
Control register locations 64-67 are assigned as special purpose registers (R0-R2). Each of these registers is assigned a special function and has a special format.

R0 – (Real Time Clock)



The contents of the Real Time Clock are decremented once each 200 microseconds. A Real Time Clock interrupt occurs when the clock count is decremented through zero. By loading the Real Time clock with the value 5000, an interrupt will occur in exactly one second.

R1 – (Repeat Counter)



k (18 bits) = *Number of times an instruction is to be executed.*

The Repeat Counter is used to control repeated operations such as Block Transfer and Search instructions. To execute a repeated instruction *k* times, the repeat counter is loaded with *k* prior to the execution of the instruction.

R2 – (Mask Register)

The Mask Register functions as a filter in determining which portions of words are tested in repeated masked search operations or logical comparisons. *(U)* is compared to *(A)* in only those positions where corresponding one's exist in the Mask Register. In repeated masked search operations, both the Mask Register and the Repeat Counter are loaded prior to executing the actual search command.

The remaining R-Registers are not specifically assigned; they may be used as loop counters, transient registers, or to hold intermediate values or constants.

In programmed register-to-register transfers (such as loading an R-Register from an accumulator or vice-versa) only full words are transferred and any partial word selection by the j-designator is ignored.

INSTRUCTION REPERTOIRE

The UNIVAC 1108 Processor provides an unusually powerful and flexible instruction repertoire. Most 1108 instructions are accessed and completed in one storage cycle. In addition to a complete set of standard instructions including an extremely fast set of single and double precision floating point instructions, the repertoire includes those which permit fast and simplified control by the Executive System operating in a multiprogramming or multiprocessing environment.

Instruction Format

The format of the 1108 instruction is illustrated below followed by an explanation of each field. Some fields have different meanings depending on the class of instruction.

f	j	a	x	h	i	u
35 30	29 26	25 22	21 18	17	16	15 0

f – (Function Code)

These six bits specify the operation to be performed. For function codes above 70_8 the f and j fields are combined to produce a 10 bit function code. If an illegal code is encountered in an instruction, an interrupt occurs.

j – (Partial Word or Immediate Operand Designator)

For function codes less than 70_8, the j designator specifies partial word or immediate operand selection. (See the ARITHMETIC section for the various partial word selections allowed.)

a – (Control Register Designator)

The a-field designates which control register, within a group selected by the function code, is involved in the operation. For some operations the a-field refers to an arithmetic register; for others an index register or other control register; in input/output instructions it specifies the channel and its associated input or output channel control register location.

x – (Index Register Designator)

The x-field specifies one of the 15 index (x) registers to be used in address modification. When index register 00 is designated, indexing is suppressed.

h – (Index Modification Designator)

The h-field controls modification of the index value (Q) by the increment field (Δ) after indexing. If $h=0$, do not modify. If $h=1$, the index register's right half is modified by the contents of its left half.

i – (Indirect Address Designator)

The i designator controls the use of indirect addressing during instruction execution. If $i=0$, the instruction functions normally. If $i=1$, the 22 least significant bit positions of the instruction (x, h, i and u fields) are replaced in PCR with the 22 least significant bit positions of (U). Indirect addressing will continue so long as $i=1$ with full indexing capability at each cascade level.

u – (Address Field)

The u-field normally specifies the operand address. For certain instructions it is used to hold constants; for example, the shift instructions use the 7 least significant bit positions to hold the shift count. In all instructions the u-field may be modified by the contents of an index register.

1108 Instruction Families

The commands of the 1108 repertoire are grouped according to their function, rather than by a numeric listing of operation codes. The octal operation code is given in parenthesis. By this grouping it is possible to gain a better appreciation of the power of the repertoire.

DATA TRANSFER

To load the arithmetic registers:
Load A (10)
Load Negative A (11)
Load Magnitude A (12)
Load Negative Magnitude A (13)

To load other control registers (Index and R Registers):

Load R (23)
Load X (27)
Load X Modifier (26)
Load X Increment (46)

To load two arithmetic registers with one instruction:
Double Load A (71, 13)
Double Load Negative (71, 14)
Double Load Magnitude (71, 15)

To store the arithmetic registers:
Store A (01)
Store Negative A (02)
Store Magnitude A (03)

To store other control registers:
Store X (06)
Store R (04)

To store two arithmetic registers with one instruction:
Double Store A (71,12)

Two special purpose transfers:
Store Zero (05)
Block transfer, repeated (22)

With any transfer instructions, except double length, it is possible to move selected parts of words. That is, the partial word feature allows any sixth, third, or half word to be loaded into the lower portion of an arithmetic register, when using load instructions. Similarly, when using a Store instruction, any sixth, third, or half word portion of a word can be transmitted from the lower portion of an arithmetic register, index register, or R-register.

FIXED POINT ARITHMETIC

Full word operations on arithmetic registers:
Add to A (14)
Add Negative A (15)
Add Magnitude to A (16)
Add Negative Magnitude to A (17)
Add Upper (20)
Add Negative Upper (21)
Multiply Integer (30)
Multiply Single Integer (31)
Multiply Fractional (32)
Divide Integer (34)
Divide Single Fractional (35)
Divide Fractional (36)

Double length operations on two arithmetic registers:
Double Precision Fixed Point Add (71, 10)
Double Precision Fixed Point Add Negative (71, 11)

Special format operations:
Add Halves (72, 04)
Add Negative Halves (72, 05)
Add Thirds (72, 06)
Add Negative Thirds (72, 07)
Add to X (24)
Add Negative to X (25)

FLOATING POINT ARITHMETIC

In the 1108 there are two basic types of floating point: single precision and double precision. Both types employ the same kind of arithmetic, one's complement. They differ primarily in that single precision uses one word operands, whereas double precision uses two word operands.

Single precision:
Floating Add (76, 00)
Floating Add Negative (76, 01)
Floating Multiply (76, 02)
Floating Divide (76, 03)
Load and Unpack Floating (76, 04)
Load and Convert to Floating (76, 05)

Double precision:
Double Precision Floating Add (76, 10)
Double Precision Floating Add Negative (76, 11)
Double Precision Floating Multiply (76, 12)
Double Precision Floating Divide (76, 13)
Double Load and Unpack Floating (76, 14)
Double Load and Convert to Floating (76, 15)

Miscellaneous:
Magnitude of Characteristic Difference to Upper (76, 06)
Characteristic Difference to Upper (76, 07)
Floating Expand and Load (76, 16)
Floating Compress and Store (76, 17)

INDEX REGISTER INSTRUCTIONS

These instructions can be used when changing, loading, etc. Index registers:

Add to X (24)
Add Negative to X (25)
Load X Modifier (26)
Load X (27)
Store X (06)
Load X Increment (46)
Load Modifier and Jump (74, 13)
Test Less or Equal to Modifier (47)
Jump Modifier Greater and Increment (74, 12)

The above instructions address the appropriate index register. Four of the index registers are overlapped with the arithmetic registers; thus all arithmetic instructions are available for working with these 4 index registers, such as multiply or shift.

LOGICAL FUNCTIONS

The logical or Boolean operations are defined as shown by the truth tables below:

Logical AND

	01
0	00
1	01

Inclusive OR,

	01
0	01
1	11

Exclusive OR,

	01
0	01
1	10

The three simple logical instructions are:

Logical Or (40)
Logical Exclusive Or (41)
Logical And (42)

One special replace instruction:

Masked Load Upper (43)

Several other instructions employ logical operations in combination with other functions, such as, the Repeated Masked Searches.

SHIFT INSTRUCTIONS

Twelve shift functions are available in the 1108. These are provided in the form of circular, logical, and algebraic shifts. Circular shifts are end-around. Logical shifts fill in zeros on the end opposite the shift direction, whereas arithmetic shifts fill in sign bits. The shift count (from 0 through 72 places) is taken from the u-field of the shift instruction.

Single Shift Circular	(73, 00)
Double Shift Circular	(73, 01)
Single Shift Logical	(73, 02)
Double Shift Logical	(73, 03)
Single Shift Algebraic	(73, 04)
Double Shift Algebraic	(73, 05)
Load Shift and Count	(73, 06)
Double Load Shift and Count	(73, 07)
Left Single Shift Circular	(73, 10)
Left Double Shift Circular	(73, 11)
Left Single Shift Logical	(73, 12)
Left Double Shift Logical	(73, 13)

REPEATED SEARCH INSTRUCTIONS

Search instructions operate as repeated compare operations, skipping the next instruction when a condition is met or taking the next instruction in sequence when the condition is not met and the repeat count equals zero.

Algebraic (Sign considered):

Search for Equal	(62)
Search for Not Equal	(63)
Search for Less or Equal	(64)
Search for Greater	(65)
Search for Within Range	(66)
Search for Not Within Range	(67)

Masked Algebraic (Sign considered):

Masked Search for Equal	(71, 00)
Masked Search for Not Equal	(71, 01)
Masked Search for Less or Equal	(71, 02)
Masked Search for Greater	(71, 03)
Masked Search for Within Range	(71, 04)
Masked Search for Not Within Range	(71, 05)

Masked Alphanumeric (Sign treated as data):

Masked Alphanumeric Search for Less or Equal	(71, 06)
Masked Alphanumeric Search for Greater	(71, 07)

UNCONDITIONAL JUMPS

These instructions transfer control to the location specified by the indexed u address.

Store Location and Jump	(72, 01)
Prevent All Interrupts and Jump	(72, 13)
Load Modifier and Jump	(74, 13)
Allow All Interrupts and Jump	(74, 07)

CONDITIONAL JUMP INSTRUCTIONS

If the condition is met, program control is transferred to the instruction location specified by u. If not, the next instruction in sequence is executed.

Jump on Greater and Decrement	(70)
Double Precision Zero Jump	(71, 16)
Jump on Positive and Shift	(72, 02)
Jump on Negative and Shift	(72, 03)
Jump on Zero	(74, 00)
Jump on Non Zero	(74, 01)
Jump on Positive	(74, 02)
Jump on Negative	(74, 03)
Jump on Keys, Jump	(74, 04)
Halt on Keys and Jump, Halt and Jump	(74, 05)
Jump on No Low Bit	(74, 10)
Jump on Low Bit	(74, 11)
Jump Modifier Greater and Increment	(74, 12)
Jump on Overflow	(74, 14)
Jump on No Overflow	(74, 15)
Jump on Carry	(74, 16)
Jump on No Carry	(74, 17)
Jump on Input Channel Busy	(75, 02)
Jump on Output Channel Busy	(75, 06)
Jump on Function in Channel	(75, 12)

TEST (OR SKIP) INSTRUCTIONS

If the condition is met, the next instruction is skipped. If not, the next instruction is executed.

Test Even Parity	(44)
Test Odd Parity	(45)
Test Less or Equal to Modifier	(47)
Test for Zero	(50)
Test for Non Zero	(51)
Test for Equal	(52)
Test for Not Equal	(53)
Test for Less or Equal	(54)
Test for Greater	(55)
Test for Within Range	(56)
Test for Not Within Range	(57)
Test for Positive	(60)
Test for Negative	(61)
Double Precision Test Equal	(71, 17)

OTHER INSTRUCTIONS

Execute	(72, 10)
No Operation	(74, 06)

EXECUTIVE CONTROL INSTRUCTIONS

This group of instructions allows proper Executive System control of programs operating in a multiprogramming and multiprocessing environment.

Executive Return	(72, 11)
Store Channel Number	(72, 14)
Load Processor State Register	(72, 15)
Load Storage Limits Register	(72, 16)
Initiate Synchronous Interrupt	(73, 14)
Select Interrupt Module	(73, 15)
Load Channel Select Register	(73, 16)

These instructions are used for establishing Processor State, storage limits boundaries, interrupt module locations, identification of I/O channels, and interprocessor communication and task assignment.

INPUT/OUTPUT INSTRUCTIONS

This group of instructions allows the program (usually the Executive System) to initiate and control input/output operations.

Load Input Channel	(75, 00)
Load Input Channel and Monitor	(75, 01)
Jump on Input Channel Busy	(75, 02)
Disconnect Input Channel	(75, 03)
Load Output Channel	(75, 04)
Load Output Channel & Monitor	(75, 05)
Jump on Output Channel Busy	(75, 06)
Disconnect Output Channel	(75, 07)
Load Function in Channel	(75, 10)
Load Function in Channel and Monitor	(75, 11)
Jump on Function in Channel	(75, 12)
Allow Function in Channel	(75, 13)
Allow All Channel Interrupts	(75, 14)
Prevent All Channel Interrupts	(75, 15)
Allow Channel Interrupt	(75, 16)
Prevent Channel Interrupt	(75, 17)

EXECUTIVE SYSTEM CONTROL FEATURES

The UNIVAC 1108 System operates in a multiprogram or multiprocessing environment and to initiate and preserve this, the Executive Routine must be in complete control of the total system. Special hardware features are provided to permit this control.

The multiprogramming and multiprocessing capabilities of the UNIVAC 1108 System are based upon Guard Mode operation, the setting aside of certain instructions, registers, and storage locations for the exclusive use of the Executive Routine, assuring protection against the interaction of unrelated programs.

Interrupts

The interrupt network of the UNIVAC 1108 System is extensive, and provides the basis for efficient real time, multiprogram and time-sharing operations on the 1108. The UNIVAC 1108 System has more than 125 separate and distinct interrupt conditions. The synchronization of input/output activities and response to real time situations is accomplished through these interrupts. The interrupt is a control signal that may be received from a peripheral subsystem (external interrupt) or from the control or indexing section of the central processor. Specific interrupt locations are located in the lower regions of main storage for the conditions involved. Interrupt locations are programmed to capture the interrupted address and enter interrupt response subroutines in the Executive Routine. Special interrupts are provided for certain error conditions within the central processor. These may result from a programming fault such as illegal instruction, arithmetic overflow, a central storage parity error or an Executive Routine violation (attempt to write into a protected area of storage or violation of guard mode). These faults have special interrupt locations in memory and are used by the Executive Routine to take remedial or terminating action when they are encountered.

The following diagram describes the specific interrupt locations.

DECIMAL ADDRESS	OCTAL ADDRESS	FIXED ASSIGNMENT
128-141	200-215	Reserved
142	216	Day Clock
143	217	Day Clock Interrupt
144	220	Input Monitor Interrupt
145	221	Output Monitor Interrupt
146	222	Function Monitor Interrupt
147	223	External Interrupt
148	224	ESI Input Monitor Interrupt
149	225	ESI Output Monitor Interrupt
150	226	Power Loss Interrupt
151	227	ESI External Interrupt
152	230	Status Word for External Interrupt
153	231	Real Time Clock Interrupt
154	232	External Synchronous Interrupt #1
155	233	External Synchronous Interrupt #2
156	234	Main Storage Parity Error (Module #1)
157	235	Main Storage Parity Error (Module #2)
158	236	Main Storage Parity Error (Module #3)
159	237	Main Storage Parity Error (Module #4)
160	240	Control Register Parity Error
161	241	Illegal Instruction Interrupt
162	242	Executive Return Interrupt
163	243	Guard Mode Interrupt
164	244	Not Used
165	245	Floating Point Underflow Interrupt
166	246	Floating Point Overflow Interrupt
167	247	Divide Fault Interrupt
Last address-1		Central Store Parity Error (Module #1 Alternate)

Executive Control Instructions

This group of instructions allows the proper control of programs operating in a multiprogramming or multiprocessing environment.

Executive Return	(72, 11)
Store Channel Number	(72, 14)
Load Processor State Register	(72, 15)
Load Storage Limits Register	(72, 16)
Synchronous Interrupt	(73, 14)(#1 – #2)
Select Interrupt Module	(73, 15)
Load Channel Select Register	(73, 16)

These instructions are used for establishing Processor State, storage limits boundaries, interrupt module locations, identification of I/O channels, and interprocessor communication.

GUARD MODE

Guard Mode prevents programs from executing any of a set of instructions reserved for the Executive. It also protects certain locations reserved for Executive Operations.

Guard Mode is established by the LOAD PROCESSOR STATE REGISTER Command. Execution of this with an appropriate PSR Bit pattern is the *only* way that Guard Mode can be made operative and provides the only access to the PSR directly. Under Guard Mode, an attempt to perform any of the privileged instructions or tunctions listed below will cause a processor interrupt.

Instructions

Load Processor State Register
Load Storage Limits Register
Initiate Interprocessor Interrupt
Select Interrupt Module
Load Channel Select Register
Stop Instructions
All I/O Instructions

Functions

A Write in Control Registers 32-64 or 80-127.
Disabling of interrupts foı more than 100 microseconds.

It is possible for any program to use the PREVENT ALL INTERRUPTS AND JUMP Instruction, thereby allowing real time programs to disable non real time interrupt processing temporarily.

Under Guard Mode, however, interrupts cannot be disabled for over 100 microseconds.

Guard Mode is disabled by the occurrence of an interrupt, which stores the contents of PSR in Index Register 0, clears the PSR codes and establishes Executive Mode operation.

PROCESSOR STATE REGISTER

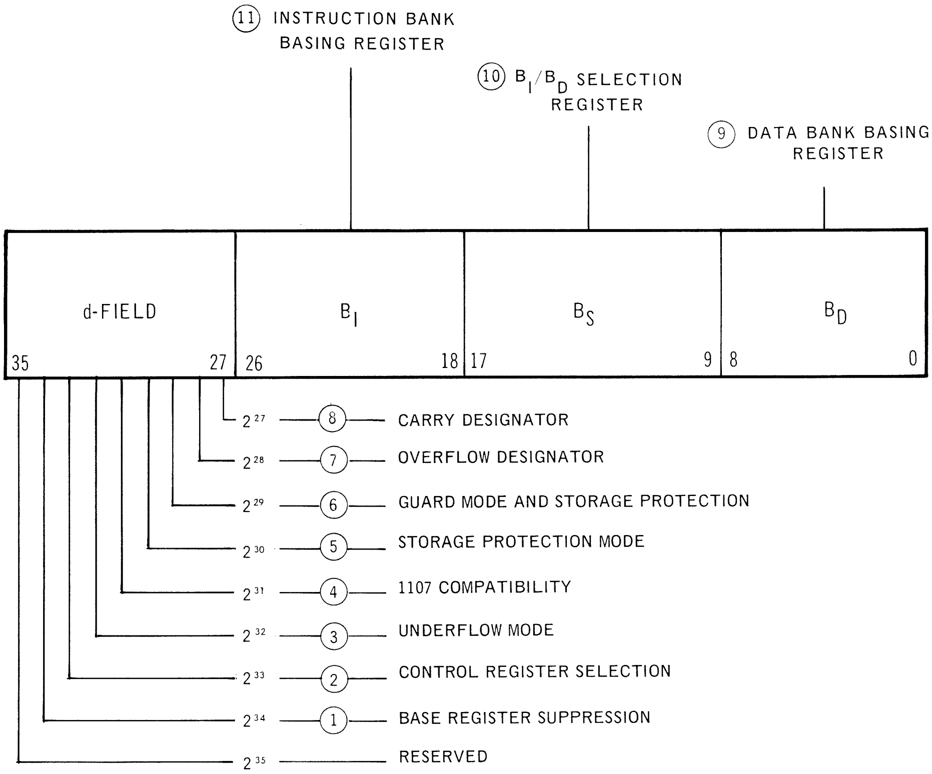
The Processor State Register (PSR) is a 36 Bit representation of various states and conditions affecting the current operations of the 1108 Processor. Through this register the Executive sets up control modes for itself, governs the operation of worker programs, and registers status information concerning worker programs when it regains control via interrupts.

A special instruction LOAD PROCESSOR STATE REGISTER is available to the Executive system for loading PSR, and governing the following functions and conditions:

- Program Basing addresses
- Carry and overflow status
- Guard Mode
- Storage protection mode
- 1107 compatibility mode
- Floating underflow mode (Double precision operations)
- Base register suppression
- Control register process selection

Storing of the PSR is done automatically in the following manner: when an interrupt occurs, program carry and overflow status is registered in PSR and its contents are transferred to Control Register location 00 (Index Register zero). PSR is then force-cleared to a state of readiness for Executive operations (which will be described later). By safe-storing Control Register 00 the Executive can reinstate conditions as control is passed to each program.

The format of the PSR is as follows:



11 INSTRUCTION BANK BASING REGISTER
10 B_I/B_D SELECTION REGISTER
9 DATA BANK BASING REGISTER
d-FIELD
B_I
B_S
B_D
35
27
26
18
17
9
8
0
2^27
8 CARRY DESIGNATOR
2^28
7 OVERFLOW DESIGNATOR
2^29
6 GUARD MODE AND STORAGE PROTECTION
2^30
5 STORAGE PROTECTION MODE
2^31
4 1107 COMPATIBILITY
2^32
3 UNDERFLOW MODE
2^33
2 CONTROL REGISTER SELECTION
2^34
1 BASE REGISTER SUPPRESSION
2^35
RESERVED

PROCESSOR STATE REGISTER FORMAT

(1) BASE REGISTER SUPPRESSION 2^{34}

= 0 Base Registers – allows Basing Registers to be additive to every storage reference

= 1 Basing Register addition on storage reference is suppressed when instruction *i* – designator = 1.

(2) CONTROL REGISTER SELECTION 2^{33}

= 0 Worker Program A, X & R Register Set (Locations $00\text{-}37_8$, $100\text{-}120_8$)

= 1 Executive A, X & R Register Set (Locations $120\text{-}177_8$)

The Executive passes control to Worker Programs with $2^{33} = 0$, which selects the worker set. An interrupt forces this bit to "1" after PSR has transferred to X0, making the upper A, X & R Registers available to the Executive.

(3) UNDERFLOW MODE – Double Precision Floating Point Operations 2^{32}

= 0 Interrupt on D.P. Floating Underflow

= 1 Clear results to zero and continue

This is a program-requestable option which is set up for the program by the Executive.

(4) 1107 COMPATIBILITY MODE 2^{31}

= 0 1108 Mode – Full Range Addressing

= 1 The upper two bits of the effective address (U) are stripped allowing 1107 program compatibility with the 1108. In this mode, only the lower 65,000

(5) STORAGE PROTECTION MODE 2^{30}

= 0 Write Protection Only under Guard Mode

= 1 Read, Write and Jump storage protection under Guard Mode. With Guard Mode "OFF", storage protection is disabled allowing reads, writes or jumps anywhere in storage.

(6) GUARD MODE AND STORAGE PROTECTION

= 0 Guard Mode "OFF". All instructions and storage references to Access Control Registers ($40\text{-}77_8$), the Real Time Clock (100_8) and Executive Control Registers ($120\text{-}177_8$) are permitted.

= 1 Guard Mode "ON". Invalidates all instructions and control register references described above to enforce the integrity of the system. Only when Guard Mode is on are the contents of the storage limits register effective in storage protection.

(7) OVERFLOW DESIGNATOR

= 1 Arithmetic Overflow condition extant at time of interrupt.

(8) CARRY DESIGNATOR

= 1 Arithmetic Carry condition extant at time of interrupt.

(9-11) BASING REGISTERS. These registers provide the absolute basing values on which programs "FLOAT" during execution.

ARITHMETIC SECTION

In the UNIVAC 1108 System the manipulation of data (addition, subtraction, multiplication, division, shifting) is accomplished in the arithmetic section of the central processor. During the execution of an arithmetic instruction, temporary internal storage registers within the arithmetic section itself are used for actual computation. The following are features of the arithmetic section:

- By means of the j-designator in the instruction word, a portion of one of the operands (half, third, or sixth word) can be selected for use in the arithmetic operation.
- Special split word arithmetic instructions provide for simultaneous addition or subtraction of corresponding half or third words in two operands.
- Using the shift matrix, multiposition shifts are accomplished in the same time as a one place shift. Right and left shifts of single or double length operands can be specified. Left shifting is logical (zeros are filled to the right). Right shifts are logical or arithmetic (sign bits are filled to the left).
- Sixteen arithmetic registers in the control register section, acting as sixteen accumulators, allow parallel and cumulative computation. Full double precision floating point arithmetic is provided.
- When the results of arithmetic operations are in double length form, they are automatically stored in the control registers and are available for retrieval as double length results.
- Alphanumeric compares utilizing the Mask Register allow fields of any size, contained within a 36-bit word, to be directly compared.

Adder

The adder in the UNIVAC 1108 System is a one's complement subtractive adder for 36 bit or 72 bit operations. For purposes of analysis and debugging, the programmer may manually simulate the computer operation by simple binary or octal addition.

Two special internal designators are associated with the arithmetic adder, the overflow designator and the carry designator. The fixed point addition and subtraction instructions, single and double precision, are the only instructions which affect these two designators. Before the execution of one of these instructions both designators are cleared.

The overflow designator is set upon generation of a significant bit in the sign position. A positive result of two negative quantities or a negative result of two positive quantities will set the overflow designator. The carry designator is set whenever an end around carry is generated, and indicates the involvement of a negative quantity as one of the operands.

After the instruction has been performed, the designators remain either set or clear until another one of the designated arithmetic instructions is initiated. Both designators are set in time to be tested immediately after the specified instruction has been executed.

The settings of the carry and overflow designators are always contained in the PSR when an interrupt occurs and control is given to the Executive System. These designator conditions are restored when control is returned to the affected program.

Arithmetic Accumulator

The sixteen arithmetic accumulators can be addressed directly by the programmer and are available for storing operands and results of arithmetic computations. These arithmetic accumulators should not be confused with the non-addressable transient registers contained within the arithmetic section itself used in actual computation.

With the ADD to X and Add Negative to X instructions, the index registers also act as accumulators in the same manner as the arithmetic registers.

Partial Word Transfers

To minimize shifting and masking and to allow computation of selected portions of words, the 1108 System allows for the transfer of partial words into and out of the arithmetic section in a varying pattern. By selecting the coding of the j designator in the instruction word, a programmer may transfer a chosen portion of an operand to or from a control register. The transfer to an arithmetic register may also be accompanied by sign extension for subsequent arithmetic operations, depending on the j designator. (The Figure on page 29 displays the data paths for the various partial word instructions.)

Split Word Arithmetic

The UNIVAC 1108 System provides for split word addition and subtraction on whole words, half words or third words simultaneously. The right halves of two operands, for example, are added and the sum is stored in the right half of the selected accumulator. At the same time, the left halves of the same two operands are added and the result is stored in the left hand of the same accumulator. There is no carry interaction between the halves. The same holds true for thirds of words. Each partial word operates as an independent arithmetic register, with its own end around borrow.

Shifting

The UNIVAC 1108 System is capable of performing both single length shifting (36 bits) or double length shifting (72 bits), treating the latter as if operating with a single 72 bit register. A high speed shift matrix makes execution time independent of the number of places involved in the shift. An operand can be shifted from 0 to 72 positions in one storage cycle time.

A word, or two words to be shifted are displayed in a 72 × 72 shift matrix, each row of the matrix being offset or ''shifted'' one position from the row above it. The number of positions to be shifted merely indicates which row is to be chosen for the results.

Six types of shift operations are provided.

- Right Circular – bits shifted out at the right appear at the left.
- Left Circular – bits shifted out at the left reappear at the right.
- Right Logical – zeros replace bits shifted out of the most significant positions.
- Left Logical – zeros replace bits shifted out of the least significant positions.
- Right Arithmetic – sign bits replace bits shifted out of the most significant positions.
- Scale Factor Shift – a single or double accumulator left shift which positions the word and simultaneously counts number of shifts required until $A_{35} \neq A_{34}$.

Double Precision Fixed Point Arithmetic

The UNIVAC 1108 System provides full 72-bit, double precision fixed point addition and subtraction. Operands are processed as if they occupied a single 72 bit register. Bit 71, the high order bit, is considered the sign bit.

In addition, several arithmetic instructions produce two word results. With fixed point multiplication, a double length product is stored in two arithmetic registers for integer and fractional operations. Integer and fractional division is performed upon a double length dividend with the quotient retained in A and the remainder retained in A + 1.

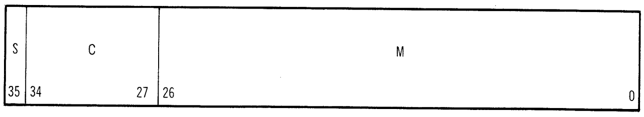
Floating Point Arithmetic

The UNIVAC 1108 System is equipped with an extensive, hardware repertoire of Floating Point Instructions, operating in single or double precision. Their data formats are as follows:

Single Precision with Partial Double Precision Result

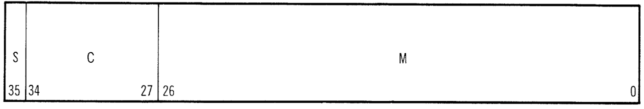
Range: 10^{+38}
10^{-37}
Precision: Eight Decimal Digits

SOURCE OPERAND FORMAT

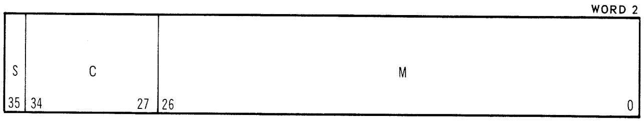


RESULT FORMAT

WORD 1



WORD 2



S = Sign
C = Characteristic
M = Mantissa

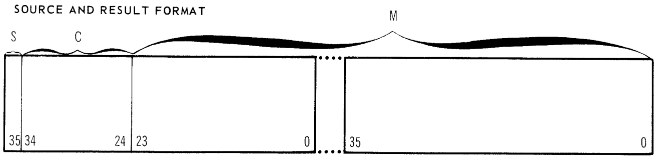
Word 1 is the significant portion of the result. Word 2 contains the unnormalized residue of the floating point operation. Mathematical error tracing can determine how much accuracy is being lost in calculations using this format. The residue word is displaced 27 bits to the right of the decimal point in the significant word – hence its characteristic is always adjusted by –27. The two word result of this single precision operation is developed in two contiguous Arithmetic Registers.

Full Double Precision

Range: 10^{+304}
10^{-303}

Precision: 18 Decimal Digits

SOURCE AND RESULT FORMAT



As noted above, full double precision operations do not require a repeated sign and exponent in the 36 least significant bits.

In any of the described floating point formats the characteristic can assume a range of values as follows:

Single precision (8 bits): 000 - 255
Double precision (11 bits): 0000 - 2047

To express negative characteristics, the 1108 hardware biases or floats the characteristic on a midvalue. The sign bit of the floating point word applies to the mantissa. The true and biased characteristic ranges are as follows:

	True	Biased
Single precision	–128 to +127	0 – 255
Double precision	–1024 to +1023	0 – 2047

A positive mantissa is normally assumed to be in range $1/2 \leq M < 1$. Such a value will place a "1" bit in the most significant bit position. When this condition exists, the floating point number is said to be normalized. A negative mantissa causes the entire floating point word to be complemented, and a "0" would appear in this position.

Floating point instructions are also provided for the following operations:

- Determining characteristic differences
- Packing and unpacking characteristics and mantissas (single and double precision)
- Conversion – single to double precision
  double to single precision

INPUT/OUTPUT CONTROL

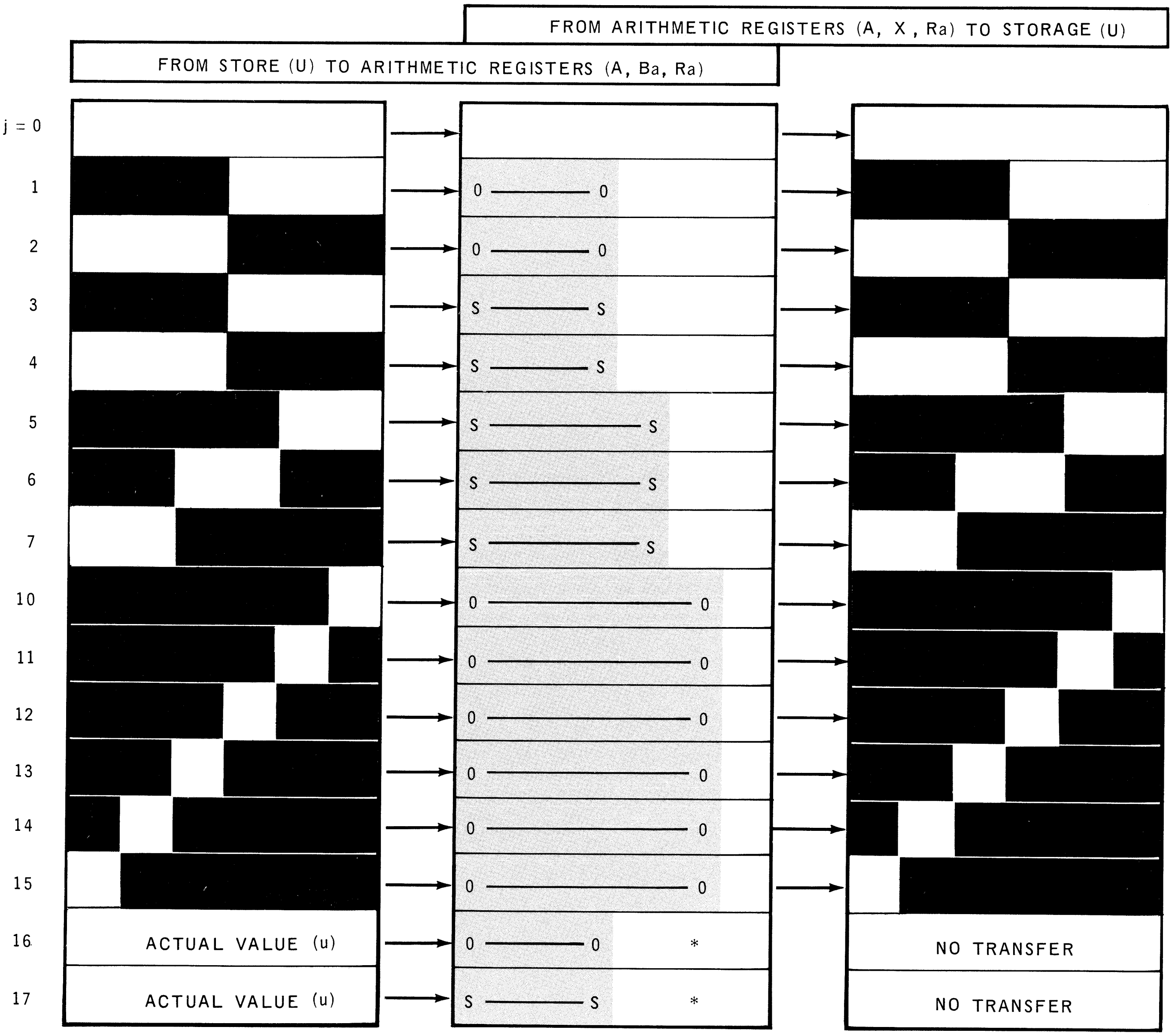
The input/output section of the UNIVAC 1108 System provides the data paths and control circuits necessary for direct communication between main storage and the peripheral equipment channels.

Communication between main storage and the peripheral units of the 1108 System may occur over 16 input/output channels. Each channel allows bidirectional transfers of data and control signals between storage and peripheral devices on that channel. Each channel contains 72 data lines (36 for input data and 36 for output data); plus additional lines which are used for transmitting control signals.

Standard peripheral subsystems utilize both the input and corresponding output lines of the same channel. Data transfers on these units (magnetic tape units, punched card units, mass storage devices, etc.) are bidirectional, but they do not occur at the same time. Data flows in only one direction over a single channel during any given interval.

The input/output section functions as a small processor. Programmed input/output transfers load index values into the Access Control Word locations and establish desired peripheral activity. From that point on, the I/O control section scans the input/output channels automatically, accepting data from the peripheral subsystems at the natural rate of the equipment. When a peripheral subsystem transmits or requests a word of data, its associated Access Control Word (described below) is referenced as an index and the I/O control section transfers the data word to or from main storage, updates the access control word and tests for a terminal condition. The actual transfer of data from the channel to main storage is accomplished at the end of the instruction cycle in parallel with extended arithmetic sequences. Access Control Word indexing and testing is performed in the indexing section in effectively zero-time, in parallel with normal instruction execution and indexing.

J-DETERMINED PARTIAL WORD OPERATIONS


FROM ARITHMETIC REGISTERS (A, X , Ra) TO STORAGE (U)
FROM STORE (U) TO ARITHMETIC REGISTERS (A, Ba, Ra)
j = 0
1
2
3
4
5
6
7
10
11
12
13
14
15
16
17
0 — 0
S — S
ACTUAL VALUE (u)
ACTUAL VALUE (u)
*
*
NO TRANSFER
NO TRANSFER

□ = Portion of word involved in the operation

■ = Portion of word not involved in the operation

▨ = Remainder of word in arithmetic register
0 – 0 = zero fill (may be plus or minus zero)
S – S = sign fill (sign of j-determined final contents of A)

* If x-designator = 0, the transferred value = h, i, u
If x-designator ≠ 0, the transferred value = u + $(X)_0$

When all sixteen input/output channels are operating concurrently, word transfers are multiplexed to provide an instantaneous communication rate of up to 1.4 million 36 bit words (8 million characters) per second.

Normal Input/Output Mode

Each channel operates in one of three states: *input, output* or *function.* The *input* and *output* states are employed when transferring data to or from the main storage. The *function* state is the means by which the central processor establishes the initial communication path with a peripheral subsystem, and is basically an output state. During this state of transmission, the central processor sends one or more function words to a peripheral subsystem. These function words direct the units to perform a desired operation.

Access Control Registers

The actual word by word transmission (regardless of the transfer state) over a given channel is governed by the access control registers in the Control Register Section. Two of these registers, one for input and one for output, are assigned to each of the sixteen channels.

Associated with these registers is an input/output access control word format. This word consists of three sections: G is the incrementation designator; W is the number of words to be transferred; and V is the initial central storage address to or from which data will be transferred.

Access Control Word Format

G	W	V
35 34	33 18	17 0

In initiating an input/output operation, the appropriate access control word is placed in the access control register corresponding to the channel associated with the designator of a load function in channel, load input channel, or load output channel instruction.

The V portion of this word contains the address to or from which the first word will be transferred. After each word is transferred, W is decremented by 1 and tested for zero. A zero terminates the transfer operation; non-zero causes the transfer of the next word in the data block. Depending upon the contents of G, subsequent words will be transferred to or from an initial main storage address that will: 1) ascend in value; 2) descend in value, or 3) not change. The bit configuration in G, along with operations they specify, are as follows:

G	OPERATION	NEXT ADDRESS
00	Increment V Address	V + 1
01	Inhibit Increment	V
10	Decrement V Address	V − 1
11	Inhibit Decrement	V

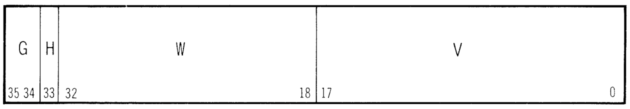
Externally Specified Index Mode (ESI)

The ESI feature in conjunction with UNIVAC data communication equipment allows multiple communications lines to automatically transfer characters to and from main storage on a self-controlled basis without disturbing the program sequence of the UNIVAC 1108 Processor. Externally Specified Index is provided for each of the sixteen input/output channels on the UNIVAC 1108 System.

When operating in the ESI mode, each communications line has its data flow governed by its own unique access control word. The ESI access control words are located in main storage to allow accommodation of many multiplexed communications lines. As each line transfers a character to or from main storage the address of its unique ESI access control word is presented to the input/output channel by the UNIVAC data communications equipment and is displayed on the upper 18 bits. Therefore no program monitoring is necessary to control the orderly flow of information from the communications lines.

Upon generation of an ESI interrupt, detection of full or empty buffer, the address of the ESI access control word of the terminating line is stored in the ESI status word for that channel. There are 16 ESI status words located in Control Registers 32-47.

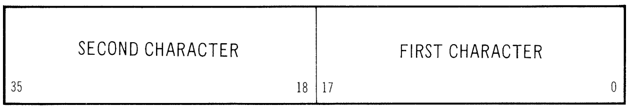
The ESI access control word has a format very similar to the normal I/O access control word. For ESI the format is:



V is the 18-bit data address
H is the half word counter
When H = 0, use the lower half of location specified by V
H = 1, use the upper half of location specified by V
W is the character count of message (15 bits)
G is the 2-bit field and is used as follows:
For G = 00, increment V by 1
G = 10, decrement V by 1
G = 01 or 11, do not change V

The alternation of V occurs once every two characters, and this is conditional on the state of H.

The format for ESI data words is two words (characters) per 36-bit word, arranged as follows:



The first character of an incoming message will cause the appropriate ESI access control word to be read out of memory. The H-bit will be found to be zero and so the character will be stored in the lower half of the location specified by V. Regardless of the setting of G, V will not be altered. The H-bit is set to one, W is decremented, and the control word stored back in memory. The second incoming character of this same message will again cause the ESI control word to be read out. This time, H = 1, so the character will be stored in the upper half of location V. This time, since H = 1, V will be incremented or decremented according to the setting of G. H is set back to zero, W is decremented, and the control word stored back in memory.

Priority Control Network

Channel operations are sequenced and synchronized by a priority control network within the input/output section of the UNIVAC 1108 Processor.

Although all sixteen channels may be available for data transmissions between the central processor and peripheral units at the same time, only one channel will actually be logically connected with the central processor during a given interval.

Priority control circuits, along with the access control circuits, resolve situations where two or more channels simultaneously attempt to communicate with the central processor. Access to the UNIVAC 1108 Processor is granted on the basis of the following priority schedule.

1. *Output or Function (Alternate between Input and Output)*
2. *Input (Alternate between Input and Output)*
3. *Real Time Clock Decrement*
4. *Real Time Clock Interrupt*
5. *External Interrupt (ESI)*
6. *Input Monitor (ESI) (From Indexing Section)*
7. *Output Monitor (ESI) (From Indexing Section)*
8. *External Interrupt (Not ESI)*
9. *Input Monitor (Not ESI)*
10. *Output Monitor or Function (Not ESI)*
11. *Interprocessor Interrupt #1*
12. *Interprocessor Interrupt #2*

When two or more channels simultaneously request the central processor to process operations occupying the same priority level, access is granted to the lowest numbered channel. When both an input and an output data request are present (items 1 and 2 in the schedule), priority will alternate, i.e., process one input data word, alternate and process one output data word, alternate and process one input word, etc.

Simultaneous input and output monitor interrupts are handled in a manner similar to that employed for simultaneous input and output data requests. An operation of higher priority, occurring while the processing of monitor interrupts is being alternated will replace the next scheduled operation and then alternate to the opposite of that which was replaced.

Input/Output – Peripheral Control

The input/output section of the UNIVAC 1108 Processor can accommodate a wide range of peripheral subsystems. These include magnetic drum and magnetic tape devices, card punches and readers, printers, and other computer systems. In addition to these, the UNIVAC 1108

Processor can also communicate with many real time input/output devices through the UNIVAC Standard Communications Subsystem.

A standard peripheral subsystem, whether it be card, tape or drum, consists of one or more units of the same type, a control unit for the particular type of equipment and a Channel Synchronizer.

Input/Output Instructions

This group of instructions allows the program (usually the Executive) to initiate and control input/output operations.

Load Input Channel	(75, 00)
Load Input Channel and Monitor	(75, 01)
Jump on Input Channel Busy	(75, 02)
Disconnect Input Channel	(75, 03)
Load Output Channel	(75, 04)
Load Channel and Monitor	(75, 05)
Jump on Output Channel Busy	(75, 06)
Disconnect Output Channel	(75, 07)
Load Function in Channel	(75, 10)
Load Function in Channel and Monitor	(75, 11)
Jump on Function in Channel	(75, 12)
Allow Function in Channel	(75, 13)
Allow All Channel Interrupts	(75, 14)
Prevent All Channel Interrupts	(75, 15)
Allow Channel Interrupt	(75, 16)
Prevent Channel Interrupt	(75, 17)

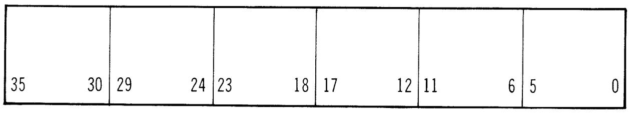
Information Words

There are four types of information words which are transmitted through the data lines connecting the peripheral subsystem to the central processor.

- Data Words
- Function Words
- Identifier Words
- Status Words

DATA WORDS

A Data Word consists of 36 bits (or six 6 bit characters).



The contents of bit positions 35 through 30 contain the most significant character, while the contents of bit positions 05 through 00 constitute the least significant character. Data is transferred on a word by word parallel basis, an Input Data Request signal is generated, signifying that a Data Word is on the 36 input data lines. After the central processor has accepted the Data Word, it generates an Input Acknowledge signal. During output operations, the peripheral control unit generates an Output Data Request signal which informs the central processor of the unit's availability. After placing the output data word on the data lines, the UNIVAC 1108 Processor generates an Output Acknowledge signal, signalling the peripheral control unit that the 36 data lines contain a Data Word.

Upon accepting this word, the peripheral control unit will generate another Output Data Request signal.

FUNCTION WORDS

The Function Word contains the operating Instructions for the peripheral units. A Function Word, like a Data Word, consists of six 6 bit characters. The Channel Synchronizer, upon receiving a Function Word, interprets the Function Code. If more than a single unit is serviced on the same peripheral subsystem, the Function Code will include a unit-select code.

A Function Word is sent from the UNIVAC 1108 Processor to the peripheral subsystem in the same manner as output data words. To distinguish between the two, the processor generates an External Function signal after placing the Function Word on the 36 data lines. This signal is transmitted over the external function control line. Upon receiving the signal, the peripheral subsystem is altered to receive a Function Word.

STATUS WORDS

The Status Word contains any error information which may be generated by the peripheral control unit and the Channel Synchronizer. The Status Word is 36 bits and is transmitted over the data lines. It is always generated by a peripheral subsystem when it sends an interrupt. The entire Status Word is transferred to main storage address 152, the External Status Word location.

The Status Word is sent to the central processor in the same manner as the input data word, except that an External Interrupt signal is generated after the Channel Synchronizer has placed the word on the input data lines. In this way, the central processor can distinguish Status Words from input data words.

IDENTIFIER WORDS

The Identifier Word is a full length computer word which is associated with a Search or a Search-Read Function. In a Search or Search-Read Function, the Identifier Word is stored in a special register in the Channel Synchronizer and compared with each word read by the peripheral unit.

Monitored Instructions

As shown in the input/output instruction repertoire instructions calling for input, output, or function transfers may be executed either with or without monitor. When executed with monitor, upon completion of the transfer an internal interrupt will be generated. When an instruction is executed without a monitor the interrupt is inhibited.

INPUT/OUTPUT CONTROLLER

The Input/Output Controller is an independent, wired logic processor which provides:

- Independent data paths between peripheral subsystems and main storage
- High speed communications capability
- Enhanced systems performance through chained buffer operations
- The ability to expand the number of input/output channels available to the 1108 System

The Input/Output Controller (I/OC) is a free standing unit with subsystem channel options of four, eight, twelve, or sixteen high speed bidirectional data channels. Each channel has an instantaneous transfer rate of 1,500,000 characters per second. The I/OC has a transfer rate capability of 8,000,000 characters per second, obtained by priority time-sharing of requests from the sixteen input/output channels. The data paths of the I/OC provide transfers between peripheral subsystems and main storage independent of the cyclic operation of the 1108 Central Processors. Up to three 1108 Central Processors may control a single I/OC.

Operations are initiated from an 1108 Central Processor by sending functional commands to the I/OC via the normal CPU channel interface. The I/OC accepts these commands and then sequences the independent data transfer.

The I/OC is functionally similar to the input/output section of the 1108 Central Processor, a basic deviation being that rather than using the I/O Access Control Registers of the Central Processor to control peripheral operation, the I/OC contains its own high speed Index Memory for buffer control. The basic Index Memory provides 256 thirty-six bit words, with an optional 256 word module providing expanded ESI capabilities. An additional sixteen 5-bit Associative Registers are used to associate each of the sixteen input/output channels with a particular Index Memory location. In the Non-Chain mode of operation, a single Index Memory location is used to define the Buffer Control Word. When in the Chain mode, the Associative Register is used to indicate the first of a series of Buffer Control Words in Index Memory. Termination of an operation in the Chain mode is effected by recognition of an End-of-Chain code in Index Memory. The 1108 Central Processors may both load and store information in the Index Memory and associative Registers of the I/OC.

INDEX MEMORY ALLOCATION

ADDRESS	ASSIGNMENT
0 – 15	EXTERNAL FUNCTION CONTROL WORDS
16 – 63	NORMAL BUFFER CONTROL WORDS
64 – 255	ESI BUFFER CONTROL WORDS
256– 511	ADDITIONAL ESI BUFFER CONTROL WORDS

Since the Input/Output Controller is capable of accessing main storage independently of the Central Processor, main storage parity validation is also performed by the I/OC. Should a parity error be detected by the I/OC, the CPU is interrupted and a status code transmitted indentifying the unique main storage module or Index Memory module in which the parity error occurred.

6. PERIPHERAL SUBSYSTEMS

Peripheral Subsystems are attached to the 1108 or to the independent I/O controller through general purpose input/output channels, associated with each. Because of the general purpose nature of these channels there is no restriction in the way that peripheral subsystems may be attached to them. The governing factor for peripheral attachment is the transfer rate of the devices in the subsystem. Since the channels are priority ordered, very high transfer rate or real time equipment are attached to the lowest numbered channels.

With this adaptable input/output technique the UNIVAC 1108 can communicate with many real time devices such as analog/digital, key sets, communication terminals, tracking and radar systems, display systems, and other information processing systems.

The standard UNIVAC 1108 peripheral subsystems are:

High Performance Drums

FH 432 Magnetic drum
FH 1782 Magnetic drum

Mass Storage

Modular FASTRAND
FASTRAND II

Magnetic Tape Systems

UNISERVO VIC
UNISERVO VIIIC

Auxiliary Systems

Punch Card Subsystem (Reader/Punch)
Multiple High Speed Printer Subsystem
UNIVAC 1004 Online Subsystem

Satelite Systems

UNIVAC 1004 Remote
UNIVAC 418

In addition to the standard UNIVAC 1108 Subsystems, UNIVAC 1107 Subsystems will function with the 1108 as well. Included in this category are:

UNISERVO IIA
UNISERVO IIIA
UNISERVO IIIC
UNISERVO IVC
Paper Tape Subsystem
FASTRAND I
FH-880

THE 'FH' SERIES OF MAGNETIC DRUMS

The UNIVAC Flying Head (FH) series of high speed large capacity magnetic drum storage units provide modular auxiliary storage essential for the operation of large and complex systems. These units vary from the ultra fast FH 432 (with an average access time of 4.3 milliseconds) to the large capacity (12.5 million alphanumeric characters) FH 1782 which provides extensive fast access storage that can be used for large data files that have to be referenced frequently.

''FH'' Magnetic Drum subsystems have an individual read-write head for each track.

Any word of an FH series drum is available to the system in an average access time of half (revolution – 4.3 milliseconds (FH 432) and 17.0 milliseconds (FH 1782).

Every word in all FH subsystems is individually addressable so that the fullest use can be made of premium storage. Offline search operations can be initiated by which the Control Unit/Synchronizer matches the contents of any drum area, up to the entire size of the subsystem, with a designated Identifier Word, and will, upon finding a match commence reading to Main Storage. This entire process is carried out offline without any Processor attention once the Input/Output search function has been initiated and the Identifier Word designated. This feature is frequently used in the scanning of large data tables when the exact location of an item is unknown.

A new function has been incorporated in the control logic of the FH 432 and FH 1782 subsystems to predict and reduce storage access time. This function enables the program to request the current storage level under the read-write heads on a particular drum unit. The Input/Output handler can then select from the subsystem queue the data request that can be serviced fastest. It is possible to sequence drum requests and to make multiple accesses to a drum unit during a single revolution, instead of having to wait an average of half a revolution for each request.

The transfer rate of data to and from the FH drum subsystems is in line with the ultra fast computing power available. The standard rate is 1,440,000 alphanumeric characters per second. By altering a patchboard, drum transfer rates may be matched to system loads by *interlacing* to provide transfer rates of 720,000; 360,000; 180,000; or 90,000 alphanumeric characters per second.

Through the addition of Multi-Processor Adapters, a Single or Dual-channel FH drum subsystem may be accessed, by multiple processors. This not only permits safeguards in case of failure but also enables all processors to access all drum units as common storage, so that multi-processors can share a single major task, or tasks can be allocated to individual Processors but with data storage being shared.

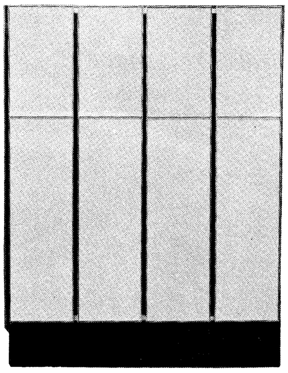
With all FH drum subsystems either one or two Control Unit/Synchronizers are available, using one or two input/output channels. The dual channel capability permits Read/Read, Read/Write, Write/Read and Write/Write operations simultaneously on any two drum units of the subsystem. If required, a Search function may by substituted for any of the Read functions. As an additional reliability measure, each Control Unit/Synchronizer of a Dual Channel CU/S has its own power supply. Thus, in case of a failure of one of the power supplies, the subsystem can still operate on a single channel basis.

The philosophy of the UNIVAC 1108 Operating System is to use drums instead of magnetic tapes as much as possible – this reduces manual handling and access and transfer time when compiling and assembling, and during basic batch input/output operations.

These magnetic drum subsystems have many advantages in standard data processing as well as real time operation. This is especially true in applications where rapid file processing and sort/merge routines are more prevalent. Large capacity with rapid accessibility provides convenient intermediate storage. Instead of multiple tape units the use of the drum subsystems frees the tape units for primary input/output demands.

Drum subsystems allow an extensive Executive Control System without undue Main Storage utilization or operating inefficiency. The unequaled access time of the FH 432 drum, combined with its time saving offline search feature, permits lesser used control segments to be stored outside of Main Storage, and read into a common 'overlay' area only when required. This arrangement greatly reduces the size of expensive Main Storage required for the Executive System.

FLYING HEAD-432 MAGNETIC DRUM



A minimum FH 432 Magnetic Drum Subsystem consists of 786,432 36-words of data storage (three drums), a control unit/synchronizer, and power supplies for this storage packaged in two cabinets. Each cabinet after this has the capability of containing 524,288 36-bit words of storage (two additional drums) but is modular by drum for system purposes. Of the 432 tracks on each drum 384 are used for data purposes; the remaining tracks are used for spares, parity and timing functions. There are 2,048 words of data per 3 tracks. Reading and writing are bit-serial and staggered from 3 tracks simultaneously at a transfer rate of 80,000 words per second per track, providing a maximum transfer rate of 240,000 words or 1,440,000 characters per second.

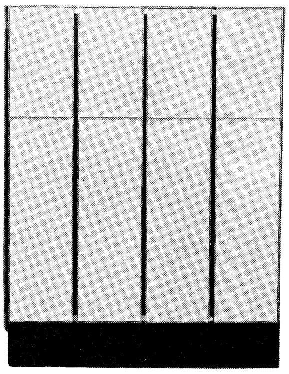
Up to eight FH 432 Magnetic Drum units may be accommodated in a single subsystem, using single or dual Control Unit/Synchronizers, and by means of a Multi-Processor Adapter, being attached to up to four Processors. Maximum subsystem capacity is 2,097,152 words or 12,582,912 alphanumeric characters.

FH 432 units may be intermixed with FH 1782 units in the same subsystem to provide a powerful blend of ultra high speed and large capacity storage. This mixed subsystem is described in a later section.

CHARACTERISTICS

STORAGE CAPACITY	262,144 computer words of 36 data bits plus parity bits or 1,572,864 alpahnumeric characters per drum
AVERAGE ACCESS TIME	4.25 milliseconds
DRUM SPEED	7,200 revolutions per minute
NUMBER OF READ/WRITE HEADS	432 – one per track
CHARACTER TRANSFER RATE (MAX.)	1,440,000, 720,000, 360,000, 180,000, 90,000
WORD TRANSFER RATE (MAX.)	240,000, 120,000, 60,000, 30,000, 15,000
I/O CHANNELS REQUIRED	1 or 2 per subsystem
NUMBER OF DRUMS PER SUBSYSTEM	8 (total of 12,582,912 chars.)

FH 1782 MAGNETIC DRUM SUBSYSTEM



The Flying Head 1782 (FH 1782) magnetic drum is identical to the FH 880 drum but with a two and two-thirds greater storage capacity achieved partly by an increase in the number of tracks (to 1,760) and partly by an increase in the recording density (to 691 bits per inch). Each track has its own read/write head and average access time is unchanged at 17 milliseconds.

A single FH 1782 drum will hold 2,097,152 computer words, equivalent to 12,582,912 alphanumeric characters. Up to eight FH 1782 drums can be accommodated in a single subsystem giving a subsystem capacity of 100,663,296 characters.

As a result of the increased storage capacity available, the character transfer rates have been increased to be identical with those of the FH 432 drum; this arrangement enables FH 1782 drums to be associated on the same subsystem with FH 432 drums, as described in the next section.

CHARACTERISTICS

STORAGE CAPACITY	2,097,152 computer words of 36 data plus parity bits or 12,582,912 alphanumeric characters per drum.
AVERAGE ACCESS TIME	17 milliseconds
RECORDING DENSITY	691 bits per inch
DRUM SPEED	1,800 revolutions per minute
NUMBER OF READ/WRITE HEADS	1760 (40 blocks with 49 heads per block)
CHARACTER TRANSFER RATE	1,440,000; 720; 360,000; 180,000; 90,000
WORD TRANSFER RATE	240,000; 120,000; 60,000; 30,000; 15,000
I/O CHANNELS REQUIRED	1 or 2 per subsystem
NUMBER OF DRUMS PER SUBSYSTEM	8 (total of 100,663,296 chars)

FH 432/FH 1782 MAGNETIC DRUM SUBSYSTEM

A valuable option with UNIVAC 1108 drum subsystems is the ability to associate in the same subsystem the ultra high speed FH 432 drum with the fast high capacity FH 1782 drum. Any combination of eight drums may be mixed on a subsystem.

This subsystem arrangement is of significant importance in the UNIVAC 1108 Storage Configuration. An efficient blend can be made of high speed storage for rapidly required software, program segments, tables and indices, and slower speed but large capacity storage for less frequently .used program segments, data files, and message assembly/disassembly areas. A judicious mix of speed, capacity and economy can be planned, and a mix that can readily be altered as requirements change. Character transfer rates are identical for the FH 432 and FH 1782 drum units, the only functional difference in a data transfer is the variation in access time.

FASTRAND MASS STORAGE SUBSYSTEMS

FASTRAND Mass Storage subsystems provide very large capacity random access storage, and are available in two models each with different speed and storage capabilities, thus permitting the use of a hierarchy of random access storage units so that an exact blend of speed, size and cost can be selected to satisfy any random access requirements. Even greater flexibility is enabled by the ability to mix the various types of FASTRAND units on a single or dual channel subsystem, and by the FASTBAND option, available on a unit basis, which provides fixed read/write heads on some tracks giving rapid access to these areas of storage.

FASTRAND units use large magnetic drums, one with MODULAR FASTRAND, two with FASTRAND II. These drums are similar in concept to those used in the FH 432 and FH 1782 subsystems, in that they employ flying heads. However, to reduce cost, only a limited number of read/write heads are used which move laterally over many recording tracks.

Modular FASTRAND, apart from using a single drum which is smaller, permits a faster revolution speed, and contains one twelfth of the storage of a FASTRAND II Unit. In addition, optional features on both units include 24 (16 on Modular FASTRAND) additional tracks with fixed read/write heads (FASTBANDS) for rapid access (35 milliseconds average access, 17.5 milliseconds for Modular FASTRAND), and an incremental drum write lockout feature for data protection.

The design features which combine to give a FASTRAND SUBSYSTEM an unprecedented high operating speed among those devices in the mass storage family are short access time, a high rate of data transfer, and an offline search operation.

There are 64 read/write heads (16 on Modular FASTRAND) per drum unit with all heads connected to a common positioning mechanism. As a result, the subsystem positions all of the heads in a drum unit with one movement of its positioning mechanism in an average time of 57 milliseconds (50 ms. on Modular FASTRAND). The maximum head positioning time is 86 milliseconds (79 milliseconds on Modular FASTRAND) over all the tracks, and the minimum is 30 milliseconds. Average latency is half a drum revolution time, is 35 milliseconds on FASTRAND II, and 17.5 milliseconds on Modular FASTRAND. Average access, therefore, varies from 35 milliseconds for data in tracks currently under the heads, to 156 milliseconds for maximum head movement, with an average of 92 milliseconds. These times are for FASTRAND II units, figures for Modular FASTRAND are 17.5 to 114 maximum, giving an average overall access time of 67.5 milliseconds. These average times can invariably be reduced by good systems design, data layout and programming.

Additionally, the FASTRAND subsystem has an independent position control feature which allows greater flexibility, and can be used to decrease the average access time. This is done in a multi-unit subsystem by concurrently pre-positioning the heads in each Drum Unit. Pre-positioning the heads saves time, because once the position instructions have been transmitted to the Drum Units, the computer can immediately resume operation on the main routine. The computer does not have to wait for the positioning operation to be completed. Then, whenever specifically instructed to read or write from a pre-positioned unit, the only time delay is for the latency and address circuit activation. These are the mechanical design features which contribute to the FASTRAND'S operating speed.

If the circumstances are such that the head positioning time (due to pre-positioning) and the latency are both zero, then the minimum access time is 0.8 milliseconds. This time interval allows certain addressing circuits to be activated. Normally, such activation starts simultaneously with the other processes involved; in the best case situation described above, it is the only delay. In the case of mean (92 or 67.5 milliseconds) or maximum access time (156 or 102 milliseconds), the addressing circuit time need not be considered because it is overlapped by other time factors.

In any effort to reduce processing time, "offline search" represents an important advantage. This is the operation in which the computer instructs the FASTRAND subsystem to locate a specific piece of data, and then goes on with the processing while the storage search takes place. Then, at the termination of this variable length search the computer is notified, and is given the results. The computer may take action as provided in the program.

Also, all other functions of the FASTRAND are buffered so that the computer may continue its work while records are being read from or written on the drums.

The FASTRAND subsystem, therefore, presents a balanced combination of the two primary data storage requirements: capacity and speed.

All data on a FASTRAND subsystem is recorded with word and sector parity. These parity bits are automatically checked upon writing and reading.

A single FASTRAND subsystem can accommodate any mixture of up to eight FASTRAND II and MODULAR FASTRAND units. Any of the Input/Output channels of a UNIVAC 1108 Processor and an Input/Output Controller can accommodate a FASTRAND subsystem, and as with the 'FH' series of drums, either single or dual channel operation is possible, providing full scale two drum unit simultaneity, with access to the subsystem by one to four Processors or Input/Output Controllers.

The MODULAR FASTRAND Mass Storage subsystem is a medium capacity random access mass storage device. A subsystem contains from one to eight MODULAR FASTRAND mass storage units.

Each MODULAR FASTRAND Mass Storage Unit in the subsystem contains one magnetic drum. This drum is similar to those used in the Flying Head-880 Magnetic Drum Subsystem in that they employ flying head read/write drums. However, the read/write heads on these drums are moved laterally over many recording tracks.

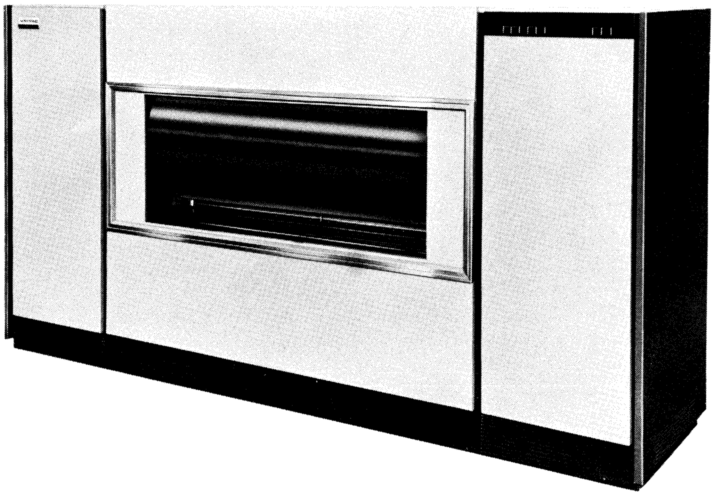
Optional features include 16 additional tracks with fixed read/write heads (FASTBANDS) for rapid access (17.5 ms. average) and an incremental drum write lockout feature for data protection.

The 16 Read-Write heads per unit are connected to a common positioning mechanism, the subsystem positions all the heads in a unit with one movement of its mechanism in an average time of 50 milliseconds. The maximum head-positioning time is 79 milliseconds, the minimum 30 milliseconds.

The MODULAR FASTRAND subsystem has an independent positioning control allowing each unit multi-unit subsystem to simultaneously preposition its heads. The only time delay is for latency and address circuit activation.

The MODULAR FASTRAND magnetic drum rotates at 1,760 RPM, with maximum latency of 35 milliseconds.

FASTRAND* II MASS STORAGE UNIT



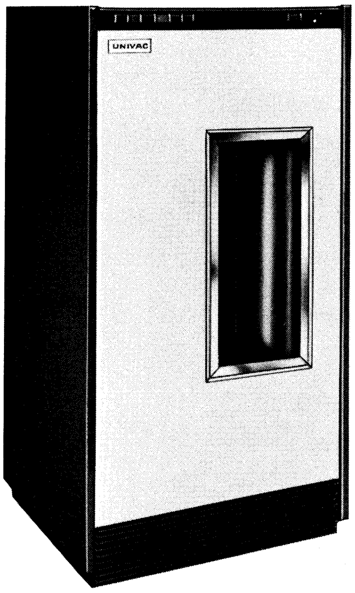
CHARACTERISTICS	
STORAGE CAPACITY (PER UNIT)	22,020,096 word or 132,120,576 alphanumeric characters
AVERAGE ACCESS TIME	92 milliseconds
RECORDING DENSITY	1,000 bits per inch
TRACKS PER INCH	106
DRUM SPEED	870 revolutions per minute
MOVEABLE READ/WRITE HEADS	64
CHARACTER TRANSFER RATE	153,750 characters per second
WORD TRANSFER RATE	25,625 words per second
**FASTBANDS (FIXED READ/ WRITE HEADS)	24
FASTBAND AVERAGE ACCESS TIME	35 milliseconds
FASTBAND STORAGE CAPACITY (PER UNIT)	258,048 characters
**WRITE LOCKOUT PROTECTION	Yes
I/O CHANNELS	1 or 2 per subsystem
NO. OF UNITS PER SUBSYSTEM	8 (1,056,964,608 characters per subsystem)

** Trademark of Sperry Rand Corporation*

***Optional*

NOTE: FASTBAND Storage is additional to standard unit storage.

MODULAR FASTRAND* UNIT



CHARACTERISTICS

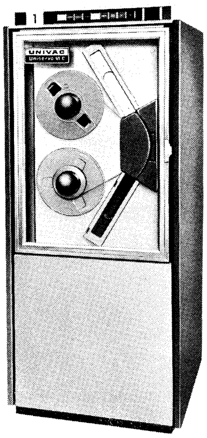
STORAGE CAPACITY (PER UNIT)	1,835,008 words or 11,010,048 alphanumeric characters
AVERAGE ACCESS TIME	67.5 milliseconds
RECORDING DENSITY	1,000 bits per inch
DRUM SPEED	1,760 revolutions per minute
MOVEABLE READ/WRITE HEADS	16
CHARACTER TRANSFER RATE	153,750 characters per second
WORD TRANSFER RATE	25,625 words per second
**FASTBANDS (FIXED READ/ WRITE HEADS)	16
FASTBAND AVERAGE ACCESS TIME	17.5 milliseconds
FASTBAND STORAGE CAPACITY (PER UNIT)	86,016 characters
**WRITE LOCKOUT PROTECTION	Yes
I/O CHANNELS REQUIRED	1 or 2 per subsystem
NO. OF UNITS PER SUBSYSTEM	8 (88,080,384) characters per subsystem)

**Trademark of Sperry Rand Corporation*

***Optional*

NOTE: FASTBAND Storage is additional to standard unit storage.

UNISERVO* VIC MAGNETIC TAPE UNIT



The UNISERVO VIC is a low cost unit with moderate speed and transfer rates for applications which do not require massive file passing, extensive sorting, or other requirements that dictate the acquisition of high speed magnetic tape subsystems.

A UNISERVO VIC subsystem can accommodate up to 16 magnetic tape units connected to one or two input/output channels. Dual channel operations permit any two input/output functions, in addition to rewind, to be performed simultaneously on any two magnetic tape units.

The Master/Slave concept is employed in the logic of the UNISERVO VIC Subsystem. That is, circuitry has been built into one of the UNISERVO VIC units which will allow it to govern up to three other UNISERVO VIC units for certain electronic control functions. In a maximum subsystem of 16 units, there could be four "master" units and twelve "slaves."

Data packing density is selectable at either 200, 556, or 800 characters per inch. The physical tape passing speed is 42.7 inches per second, giving maximum transfer rates of 8,540, 23,741 and 34,160 characters per second respectively. Rewind takes place at 160 inches per second, enabling a full reel of 2,400 feet to be rewound in 180 seconds. The 800 character per inch packing density will normally be used, the 200 and 556 densities being used only for compatibility purposes. At this density more than 11,520, 000 characters may be stored on a single reel with data held as 600 character blocks.

Data may be recorded in variable length blocks under program control, with character and block (i.e., horizontal and vertical) parity. A read after write head allows immediate verification of all data written, and under the control of the software Input/Output Handler, repeated read and write operations are automatically undertaken whenever read or write errors occur.

Programming problems are insignificant since the circuitry of the UNISERVO VIC Synchronizer, combined with the software Input/Output Handler, deals with all operations except the writing of the input/output instruction and the system response to a non-recoverable error.

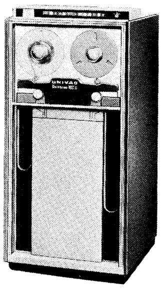
UNISERVO VIC Magnetic Tape units are fully compatible with IBM 727, 729 Model I through VI, and 7330 units in seven channel mode, and with IBM 2400 Series Models 1 through 3 units in nine channel mode, and with industry compatible units produced by other manufacturers. The UNISERVO VIC synchronizer is fitted with a hardware Translator that converts to and from binary coded decimal and Fieldata code, thus assuring tape compatibility between installations.

* *Trademark of Sperry Rand Corporation*

UNISERVO VIC Magnetic Tape Unit

CHARACTERISTICS	
TRANSFER RATE	8,500/23,700 and 34,200 chars. per second. Optional 8,500/23,700 and 34,200 bytes per second
RECORDING DENSITY	200,556 and 800 6-bit chars (optional bytes) per inch
TAPE SPEED	42.7 inches per second
TAPE WIDTH	0.5 inch
TAPE LENGTH	2,400 feet
THICKNESS	1.5 mils
BLOCK LENGTH	Variable
SPACE BETWEEN BLOCK	0.75 inch (7 channel mode) 0.60 inch (9 channel mode)
CHANNEL ON TAPE	7 channels, 6 data, 1 parity Optional 9 channels, 8 data, 1 parity
MAXIMUM NUMBER OF UNITS IN SUBSYSTEM	16
SPECIAL FEATURE	Backward read
PROCESSOR INPUT/OUTPUT CHANNELS	1 or 2

UNISERVO VIIIC MAGNETIC TAPE UNIT



A UNISERVO VIIIC subsystem will accommodate up to 16 magnetic tape units, and may incorporate either one or two Control Unit/Synchronizers attached to one or two input/output channels. Dual channel operation permits any two input/output functions, including write/write, in addition to rewind, to be performed simultaneously on any two magnetic tape units, thus effectively doubling maximum subsystem transfer rates. With the addition of a Multiple Processor Adapter up to four Processors may access the subsystem under program control.

UNISERVO VIIIC Units may be specified with seven or nine channel mode. In seven channel mode one parity and six data bits are recorded in each frame across the width of the tape. A single 6 bit alphanumeric character, or a 6 bit binary value may be stored per frame. In nine channel mode one parity and eight data bits are recorded in each frame across the width of the tape, permitting either a single 6 bit alphanumeric character, two 4 bit binary values to be stored per frame.

Data packing density is set by program and manual switch on each unit to either 200, 556 or 800 frames per inch. Physical tape speed is 120 inches per second, giving maximum transfer rate of 24,000, 66,720 and 96,000 characters per second (in seven channel mode), or bytes per second (in nine channel mode). Rewind takes place at 360 inches per second, enabling a full reel of 2,400 feet to be rewound in 80 seconds. The 800 frame per inch packing density will normally be used, the 200 and 556 densities being used only for compatibility purposes.

Reading may take place with the tape moving either forward or backward – the latter feature being valuable in often obviating rewind time especially during sort/merge operations. Writing takes place when the tape is moving in a forward direction only.

Data may be recorded in variable length blocks under program control, with character and block (i.e. horizontal and vertical) parity. A read after write head allows immediate verification of all data written, and under the control of the software Input/Output Handler, repeated read and write operations are automatically undertaken whenever read or write errors occur in an attempt to recover.

Whole or partial word search is effectively conducted off-line – once the search instruction has been transferred to the Magnetic Tape Control Unit Synchronizer the Processor is not used at all until the search is completed, and its success or failure indicated by an interrupt.

Programming problems with this tape subsystem are insignificant since the circuitry of the Control Unit/Synchronizer, combined with the software Input/Output Handler, deals with all operations except the request for an input/output function and the system response to a non-recoverable error.

UNISERVO VIIIC Magnetic Tape Units are fully compatible with IBM 727, 729 Model I through VI, and 7330 units in seven channel mode, and with IBM 2400 series Models 1 through 3 units in nine channel mode, and with industry compatible units produced by other manufacturers. The UNISERVO VIIIC synchronizer is fitted with a hardware Translator that converts to and from binary coded decimal and FIELDATA code, thus easing tape compatibility problems between installations.

UNISERVO VIIIC Magnetic Tape Unit

CHARACTERISTICS	
TRANSFER RATE	24,000, 66,720 and 96,000 characters per second Optional 24,000, 66,720 and 96,000 bytes per second
RECORDING DENSITY	200,556 and 800 chars. (bytes optional) per inch
TAPE SPEED	120 inches per second
TAPE WIDTH	0.5 inch
TAPE LENGTH	2,400 feet (plastic)
THICKNESS	1.5 mils.
BLOCK LENGTH	Variable
SPACE BETWEEN BLOCK	0.75 inch (7 channel) 0.6 inch (9 channel)
CHANNELS ON TAPE	7 channels, 6 data, 1 parity Optional 9 channels, 8 data, 1 parity
UNITS PER CONTROL	16
SPECIAL FEATURE	Backward Read
PROCESSOR INPUT/OUTPUT CHANNELS	1 or 2

UNISERVO VIC/VIIIC SUBSYSTEM

The UNISERVO VIC/VIIIC Subsystem permits any combination of UNISERVO VIC and UNISERVO VIIIC Magnetic Tape Units, up to a maximum of 16, to be intermixed on a single or dual input/output channel A Multiple Processor Adapter may be attached to permit access from one to four Processors.

The UNISERVO VIC and UNISERVO VIIIC units in a mixed subsystem may have 7 to 9 channel capability in any unit combination. The only restriction is that ''Slave'' UNISERVO VIC units must have the same channel capability as their associated ''Master'' unit.

This mixed magnetic tape subsystem enables an infinitely flexible blend of high speed and economic medium speed tape units to be used in conjunction. In addition the 7 and 9 channel unit options permits both data recorded in traditional industry compatible form to be handled, and yet at the same time allows the upgrading of these records in line with the ASCII code and faster decimal handling rates.

UNIVAC 1004 SUBSYSTEM

The UNIVAC 1004 is a powerful processing unit in its own right, with arithmetic, logical, and editing capabilities allied to a modular core storage. Standard peripheral units are a 400 cpm or 615 cpm Card Reader, and a High Speed Printer, operating at 400 lpm or 600 lpm with a 63 character set and a 132 character print line width.

Optional units are a second bank of 961 characters of core storage (with a character cycle time of 8 or 6.5 microseconds), a second Card Reader (400 cpm), a Card Punch or Card Read Punch (200 cpm), Paper Tape Reader (400 cps), Paper Tape Punch (110 cps), 1 or 2 UNISERVO VIC Magnetic Tape Units, and a communications facility utilizing one of the range of Data Line Terminals.

A 1004 can be connected on-line to a single input/output channel of a UNIVAC 1108 System to provide Card Reading, Card Punching, and Printing capability. Any combination of 1004 peripheral units may be used with the 1108 System and many units may be functioning on-line at the same time.

The 1004 retains its innate processing power when in the 1108 configuration, and this may be utilized in conjunction with the 1108 internal program. At any time the 1004 may readily be switched to off-line mode, and will then operate as a standard 1004.

Peripheral operations such as card-to-tape, tape-to-card and tape-to-print can be carried out using the UNISERVO VIIIC subsystem and the 1004. If required one or two compatible UNISERVO VIC units may be attached to the 1004 providing a powerful on or off-line satellite system for large scale data inscription and print-out.

UNIVAC 1004 SUBSYSTEM

CHARACTERISTICS	
CARD READING SPEED	400 or 615 cpm
CARD PUNCHING SPEED	200 cpm
PRINTING SPEED	400 or 600 lpm
PRINTABLE CHARACTERS PER LINE	63 plus space
NUMBER OF CHARACTERS PER LINE	132
NUMBER OF LINES PER INCH	6, 8, or 10 optional
MAGNETIC TAPE	1 or 2 of UNISERVO VIC Units
IBM COMPATIBILITY	Yes – any 6 bit code compatibility – program controlled odd or even parity check
MAGNETIC TAPE CODE CONVERSION	Available in 1004 using the automatic translate feature
TAPE TRANSFER RATE	8,500; 23,700 and 34,200 characters per second
RECORDING DENSITY	200,556 or 800 ppi
PAPER TAPE READ	400 ch.p.s.
PAPER TAPE PUNCH	110 ch.p.s.
COMMUNICATION ABILITY	Data Line Terminals
NUMBER OF INPUT/OUTPUT CHANNELS USED	1

THE UNIVAC 1004 AS A REMOTE DATA PROCESSOR

The UNIVAC 1004 may function as a remote data processor connected via a communications line to the UNIVAC 1108 system. This is achieved by attaching one of a range of Data Line Terminals (DLTs) to the 1004 which will permit transmission at speeds of 1200, 2000, 2400 or 40,800 bauds, dependent upon the type of DLT and communications facility employed. The interface with the UNIVAC 1108 system is via either the 1108 Communications Terminal Module Controller (CTMC) (for 1200, 2000 or 2400 bauds) or a Communications or Word Terminal Synchronous (CTS or WTS) for 40,800 bauds.

In this configuration the 1004* may operate as a general purpose card/paper tape/magnetic tape and/or printer terminal. Data may be transmitted between the 1004 and 1108 using any convenient media. Data to be transmitted is packed by program into blocks, with block and character parity automatically ascribed, and transmission is repeated until the receiving unit (i.e. the 1004 or 1108) indicates that the block has been correctly received. When the 1004 is not being used for data transmission it may function normally as a

* *(1004-Model C)*

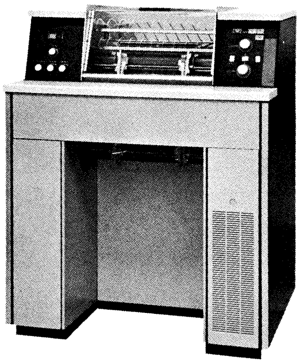
free standing processor, and can be used for local tasks that do not require the attention of the UNIVAC 1108 system.

Character transfer speeds will obviously vary in proportion to the speed of the communication facilities employed. However, since the 1004 is a small computer in its own right, it has the ability to pack and unpack data in any format. A number of standard transmission programs are available for remote 1004 operation, and one of these automatically inserts them upon receipt of compressed data from the 1108. Special control characters are inserted and sensed in each data block to control this compression and expansion. In this manner the transmission rate of the communications facility is utilized in the most efficient way, since only significant data plus a few control characters will be transmitted. Speeds thus cannot be measured in terms of cards per minute or lines per minute, but in the number of significant data characters transmitted in a unit period of time.

The use of remote 1004 Data Processors in a widespread data processing operation makes available to the remote user the full capability of an 1108 system for the relatively short periods that this power is required. Effectively the remote user can demand 1108 facilities at any time – subject to the overall assent of the System Supervisor – send his program and data down the line, request task completion by a certain time and receive the completed output on his own system in his own office or facility. As described under the Executive System, this Demand Processing feature has been specifically built in to the Operating System to accommodate this very type of operation.

The remote user can thus utilize the 1108 power merely for the few large tasks that he may have. For the remainder of the day he can process his relatively minor jobs on his own equipment – the 1004 – under his direct control. These jobs will tend to be those characterized by large volume input/output with little processing – jobs that can be readily handled on small economic equipment locally, and that are not suitable for transmission to a large system at a distant central site.

MULTIPLE LINE PRINTER SUBSYSTEM



The Multiple Line Printer Subsystem provides an economical means (both from the standpoint of cost and the number of input/output channels required), of attaching more than one Line Printer. The subsystem is capable of controlling 1 to 4 Line Printers. It may be attached directly to an input/output channel of an 1108 Processor or Input/Output Controller or indirectly via a Multiple Processor Adapter. There is a 132 character buffer associated with each printer.

The program controls the printing operation by sending function words to the subsystem that specify the operation and number of lines to be spaced. The subsystem responds by generating program interrupts and supplying the status of the subsystem to the Executive System for analysis and action.

The subsystem is so designed that if one printer becomes inoperative, the operation of the others is not affected.

LINE PRINTER

The 1108 Line Printer prints alphanumeric information (63 characters) at 700 lines per minute and a selected character set (including A to Z, 0 to 9 and 5 special characters) at 922 lines per minute.

The Line Printer spaces paper at 20'' per second.

CHARACTERISTICS

PRINTING RATE	700-922 lines/minute
MAXIMUM CHARACTERS/LINE	132
MAXIMUM PRINTED CHARACTERS	63 total (26 alpha, 10 numeric, 27 special characters)
PAPER SPACING AND SPEED	0 to 63 spaces, 20''/sec.
HORIZONTAL SPACING	10 characters/inch
VERTICAL SPACING	6 to 8 lines inch, single spaced manually selected
PAPER STOCK	Any standard sprocket-fed paper 4 to 22 inches wide; up to card stock in weight.

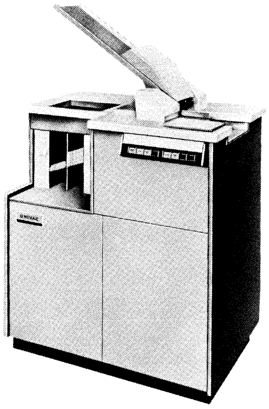
PUNCHED CARD SUBSYSTEM

The Punched Card subsystem comprises a 900 cpm Card Reader, and a 300 cpm Card Punch which are attached to the same Control Unit/ Synchronizer on a single input/output channel of an 1108 Processor or Input/Output Controller.

The Card Reader uses column parallel photodiode sensing, with automatic error checking, error cards being automatically rejected into a separate stacker. A File Feed device is standard. Cards may be read in Fieldata code, in row or column binary.

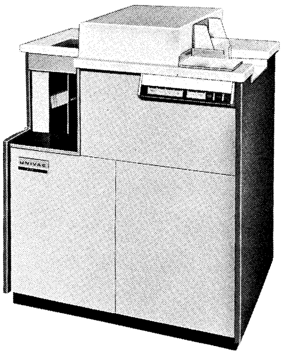
The Card Punch operates on a row-by-row basis and has an automatic check-read station. Incorrectly punched cards are recognized by the Input/Output Handler examining the status word upon the termination of the function, an'd the cards are passed to the error stacker. Under program control additional attempts to punch the data can be made, and if an unacceptable error rate is achieved the job may be suspended for maintenance action. Fieldata, row and column binary punching can be accomplished in any sequence of cards.

Card Reader



CHARACTERISTICS	
CARD READING SPEED	900 cards/minute
INPUT HOPPER CAPACITY	2,500 cards
OUTPUT STACKER CAPACITY	2,500 cards
REJECT STACKER CAPACITY	100 cards
READ MODES	FIELDATA, column binary, row binary
I/O CHANNELS	1 shared with card punch

Card Punch



CHARACTERISTICS	
CARD PUNCHING SPEED	300 cards/minute
INPUT HOPPER CAPACITY	1000 cards
OUTPUT STACKER CAPACITY	2 stackers of 850 cards each
PUNCH MODES	FIELDATA, column binary, row binary
I/O CHANNELS	1 shared with card reader

UNIVAC 1108 COMMUNICATIONS SUBSYSTEM

The UNIVAC 1108 Communications Subsystem enables the UNIVAC 1108 System to receive and transmit data via any common carrier in any of the standard rates of transmission up to 4800 bits per second. It can receive data from or transmit data to low speed, medium speed, or high speed lines in any combination.

The subsystem consists of two principle elements, the UNIVAC Communications Terminal Modules (CTM's), which make direct connection with the communication facilities, and the UNIVAC Communications Terminal Module Controller (CTMC) through which the CTM's deliver data to and receive data from the Central Processor. A Communications Terminal Module Controller may be connected to any general purpose computer channel. If required, a number of CTMC's may be connected through a Scanner Selector to the same general purpose computer channel and is dependent on the number and speed of the communications systems linked to the CTMC's.

Communications Terminal Modules (CTM'S)

There are three basic types of input and output CTM's: low speed (Up to 300 bps*), medium speed (Up to 1600 bps), and high speed (200-4800 bps). Each is easily adjusted to the speed and other characteristics of the type of line with which it is to operate. Each CTM accommodates two full duplex communications lines or two input and two output simplex communications lines. A CTM requires one position of the CTMC.

In addition to the CTM's parallel and Dial Communication Line Terminals (CLT's) may be connected to the CTMC. The parallel CLT receives and transmits in a bit parallel mode rather than bit serial. The Dial-CLT enables the Central Processor to automatically establish communications with remote points via the common carrier's switching network.

Communications Terminal Module Controller (CTMC)

The Communications Terminal Module Controller functions as the link between the processor and the CTM's and CLT's. A CTMC can handle 16 CTM's; this means that a maximum of 32 inputs and 32 outputs can be handled by a single CTMC.

The CTM's may request access to the Central Processor via the Communications Terminal Module Controller in random sequence. The CTMC automatically assigns priority among CTM's requesting access and identifies to the Central Processor the particular CTM granted access. This process is automatic and self-controlled on the UNIVAC 1108 System through Externally Specified Indexing (ESI) on each I/O Channel.

UNIVAC WORD TERMINAL SYNCHRONOUS

The Word Terminal Synchronous (WTS) complements the UNIVAC 1108 Communications Subsystem by enabling the UNIVAC 1108 Central Processor to be used more efficiently for high speed data transmission over a single communications line. Since data characters are transferred to and from Main Storage on a full word basis, six-6 bit characters per word, the I/O transfer time and the size of the area required for buffering are both considerably reduced. However, the most significant advantage of the WTS is that it minimizes the manipulation of the data by the central processor. This is accomplished by having the WTS rather than the central processor add character and message parity to outgoing messages and checking character and message parity on incoming messages. If an error in parity is detected, an external Interrupt is presented to the central processor.

This high speed transmission capability, coupled with the ability to dial connect with any station on the public network complex, will handle many of the data communications problems facing us today. The WTS allows the UNIVAC 1108 to exchange with a remote 1004. This configuration can be used for order entry and inventory control and, more important, for remote processing capabilities.

Operational Modes

There are three basic modes of operation (leased line high speed, Dial network high speed, leased line broadband), which can be used with the WTS.

Bits per second

■ LEASED LINE HIGH SPEED

Leased telephone line operation permits continuous data exchange between two fixed locations at a maximum transfer rate of 300 characters per second.

■ DIAL NETWORK HIGH SPEED

A number of high speed stations can be connected to a WTS, one at a time over dialed (DDD) network circuits. Dial network operation permits data exchange between any two subscriber locations, beginning with a connection procedure (dial) and ending with a disconnect procedure (hang up). Transfers may be initiated by either the WTS or the remote station. If the WTS initiates the transfer, the connection will be made automatically by the central processor through the automatic calling option available on the WTS. The remote station will automatically answer the call and data transfer can commence. If the remote station initiates the call, the WTS will automatically answer the call utilizing the unattended answering option. Data transfers can then take place in the normal manner. All data transfers over dial high speed circuits will be at a maximum transfer rate of 250 characters per second.

■ LEASED LINE BROADBAND

A single broadband station will be connected to a WTS by a leased Telpak A line. Leased Telpak A line operation permits continuous data exchange between two fixed locations at a maximum transfer rate of 5100 characters per second. Data transfer can be initiated by either device providing the remote hardware can handle such an operation. The data format will be such that character parity, start and end of message limits, message parity and character synchronization will be established. The CTS will be prepared to receive data at all times.

These modes of operation reflect the influence of presently available communications tariff offerings. The WTS design and packaging philosophy will allow for future higher speed operation (up to 100,000 bit/sec.) and also for operation with communications hardware using more advanced techniques.

The data sets (201-C, 201-D, 301-B) represent the latest versions of sets which have been on the market for some time. The 201-C succeeds the 201-A, the 201-D succeeds the 201-B, and 301-B succeeds the 301-A. They are basically the same sets (speed, physical makeup, modulation techniques), these changes reflecting minor updating and modifications and, in the case of the 201-C, automatic dialing compatibility.

WTS CHARACTERISTICS

SPEED	2000, 2400, 40,800 bits/sec. capability to 100,000 bps
DATA CODING	6 data bits per character
CONTROL CODING	Control characters (Synch, Start of Message, End of Message, etc.) field selected by plugboard
COMMUNICATIONS FACILITIES	Voice circuit dial or leased at 2000, or 2400 bps respectively. Broadband lease at 40,800 bps or higher
I/O CHANNELS REQUIRED	1 per CTS
I/O TRANSFER MODE	Full word basis (six 6-bit characters per transfer)

COMMUNICATIONS TERMINAL SYNCHRONOUS

The Communications Terminal Synchronous (CTS) is similar to the Word Terminal Synchronous but with less capabilities. Input/Output transfers are on a character at a time basis rather than a full word basis. Character parity is generated and checked by the CTS but not message parity.

The CTS will connect to the same type of communications lines as a WTS.

CHARACTERISTICS	
SPEED	2000, 2400, 40,800 bits/sec. capability to 100,000 bps
DATA CODING	5, 6, 7, or 8 data bits per character
CONTROL CODING	Control characters (Synch, Start of Message, End of Message, etc.) field selected by plugboard
COMMUNICATIONS FACILITIES	Voice circuit dial or leased at 2000, or 2400 bps respectively Broadband lease at 40,800 bps or higher
I/O CHANNELS REQUIRED	1 per CTS
I/O TRANSFER MODE	Character basis (1–5,6,7, or 8 bit character per transfer).

7. PROGRAMMED SYSTEMS SUPPORT

The programmed systems support (software) provided with the UNIVAC 1108 has been designed to meet the total computing requirements of the advanced users of today. The requirements and demands of the large-scale user have grown considerably in the past, and it is in response to these changing requirements that UNIVAC offers with the 1108, a software system designed to meet the present requirements and to allow the change and growth allowed to meet tomorrows challenge. The degree of effective utilization of any computing system is in direct proportion to the scope and versatility of the software. With the 1108, UNIVAC has combined many years of experience in multi-programming, and communication oriented systems to provide a system easy to operate and easy to use, yet a system which ensurés user program integrity in a demand-response environment.

The UNIVAC 1108 provides a full, complete Software System ranging from high level language compilers, to real-time data reduction routines, to concise basic mathematical functions. The major software items are:

- The Executive System
- The ASSEMBLER
- FORTRAN V
- LIFT – FORTRAN II TO FORTRAN V Translator
- COBOL (Extended and Revised)
- ALGOL
- SORT and MERGE
- Application Programs

The entire 1108 software system is constructed around the high speed FH-432 and FH-1782 magnetic drums. By utilizing these high speed drums with their rapid access and high transfer rate, backed up by Mass Storage, instead of relying on magnetic tapes, the 1108 System offers unsurpassed overall system performance.

THE EXECUTIVE SYSTEM

To take maximum advantage of the speed and hardware capabilities of the UNIVAC 1108 System and to make effective use of a given hardware configuration, a comprehensive internal operating environment has been created.

This environment allows for the concurrent operation of many programs; it allows the system to react immediately to the inquiries, requests and demands of many different users at local and remote stations; it allows for the stringent demands of real-time applications; it is able to store, file, retrieve, and protect large blocks of data; and it makes optimum use of all available hardware facilities, while minimizing job turnaround time.

Only through central control of all activities of the UNIVAC 1108 can this environment of the combined hardware and software systems be fully established and maintained to satisfy the requirements of all applications; this responsibility for efficient, flexible, centralized control is borne by the Executive System. The Executive System controls and coordinates the functions of this complex internal environment and, by presenting a relatively simple interface to the programmer allows him to use the system easily, while relieving him of concern for the internal interaction between his program and other co-existent programs.

Modularity

The technical capabilities of the UNIVAC 1108 Executive System span a broad spectrum of data processing activities. Its design is such that no penalties are imposed upon any one of these activities by the support provided for the other activities, and an installation not interested in utilization of the full spectrum may specify capabilities to be eliminated at system generation time.

Batch Processing

Foremost among these capabilities is the support provided for batch processing. Design emphasis has been placed upon the achievement of ease of run preparation and submission, minimization of job turn-around time, and minimization of operator intervention and decision requirements.

Demand Processing (Time-Sharing)

Complementing the batch processing capabilities of the 1108 Executive System are its time-sharing capabilities, the simultaneous accommodation, by the Executive System, of requests and demands from users at numerous remote inquiry terminals, operating in a demand (or conversational) mode. All facilities available to the batch processing user are also available in the demand mode; the primary difference is that the Executive System utilizes its knowledge of the existence of such demand devices to permit the user additional flexibility in the statement of and control of individual Runs. The demand user may communicate directly with the Executive, user program, or he may communicate with a conversational processor.

Real-Time

The 1108 Executive System is also designed to interface with programs which have real-time requirements. The Standard Communication Subsystem, together with efficient scheduling and interrupt processing features of the Executive System, provide an environment satisfactory for any real-time program.

Multiprogramming

Run submission may come from many sources, both remote and central; these various inputs, through the Executive System's use and control of efficient multi-programming techniques, may undergo what is essentially simultaneous input, processing, and output. Thus, in a demanding environment, the full capabilities of the 1108 can be utilized efficiently.

It should be noted that batch, demand, and real-time programs will be processed concurrently by the Executive System, whenever sufficient storage facilities are available; hence, the user of any one mode will experience little variation in his turn-around time, regardless of the proportionate mix with other types of processing.

The Executive System design is also such that extensions may easily be incorporated, to extend its capabilities into the realm of emulation.

Techniques for Utilization of Mass Storage

The UNIVAC 1108 Executive System is designed to provide installations with an effective and efficient utilization of the mass storage devices available with the 1108. The result is an unprecedented ability to relieve operators and programmers of responsibilities in maintaining and physically handling cards, magnetic tapes, etc., thus, eliminating many of the errors which heretofore inherently accompanied the use of large-scale software systems. At the same time, the overall efficiency of operating is considerably improved.

Provisions are made for the maintenance of permanent data files and program files on the mass storage devices, with full facilities for modification and manipulation of these files. Security measures are invoked by the Executive System to insure that files are not subjected to unauthorized use. Provisions are also made within the Executive System for automatic relocation of files of low frequency-usage to magnetic tape, as unused mass storage space approaches exhaustion. When the use of files relocated in such a manner is requested, they are retrieved and restored, under control of the Executive System, with no inconvenience to the user.

The UNIVAC 1108 Executive System is composed of many different routines which perform many different functions. The primary function areas are:

Executive Control Language

The Executive System offers the user a simple means of directing the execution of the individual activities of a run and to relay operational information concerning the run to the Executive. This Executive-User interface is a set of Control Commands, carefully minimized, yet capable of performing all of the desirable or mandatory functions required in a modern Executive System. The Command language is open ended and easily expanded, so that features and functions may be added as the need arises.

The basic format of an Executive Control Statement is quite simple, and is amenable to a large number of input devices. Statements are not restricted to card-image format, and may be of variable lengths. Each statement consists of a heading character (∇), for recognition purposes, followed by a command (which categorizes the statement), followed by a variable number of expressions. The end of a statement is signified by the end of a card, a carriage return, or an equivalent signal, depending on the type of input device.

The Supervisor

The Supervisor is the 1108 Executive System component that controls the sequencing, setup, and execution of all runs entering the 1108. It is designed to control the execution of an unlimited number of programs in a multi-programming environment, while allowing each program to be unaffected by the coexistence of other programs.

The supervisor contains three levels of scheduling, Coarse Scheduling, The Dynamic Allocator, and CPU Dispatching. Runs entering the 1108 are sorted into information files, these files used by the Supervisor for run scheduling and processing. Control statements for each run are retrieved and scanned by the control command interpreter to facilitate the selection of runs for setup by the coarse scheduler. The coarse scheduling of each run is primarily dependent on two factors – the priority of the run, and its facility requirements.

The Dynamic Allocator is responsible for taking runs set up by the coarse scheduler and allotting storage space according to the needs of the individual tasks of a run. Each run may be thought of as being made up of tasks, where a task is defined to be a single operation of a system processor or the execution of a user program. All tasks for a given run will be processed serially; however, tasks of separate runs will be interleaved.

When time-sharing of central storage is appropriate, the Dynamic Allocator is also responsible for initiating "storage swaps" – the writing-out of one program to drum, and replacing it temporarily in central storage with another program. Such action is taken only to provide reasonable response time to remote demand-processing terminals.

The CPU dispatching routine is a third level of scheduling which selects among the various tasks currently occupying central storage whenever it is appropriate to switch the committment of the CPU from one to another. Under normal circumstances, a batch program is allowed to use the CPU until it either becomes interlocked against some event, or until some higher priority program is freed of all of its interlocks.

Time Slicing

In order to accommodate demand mode processing, it is necessary to allow periodic time slices to be assigned to particular routines. This is accomplished by using the timing routine to interlock the currently running program and to examine at specified intervals a separate queue of periodically scheduled routines. Based on the required duty cycle of the demand routines, their priorities, and the priorities of other ready routines, the demand routine is moved to the ready queue with an adjusted priority. In this manner, even low-priority demand routines are given at least occasional use of the CPU.

Storage Compacting

Certain 1108 hardware features make feasible the dynamic relocation of programs residing in central storage – a necessity in order to provide an effective multi-programming environment. At program termination, the storage assigned is returned to the pool of available central storage.

Storage compacting is initiated if, and only if, a requirement exists for contiguous storage, and compacting can meet this requirement. Compacting is never performed unnecessarily, as the storage contents control routine always attempts to fit programs into gaps in the in-use store, if possible.

Facilities Assignment

Available facilities and their disposition are indicated to the system at System Generation time; thereafter, the Executive System assigns these facilities, as needed and as available, to fulfill the facilities requirements of all runs entering the 1108. The Executive System maintains and continually updates inventory tables that reflect what facilities are available for assignment, and which runs are using the currently unavailable facilities.

The File Control System

The 1108 File Control System has been designed to provide a user the highest possible degree of operational flexibility in storing and retrieving data, without requiring concern with the physical characteristics of the recording devices. Thus, most files are made insensitive to input/output media characteristics, as the system adjusts the interface between the file and the device. The system invokes security measures to insure that files are not subject to unauthorized use or destruction. Full facilities are provided for rollback of files from mass storage devices to magnetic tape, as well as the reconstruction of such files on the mass storage devices when they are later referenced by the user; in general, the user need not be aware of the residence of his files.

Comprehensive utility routines are available for the manipulation of files and to provide the user with the current status and structure of his files. Provisions are made for random storage and retrieval-access of data, under the direction of the user. User program files and data files are maintained and processed in the same environment.

Operator Communications

The Executive System has been designed for operation with a minimum of operator intervention. However, it is recognized that some functions frequently in use are beyond the scope of the Executive System, while others demand operator concurrence. In addition, certain information must be presented automatically to the operator, while other information must be available to answer operator requests.

Insofar as operator functions are required for a large variety of activities, the 1108 Executive System apportions these functions into five classes, thus, equally dividing operator duties in a multi-operator installation. These five functional classes are Program Status; Magnetic Tape Activity; Visual Tape Labels; Communications Activity; Card, Printer, and Paper Tape Activity.

The areas above may be associated with as many as five operator consoles or as few as one, depending on the complexity and layout of each installation. Any combination of operator functions may be associated with a physical console operating through the usual console channel or through the Standard Communication Subsystem.

The 1108 contains as standard equipment a Visual Display console to enhance operator-system communications. The advantages of a visual display to the operator are obvious and the possible display functions endless. The Executive System will include a display of information from the ∇ RUN statement, operator requests associated with I/O interventions such as (Printer Interlocks) and manually initiated CLT lines. The operator will have the option to request alternate information, such as backlog status. In the event that the available display area becomes filled up, the Executive will defer lower priority displays and compact or summarize displays.

Diagnostic System

A comprehensive diagnostic system is available within the 1108 Executive System to aid the checkout of user programs. Both allocation time and compilation or assembly time commands are available to trigger snapshot dumps. Post-mortem dumps are also available through an Executive Control Statement.

Input/Output Device Handlers

The Input/Output device handlers are responsible for controlling the activities of all input/output channels and peripheral equipment attached to the 1108. These device handlers provide the user with a full complement of device operations for magnetic tape, FH-432 and FH-1782 drums, FASTRAND, card readers, printers, communication line terminals, and others.

Auxiliary Processors

Contained within the Executive System is a set of auxiliary processors for performing functions complementing those of the source language processors such as FORTRAN. This set of processors includes the Collector for linking relocatable sub-programs, the Procedure Definition Processor for inserting and modifying Procedure Definitions in a library, the COBOL Library Processor for manipulation of COBOL Library Elements, and the DATA Definition Processor for introducing data descriptions.

Utilities

Included within the Utilities Section of the Executive System are diagnostic routines, program file manipulation routines, file utility routines, and cooperative routines for aiding the user in performing such functions as reading cards, printing line images on a printer, transferring files from device to device, and carrying out housekeeping functions required for file-residence on mass storage devices.

Processor Interface Routines

The Processor Interface Routines, provide a simple, standard interface for all processors within the system. Complete facilities are provided for the input of source-language statements and the output of the resulting relocatable binary code.

System Setup

The System Setup section of the Executive System provides an installation with a means of generating a system tailored to its particular needs, and for subsequently maintaining this system with a minimum of effort.

THE ASSEMBLER

The Assembler is an assembly program for a symbolic coding language composed of simple, brief expressions providing rapid translation from symbolic to machine language relocatable object coding for the UNIVAC 1108.

The Assembly language includes a wide and sophisticated variety of operators which allow the fabrication of desired fields based on information generated at assembly time. The instruction operation codes are assigned mnemonics which describe the hardware function of each instruction. Assembler directive commands provide the programmer with the ability to generate data words and values based on specific conditions at assembly time. Multiple location counters provide a means of preparing for program segmentation and controlling address generation during assembly of a source code program. The Assembler produces a relocatable binary output in a form suitable for processing by the loading mechanism of the system. It supplies a listing of the original symbolic coding and an edited octal representation of each word generated. Flags indicate errors in the symbolic coding detected by the assembler.

Symbolic Coding Format

In writing instructions using the Assembly language, the programmer is primarily concerned with three fields: a label field, an operation field, and an operand field. It is possible to relate the symbolic coding to its associated flowchart, if desired, by appending comments to each instruction line or program segment.

All of the fields are in free form, providing the greatest convenience possible for the programmer. Consequently, the programmer is not hampered by the necessity to consider fixed form boundaries in the design of his symbolic coding.

Mnemonic Instructions

The assembly program recognizes a set of mnemonic instructions representing the machine code instructions listed in Appendix B.

Assembler Directives

The eleven symbolic assembler directives control or direct the assembly processor just as operation codes control or direct the central computer. These directives are represented by mnemonics which are written in the operation field of a symbolic line of code and the flexibility of these directives is the key to power of the assembler. The directives are used to equate expressions, to adjust the location counter value, and to afford the programmer special control over the generation of object coding.

Additional Features

Facilities for interprogram communication permit separately assembled programs (or subprograms) to be linked together at load time. A label followed by an asterisk is defined as an external label which can be referenced by other programs, as well as by the program in which it is defined. A job to be executed may be composed of many subprograms (or elements). The recompilation of any element does not necessitate the recompilation of the remaining elements which compose the job. The program is actually constructed, using the technique above, at the time of execution, or may be constructed prior to this time.

UNIVAC 1108 FORTRAN V

FORTRAN V is an algebraic language designed primarily for scientific and engineering computations and closely resembles the language of mathematics. As a language, it is the logical outgrowth of the earlier FORTRAN languages and is generally compatible with them (although the earlier languages are not a proper subset of FORTRAN V). The UNIVAC 1108 FORTRAN V language has been extended to allow for more flexible data handling and to minimize user programming efforts, yet it is an upward compatible language from 1107 FORTRAN IV and IBM FORTRAN IV (as announced in IBM form C-28-6274-1 in 1963). UNIVAC 1108 FORTRAN has all of the features of the proposed ASA FORTRAN IV language plus many valuable extensions which, as will be shown, significantly increase the power and flexibility of the language, particularly in the areas of data handling. For further information, consult the UNIVAC 1108 FORTRAN Reference Manual.

Language Extensions

The following extensions are currently available in UNIVAC 1107 FORTRAN IV and are included in UNIVAC 1108 FORTRAN V.

1. PARAMETER Statement assigns specified integer values to specified variables at compile time, e.g. PARAMETER I=2 causes the integer 2 to replace I whenever it occurs in the source program. This facilitates the assignment of different values to frequently referenced parameters in different compilations of the same program.
2. ABNORMAL Statement
   The optional ABNORMAL statement permits increased optimization of object programs. Where common sub-expressions occur within a statement, it is obviously desirable to evaluate each sub-expression only once. Where the common sub-expressions contain function references, however, there is a possibility that the function will produce a different result upon successive references with the same arguments. Because of this possibility, most FORTRAN systems are forced to re-evaluate sub-expressions containing function references at each occurrence. UNIVAC FORTRAN V permits all functions that can produce different results from identical sets of arguments to be designated ABNORMAL. All common expressions except those that reference ABNORMAL functions are evaluated only once. When the ABNORMAL statement does not appear at all in a program, *all* function references are considered ABNORMAL and are re-evaluated at each occurrence, as in most other FORTRAN Systems.
3. Non-Standard subroutine return (of the form RETURN k) where k specifies the subroutine argument to which a return is made.
4. Statement Labels as CALL arguments.
5. A variable may have up to seven subscripts.
6. Internal subprograms are permitted.
7. Mixed Arithmetic.
   Variables of different types may occur in the same expression with two exceptions:
   - Logical Variables may not be mixed with other types.
   - Double Precision and Complex can not be mixed.
8. Extended Subscript Expressions
   Subscripts may have the form $\pm M_1 \pm M_2 \ldots \pm M_j$ where $N*K_1*K_2 \ldots K_M$

9. Forward and Backward DO Loops

10. Generalized Assigned GO TO
    The assigned GO TO need not have a list of possible assignments.

The following language enhancements are available with 1108 FORTRAN V:

1. Bit and Character Manipulation Capabilities

   A string of consecutive bits, called a field, may be defined and operated on by making use of the FLD (e_1, e_2, e_3) intrinsic function, where e_1 and e_2 determine a field of expression e_3. e_1 and e_2 are integer expressions which give the starting position (e_1) and the length (e_2) of the field being defined.

2. The NAMELIST Statement

   The non-executable NAMELIST Statement provides data-characteristic information at object time, and may be used instead of a LIST on an INPUT/OUTPUT Statement and the associated FORMAT Statement. A NAMELIST name (1-6 alphanumeric characters) is defined by its appearance in a NAMELIST statement, and thereafter may appear only in an input or output statement that requires a list.

3. The DEFINE Statement (of the form DEFINE R(a_1, ... , a_n) = e or DEFINE R=e) where R and a_i are variable names and e is any expression not involving R's which have not been defined. The DEFINE Statement permits mathematical equivalence between arrays and variables, permits subscripting of subscripts, and allows dynamic storage allocation for arrays.

4. The INCLUDE Statement (of the form INCLUDE n, LIST) where n is the name assigned to a set of FORTRAN Statements, previously filed, which are to be included in the program at this point. The word LIST is optional and if entered the "Included" Statements will be listed whenever the source program is listed.

5. The IMPLICIT Type statement of the form IMPLICIT *type* (a_1, a_2. . .), type (a_1, a_2, . . .) where type is INTEGER, REAL, LOGICAL, DOUBLE PRECISION or COMPLEX and the a_i represents alphabetic characters or a range of alphabetic characters. The IMPLICIT type statement allows the user to declare the type of variables by specifying that variables beginning with certain designated letters (the a_i) are of a certain type.

6. The ENTRY statement of the form ENTRY *name* (a_1, a_2, . . . , a_n) where *name* is the name of an entry point and where the a_i are dummy arguments. The entry statement permits an entry to a subroutine or function by a CALL Statement or a function reference that references an ENTRY statement. Entry is made at the first executable statement following the ENTRY Statement.

7. The DELETE Statement

   The compile time-interpretive DELETE statement provides the programmer with a simple facility to prevent the generation of object code. It is of the form DELETE n or DELETE n, V where n is a statement label and V is the integer 0 or 1 (or is a name assigned the value 0 or 1 via the PARAMETER statement). V=0 implies that the DELETE statement is not effective while V=1 implies that DELETE is effective.

8. Hardware Double Precision

   1108 FORTRAN V processes double precision quantities in 1108 double precision format, within:

   a. The 1108 FORTRAN V Compiler

   b. All mathematical function routines (where appropriate)

   c. The I/O Conversion programs

9. Automatic Type Assignment

   The arithmetic type of the argument is used by the compiler to determine the correct function routine (i.e. SQRT, DSQRT, CSQRT) to be called.

10. Extended FORMAT Specifications

    The 1108 FORMAT Control has been augmented by the addition of new FORMAT control specifications of the form:

    aGw.s Generalized format code that may be used to determine the desired form of data whether it is integer, real, complex or logical.

Gw.d Outputs fixed, floating, or integer representation depending on the type and magnitude of the data value.

Tw Causes the pointer in an input or output record to point to the wth character in the record.

Lw Logical field specification

11. Imbedded Comments

    A "Master Space" character will cause the compiler to ignore all subsequent information on that line.

12. Hollerith Strings may have the form 'c_1 c_2 . . . c_i' where c_i is any Hollerith character, including blank.

13. Additional Intrinsic Functions

    a. To aid in code optimization for converted FORTRAN II programs four additional intrinsic functions have been added: AND (e_1, e_2); OR (e_1, e_2); COMPL (e); BOOL (e)

14. Boolean Operations

    The Boolean functions OR, AND, XOR, and NOT have been added to operate on logical expressions. These operators can also be supplied to Boolean, integer, or real expressions to produce a Boolean result.

Compiler Organization

The UNIVAC 1108 FORTRAN V source language processor accepts FORTRAN statements and produces a highly efficient relocatable object code element, acceptable to the Executive System for execution, the Filing System for cataloging and filing, or both. FORTRAN, and all other 1108 processors, generates its own code and does not require an assembler pass.

The FORTRAN V compiler is modular in design and consists of six phases. Although the phases have been separated as such on the basis of general operations performed on the source program, not every phase processes the entire program. The basic design objective was to create a compiler which, while quite rapid as a processor, will produce an object program optimized with respect to both storage requirements and execution time.

The compilation process involves the successive execution of six phases, a summary of which is presented in the following paragraphs:

- Phase 1 of the compiler is concerned with transformation of the UNIVAC FORTRAN V program source code into an internal format. Files and tables of relational information, implicit in the source program but not easily accessed will be constructed.
- Phase 2 deals with storage assignments for variables and performs an analysis in and out of loops.
- Phase 3 and
- Phase 4 deals with arithmetic optimization and index register optimization.
- Phase 5 deals with code generation and storage assignment for those quantities not assigned storage by Phase 2.
- Phase 6 The final phase, completes the generated instruction in a relocatable binary format and optionally edits all output, including error messages.

The Compiler performs several types of optimization on a source program:

1. Local Optimization

   This involves the reduction of expressions involving constants to a single constant.

2. Inter-statement Arithmetic Optimization

   This optimization has three forms, a) the elimination of redundancies in loading of index and arithmetic registers, b) the recognition of "Common Sub-Expressions" from previous statements and c) the removal from a loop of these computations within a loop structure which are constant relative to the loop.

3. Inter-statement Indexing Optimization

   This involves a study of the DO-loop structure, entries and exits from loops, the form of subscripts and the loop parameters.

As a result of this optimization, and other features of the system, the object code produced by the Compiler is more efficient that the average programmer can write in assembly language. This powerful FORTRAN processor averages 5000 statements per minute, yet is the most effective code generating compiler in the industry. This compiler is drum oriented and achieves this performance without the use of magnetic tape.

LIFT FORTRAN II TO FORTRAN V TRANSLATOR

LIFT is a source language translator which accepts a FORTRAN II Source Language Program as input, performs a translation, and outputs a Source Language Program acceptable to a FORTRAN Compiler (not necessarily the UNIVAC FORTRAN Compiler). There is a need for translation since FORTRAN II is not a proper subset of FORTRAN V; that is, there are statement types in FORTRAN II that are not acceptable to FORTRAN V. LIFT, itself written in FORTRAN IV, is a Program File element and fully integrated with the Executive System.

Features

There are nine areas of incompatibility between FORTRAN II and FORTRAN V, and the basic purpose of LIFT is tq generate FORTRAN V Source Statements which replace the unacceptable FORTRAN II Statements.

1. The "F" Card
2. Functions
3. Boolean Statements
4. Double-Precision and Complex Statements
5. COMMON Statements
6. Arithmetic Statement Functions
7. Dimension Statements
8. Hollerith Literals
9. Implicit Multiplication.

There are also five types of FORTRAN II statements, that, although acceptable to the UNIVAC FORTRAN V processor, are converted to their FORTRAN V equivalents. LIFT offers two features that ease the transfer between computers: the ASSIGN and REPLACE card Options. The ASSIGN Card allows for a temporary change to the I/O Assignment Table, and the REPLACE Card allows the user to have every occurrence of a variable name replaced with another variable. The standard output produced by LIFT consists of a listing of the FORTRAN II program, an annotated list of the translated program, and a symbolic program element suitable for use as input to any FORTRAN V Compiler.

UNIVAC 1108 COBOL

The UNIVAC 1108 COBOL-64 is a source language processor, operating with the 1108 Executive System, which accepts statements written in the COBOL-64 language (as adapted for the 1108) and generates a complete program ready for execution. The acceptable language to the 1108 COBOL processor includes all of "required" COBOL-64 and the meaningful electives. For further information, consult the UNIVAC 1108 COBOL Programmer's Guide.

The major electives implemented on the 1108 are:

1. CORRESPONDING option of the MOVE, ADD, and SUBTRACT Verbs
2. COMPUTE Verb and formulae
3. SORT (DOD Style)
4. Segmentation
5. Mass Storage
6. INCLUDE
7. Characters used in formulas (+, –, *, /, **, =), in relation (+, >, <)
8. Literals up to 132 characters
9. The ENTER Verb
10. The LOCK option on the CLOSE Verb
11. The ADVANCING option on the WRITE Verb
12. The REVERSED option of the OPEN Verb
13. Operands used in Arithmetic can be up to 18 digits in SIZE
14. AND and OR connectors in Compound Conditions
15. Parentheses in Compound Conditions

16. All abbreviations of Conditional Statements
17. The OBJECT-COMPUTER paragraph
18. The APPLY clause
19. RERUN
20. The DATA-COMPILED clause
21. Library provisions
22. Multiple results from Arithmetic Verbs

The following features, implemented in UNIVAC 1108 COBOL, are special UNIVAC extensions to the COBOL language:

- MONITOR
- Common Storage

A Report Generator, not part of the COBOL processor, is available for COBOL programs.

The Processor

The COBOL processor is a 6 phase compiler operating within the UNIVAC 1108 minimum configuration. The compiler is completely modularized into relocatable elements and is handled as any program in the system, thus providing for easy expandability and maintenance. Likewise, the COBOL processor produces as its output relocatable binary elements stored on the drum or mass storage, which are indistinguishable from other elements in the system. Other outputs from the compiler include extensive diagnostic messages, source language listing, machine language listings, and special cross reference listings of name definitions and their references. The machine language listing consists of side by side Procedure Division statements and the corresponding generated symbolic machine code. The compiler diagnostics are of four categories:

Precautionary – print warning message and continue compilation.

Correctable – try to correct the error, print explanatory message and continue compilation.

Uncorrectable – if intent of programmer cannot be determined, print a diagnostic message, reject the clause or statement and continue compilation.

Catastrophic – when so many errors have occurred that no more useful diagnostic information can be produced, terminate the compilation.

SPECIAL FEATURES

Segmentation

COBOL programs can be segmented by use of priority numbers on procedural sections.

MONITOR

Provides dynamic program checkout facilities.

Library

A COBOL library processor is available to store and retrieve Data and Procedure Division Descriptions.

Rerun

The programmer can specify rerun after any number of records have been processed. Dumps are written on a specified output file with bypass sentinels for later restart of the program.

Common Storage

Since COBOL programs can be chained (an Executive function), intermediate data results can be maintained between programs using the Common Storage provision of UNIVAC COBOL. The elements sharing Common Storage may be from another 1108 processor (e.g. FORTRAN).

Overpunched Sign Convention

The overpunched sign convention will be implemented on the 1108.

UNIVAC 1108 ALGOL

The ALGOL language allows the mathematician or engineer to prepare programs for the UNIVAC 1108 without the necessity of becoming familiar with the details of the internal machine operation. The ALGOL Compiler then generates, from this pseudo-mathematical source language, efficient coding in a relocatable binary format acceptable to the Executive for execution, the filing system for cataloging and filing, or both.

The UNIVAC 1108 ALGOL compiler conforms to the specifications arrived at jointly by the ACM Committee on Programming Languages and the GAMM Committee on Programming, and as published in the Communications of the ACM, May and July, 1960.

UNIVAC 1108 ALGOL is an "extended hardware" representation of ALGOL '60 to employ efficiently the UNIVAC 1108 and associated peripheral equipment. Certain extensions to basic ALGOL have been implemented, namely, the ability to handle the powerful input/output logic and the ability to name strings.

UNIVAC 1108 SORT AND MERGE

The UNIVAC 1108 Sort and Merge Package has been designed to be fully modular, with every functional unit completely self-contained. This permits the various units to be optimized for their own particular tasks, enabling them to be associated in the most effective form, and allowing for each of updating and augmentation.

The Package is not a generator of specialized Sort/Merge routines; rather the user calls and adapts the independent modules for all his specific sorting needs by presenting his parameter values on control cards at load time.

In the internal sort the replacement selection method, which takes advantage of any inherent sequence in the original date, is used and strings may be written upon magnetic tape or drum. The FH-432 and FH-1782 Drums, because of their high transfer rates and rapid access, minimize processor waiting time and thus greatly enhance efficient sort operations. Any random access unit areas may be defined by the Supervisor and these will automatically be used by the subsystem if advantage can be gained thereby.

The input data to be sorted may be held on magnetic tape, punched cards or magnetic drum.˙ User own coding may be inserted on the first and final passes of the Sort and Merge operation and may also replace the standard comparison routines. Sorting generally requires the use of two magnetic tape units, although additional units can be employed to give faster times.

Keys may be in multiple form and can be recorded, modified, and packed. Standard collating sequences are intrinsically provided for, but the user may define any collating sequences he requires, up to a maximum of seven, and any combination of these may be utilized in the same run. Fixed or variable length items can be handled.

The Sort and Merge Package normally uses 20,000 words of central storage, 262,000 words of of magnetic drum storage, and Magnetic Tape Units of any kind as required, but the user may specify more central and drum storage, and additional Magnetic Tape Units to increase efficiency and speed.

MATHEMATICAL FUNCTION PROGRAMS

The UNIVAC Software System includes an extensive collection of basic mathematical subroutines and functions. This collection includes all of the standard FORTRAN functions and has been expanded by over 50% to give the programmer a more complete coverage of the "often used" mathematical routines. Each of these mathematical routines has been carefully developed to offer the programmer maximum accuracy and range with a minimum routine size and executive time. These routines are available to each of the program languages, FORTRAN V, Assembler, COBOL (through the use of the ENTER verb option), and ALGOL. One group of routines, the series of exponentiation routines (NEXPi), are automatically referenced when the FORTRAN V Source Program indicates exponentiation with the operator**, and inline exponentiation is not feasible.

Features

The various mathematical function programs are:

1. Library Functions
2. The FORTRAN Built-in Function
3. The Exponentiation Functions

These routines are available to the processors in different manners. For example, the routine SQRT provides a single precision square root of the argument (x). To utilize this routine the processors employ the following calling sequences:

- FORTRAN V Calling Sequence:
  ROOT = SQRT (x), when x is a FORTRAN V real variable.
- ASSEMBLY Calling Sequence:

```
  x       LMJ    B11, SQRT
x + 1      +     Address of x
x + 2            normal return
   The result is left in A0.
```

- COBOL Calling Sequence:
  ENTER SQRT REFERENCING X

APPLICATION PROGRAMS

The UNIVAC 1108 has an extensive library of application programs and subroutines. The major application programs, such as Linear Programming (LP), Automatic Program Tooling (APT III), and PERT TIME/COST will be briefly described, while others shall simply be mentioned by titles. This is by no means an exhaustive list of applications or subroutines, for it is simply meant to demonstrate the many, diverse, types of applications which are available with the UNIVAC 1108.

Linear Programming System

Linear programming has become one of the most useful and frequently used operations research techniques in manufacturing and transportation industries. In the production and distribution of products, LP provides a solution to minimize costs or maximize profits. The LP System developed for the UNIVAC 1108 embodies the latest advances in computer technology with the most powerful algorithm to date. The algorithm essentially employs the "product form of the inverse" method and is improved with an advanced path selection technique. The package is coded in FORTRAN V and Assembly Language.

The exceptional features of the 1108 LP System are as follows:

1. *The System can accommodate 2047 rows.*
2. *The speed and random access properties of Magnetic Drums place the System at a distinct advantage over tape and disc handling procedures.*
3. *Both single precision and double precision computations are available. The selection may be manual or automatic.*
4. *The control language of this System is far superior to any other existing LP System. It is an "interpretive control language". The sophisticated user may use macros in constructing his command string to implement the System. On the other hand, the average user may still execute his basic LP problem without detail knowledge of the control language.*
5. *The LP System is imbedded as part of the Executive System. This enhances the System as a powerful model builder. Matrix builders and output analyzers may be attached to the LP System to form corporate models. The entire model may be optimized and re-optimized in cycles in one computer run. Non-linear programming may be accomplished by solving approximated linear functions in each cycle.*

In addition to the above features, the 1108 LP System contains the flexibilities existing in other LP Systems. Vector levels can be specified and coefficients may be modified. Long LP runs can be split with restart procedures. Post-optimal parametric programming or a complete tableau can be obtained. The final output includes the objective function value, optimal basis, vector levels, and reduced costs (d_j).

APT III

APT (Automatically Programmed Tools) is a system for the computer assisted programming of numerically controlled machine tools, flame cutters, drafting machines, and similar equipment. It is production-oriented, written to simplify the effort, time, and money needed to take full advantage of numerically controlled techniques in engineering and manufacturing.

In addition to providing machine tool programming capabilities virtually impossible by manual methods, APT enhances most of the usual advantages found in numerical control: reduced lead time, greater design freedom and flexibility, lower direct costs, greater accuracy, improved production forecasting, lower tooling costs, better engineering control of the manufacturing process, and simplified introduction of changes.

The APT III program represents over one hundred man years of development and testing. After extensive experience with our earlier program, APT II, the Aerospace Industries Association made a new start and wrote APT III from the beginning, during the calendar year 1961. At the completion of this package, APT III was turned over to the Illinois Institute of Technology Research Institute for further development, under the APT long-range program. The use of certain parts of APT requires membership in this long range program.

UNIVAC participated in the original writing of APT III and has been a member of the APT long-range program from the beginning. Numerical control specialists are continually working to keep the UNIVAC 1108 APT program in the forefront of the art. As implemented on the UNIVAC 1108 System APT III will continue to conform to the latest APT long range program specifications.

PERT – A Major Applications Program

The UNIVAC 1108 PERT/COST System is a generalized applications program based upon the framework provided by the "DOD/NASA Guide to PERT/COST System Design". The DOD/NASA design is based upon the concept of costing *work packages* rather than individual network activities. A work package is a discrete unit of work required to complete a specific job or process. The work packages of a research and development project are directly related to activities or groups of activities on the project network. The work package is the basic unit of the PERT/COST System for which actual project costs are collected and compared with estimates for purposes of cost control.

The UNIVAC 1108 PERT System adheres rigorously to the DOD/NASA design and has developed a system which satisfies the conceptual aims of PERT/COST.

Many government agencies and contractors are currently processing PERT/TIME data on existing systems. PERT/COST is relatively new and will be subject to the inevitable modifications that will result from its initial pilot tests by DOD. In order to lessen the impact of these changes upon existing PERT/TIME programs and to provide for efficient integration of time and cost data, a modular design was adapted.

The modular structure of the program permits separate processing of the time networks and of the work package costing structure while simultaneously providing for integrated time and cost reporting. The PERT/TIME Module of the UNIVAC 1108 PERT/COST System accepts as input a deck of cards describing the PERT network. The cards are processed and used to update the PERT/TIME Master File. Network computations are performed and the time reports are generated. The PERT/COST Module accepts the Cost Breakdown Structure, actual and estimated costs for the project work packages, and a table of labor rates and applicable overhead percentages. Cost data is accumulated up through the Cost Breakdown "tree," integrated with the time information, and the required cost reports are generated.

BEEF* (Business and Engineering Enriched Fortran)

BEEF enriches FORTRAN to overcome FORTRAN's limitations as a data processing language and to enhance its abilities as a scientific processor. The scientific enhancement is in the form of a set of mathmetical subroutines, referred to as MATH-PACK. The list of MATH-PACK routines is contained in Appendix D and includes routines for mathematical functions, matrix arithmetic and other standard engineering requirements. To enrich FORTRAN as a data processing language, FORTRAN is augmented with a package of callable subroutines suitable for management information systems and business data processing. These routines are listed in Appendix E.

Thus, enriched FORTRAN now serves a dual purpose, suitable for a large spectrum of applications in either science or business. The advantages to the user in being able to standardize on one compiler language are obvious.

**Developed by Westinghouse Electric Corp., Baltimore Defense and Space Center.*

Statistical Routines (STAT PACK)

In addition to the mathematical routines available via MATH PACK, UNIVAC has implemented a complete set of statistical routines (called STAT PACK) for the 1108 users. STAT PACK includes routines for descriptive statistics, tests on statistical parameters (such as chi-squared tests), analysis of variance, regression and correlation analysis, factor analysis, time series analysis, multi-varient analysis and distribution functions. A list of the STAT PACK routines is contained in Appendix F.

APPENDIX A. INSTRUCTION REPERTOIRE (By Operation Type)

GLOSSARY AND CONVENTIONS

Abbreviations and symbols frequently used in the description of the instruction repertoire are given below:

( )	Contents of register or address within parentheses.
$(\ )'$	Complement of contents of register or address.
$\lvert(\ )\rvert$	Absolute value or magnitude.
$(\ )_{17\text{-}00}$	Subscripts indicate the bit positions involved. A full word is normally not subscripted. Subscripts are also used to designate octal or decimal notation.
( )c	Floating point biased exponent.
( )f	Final contents.
( )i	Initial contents.
( )m	Floating point fixed point part.
( )j	j-designated portion.
f	Function code.
j	Partial word designator or function code extension.
a	Arithmetic register designator. In input/output instructions, "a" designates an I/O channel.
A	Arithmetic Register.
x	Index register designator.
x_a	Index register designator in a-field.
X	Index Register.
X_a	Index Register specified by coding x_a.
X_Q	Modifier portion of an index register.
X_Δ	Increment portion of an index register.
r	Same as r_a .
r_a	Designator specifying an R Register. It is coded in the a-designator position of an instruction word.
R	R Register.

R_a R Register specified by coding r_a

u The base address of the operand (or the actual operand) as coded in u-field of an instruction.

U The effective address or value of the operand after application of indexing and indirect addressing.

U_d Destination address.

U_s Source address.

h h-designator of the instruction word. A value of "l" specifies incrementation of an index register.

i i-designator of the instruction word. A value of "l" specifies indirect addressing.

PSR Processor State Register

$\wedge$ Symbol denoting logical product, or logical AND.

$\vee$ Symbol denoting logical sum, or inclusive OR.

$\not\equiv$ Symbol denoting logical difference, or exclusive OR.

→ Direction of data flow.

UNIVAC 1108 WORD FORMATS

INSTRUCTION WORD

f	j	a	x	h	i	u
35 30	29 26	25 22	21 18	17	16	15 0

INDEX-REGISTER WORD

Δ	Q
35 18	17 0

NON-ESI ACCESS-CONTROL WORD

G	W	V
35 34	33 18	17 0

ESI ACCESS-CONTROL WORD

G	H	W	V
35 34	33	32 18	17 0

SINGLE-PRECISION FIXED-POINT WORD

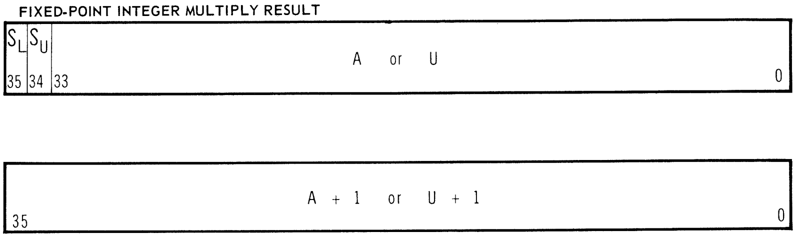
S	
35	34 0

DOUBLE PRECISION FIXED-POINT WORD

S	A or U
35	34 0

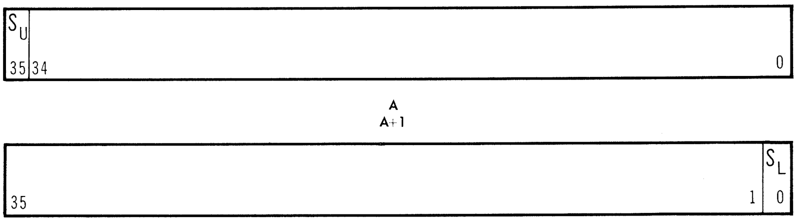
A + 1 or U + 1
35 0

FIXED-POINT INTEGER MULTIPLY RESULT

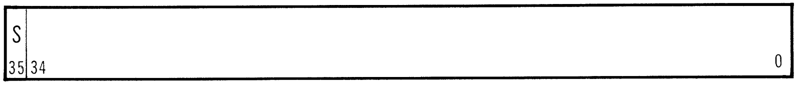


(Left-circular shift A, A+1 by one to align least-significant operand)

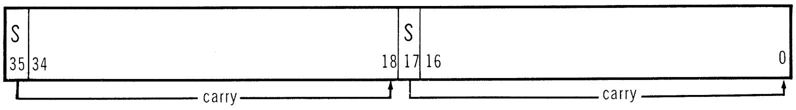
FIXED-POINT FRACTIONAL MULTIPLY RESULT



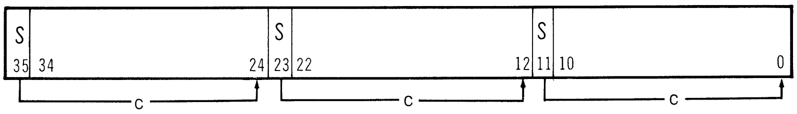
FIXED-POINT MULTIPLY SINGLE INTEGER RESULT



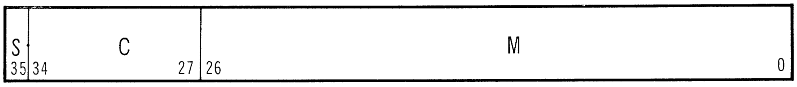
ADD-HALVES WORD FORMAT



ADD-THIRDS WORD FORMAT

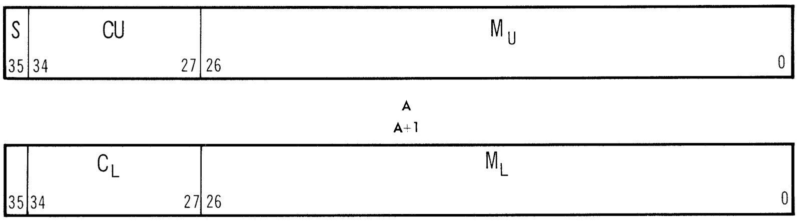


SINGLE PRECISION FLOATING POINT OPERAND

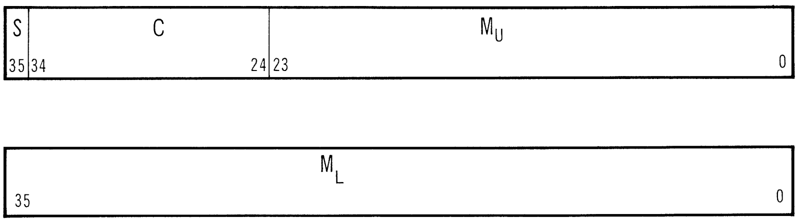


$C_L = C_U - 27$, Word 2 contains unnormalized least significant result.

SINGLE PRECISION FLOATING POINT RESULT



DOUBLE PRECISION FLOATING POINT OPERAND OR RESULT



THE INSTRUCTION FAMILIES OF THE UNIVAC 1108 REPERTOIRE

The commands of the UNIVAC 1108 are grouped according to their function, rather than by a numeric listing of operation codes. Some instructions belonging to more than one category, may appear more than once.

1. Load Instructions

Instruction Code		Mnemonic	Instruction	Execution Times + in usec.
f	j			
10	0-17	LA	Load A	.75
11	0-17	LN,LNA	Load Negative A	.75
12	0-17	LM,LMA	Load Magnitude A	.75
13	0-17	LNMA	Load Negative Magnitude A	.75
23	0-17	LR	Load R	.75
26	0-17	LXM	Load X Modifier	.875
27	0-17	LX	Load X	.75
46	0-17	LXI	Load X Increment	1.00
71	13	DL	Double Load A	1.50
71	14	DLN	Double Load Negative A	1.50
71	15	DLM	Double Load Magnitude A	1.50

2. Store Instructions

Instruction Code		Mnemonic	Instruction	Execution Times + in usec.
f	j			
01	0-15	SA	Store A	.75
02	0-15	SN,SNA	Store Negative A	.75
03	0-15	SM,SMA	Store Magnitude	.75
04	0-15	SR	Store R	.75
05	0-15	SZ	Store Zero	.75
06	0-15	SX	Store X	.75
22	0-17	BT	Block Transfer, repeat	1.50+1.5K
71	12	DS	Double Store A	1.50

+ *Times are given for alternate bank case only. In cases where instructions and operand are procured from the same bank, add .75 usec. to execution time. For function codes 01-67, add .375 microseconds to execution times for sixth or third word writes.*

* *For all comparison instructions, the first number represents the skip or jump condition, the second number is for no skip or no jump condition.*

3. Fixed Point Arithmetic Instructions

Instruction Code		Mnemonic	Instruction	Execution Times + in usec.
f	j			
14	0-17	AA	Add to A	.75
15	0-17	ANA	Add Negative A	.75
16	0-17	AM,AMA	Add Magnitude to A	.75
17	0-17	ANM,ANMA	Add Negative Magnitude to A	.75
20	0-17	AU	Add Upper	.75
21	0-17	ANU	Add Negative Upper	.75
24	0-17	AX	Add to X	.75
25	0-17	ANX	Add Negative to X	.75
30	0-17	MI	Multiply Integer	2.375
31	0-17	MSI	Multiply Single Integer	2.375
32	0-17	MF	Multiply Fractional	2.375
34	0-17	DI	Divide Integer	10.125
35	0-17	DSF	Divide Single Fractional	10.125
36	0-17	DF	Divide Fractional	10.125
71	10	DA	Double Precision Fixed Point Add	1.625
71	11	DAN	Double Precision Fixed Point'Add Negative	1.625
72	04	AH	Add Halves	.75
72	05	ANH	Add Negative Halves	.75
72	06	AT	Add Thirds	.75
72	07	ANT	Add Negative Thirds	.75

4. Floating Point Instructions

Instruction Code		Mnemonic	Instruction	Execution Times + in usec.
f	j			
76	00	FA	Floating Add	1.75
76	01	FAN	Floating Add Negative	1.75
76	02	FM	Floating Multiply	2.625
76	03	FD	Floating Divide	8.25
76	04	LUF	Load and Unpack Floating	.75
76	05	LCF	Load and Convert to Floating	1.125
76	06	MCDU	Magnitude of Characteristic Difference to upper	.75
76	07	CDU	Characteristic Difference to upper	.75
76	10	DFA	Double Precision Floating Add	2.625
76	11	DFAN	Double Precision Floating Add Negative	2.625
76	12	DFM	Double Precision Floating Multiply	4.250
76	13	DFD	Double Precision Floating Divide	17.25
76	14	DLUF	Double Load and Unpack Floating	1.50
76	15	DLCF	Double Load and Convert to Floating	2.125
76	16	FEL	Floating Expand and Load	1.00
76	17	FCS	Floating Compress and Store	1.625

5. Repeated Search Instructions

Instruction Code		Mnemonic	Instruction	Execution Times + in usec.
f	j			
62	0-17	SE	Search for Equal, repeat	2.25 + .75K
63	0-17	SNE	Search for Not Equal, repeat	2.25 + .75K
64	0-17	SLE,SNG	Search for Less or Equal, repeat	2.25 + .75K
65	0-17	SG	Search for Greater, repeat	2.25 + .75K
66	0-17	SW	Search for Within Range, repeat	2.25 + .75K
67	0-17	SNW	Search for Not Within Range, repeat	2.25 + .75K
71	00	MSE	Masked Search for Equal, repeat	2.25 + .75K
71	01	MSNE	Masked Search for Not Equal, repeat	2.25 + .75K
71	02	MSLE,MSNG	Masked Search for Less or Equal, repeat	2.25 + .75K
71	03	MSG	Masked Search for Greater, repeat	2.25 + .75K
71	04	MSW	Masked Search for Within Range, repeat	2.25 + .75K
71	05	MSNW	Masked Search for Not Within Range, repeat	2.25 + .75K
71	06	MASL	Masked Alphanumeric Search for Less or Equal, repeat	2.25 + .75K
71	07	MASG	Masked Alphanumeric Search for Greater, repeat	2.25 + .75K

6. Shift Instructions

Instruction Code		Mnemonic	Instruction	Execution Times + in usec.
f	j			
73	00	SSC	Single Shift Circular	.75
73	01	DSC	Double Shift Circular	.75
73	02	SSL	Single Shift Logical	.75
73	03	DSL	Double Shift Logical	.75
73	04	SSA	Single Shift Algebraic	.75
73	05	DSA	Double Shift Algebraic	.75
73	06	LSC	Load Shift and Count	1.125
73	07	DLSC	Double Load Shift and Count	2.125
73	10	LSSC	Left Single Shift Circular	.75
73	11	LDSC	Left Double Shift Circular	.75
73	12	LSSL	Left Single Shift Logical	.75
73	13	LDSL	Left Double Shift Logical	.75

7. Executive System Control Instructions

Instruction Code		Mnemonic	Instruction	Execution Times + in usec.
f	j			
72	11	ER	Executive Return	1.375
72	14	SCN	Store Channel Number	.75
72	15	LIF	Load Processor State Register	.75
72	16	LSL	Load Storage Limits Register	.75
73	14	ISI	Initiate Interprocessor Interrupt	.75
73	15	SIL	Select Interrupt Module	.75
73	16	LCR	Load Channel Select Register	.875

8. Unconditional Jump Instructions

Instruction Code		Mnemonic	Instruction	Execution Times + in usec.
f	j			
72	01	SLJ	Store Location and Jump	2.125
72	13	PAIJ	Prevent All Interrupts and Jump	.75
74	07	AAIJ	Allow All Interrupts and Jump	.75
74	13	LMJ	Load Modifier and Jump	.75

9. Conditional Jump Instructions

Instruction Code		Mnemonic	Instruction	Execution Times +* in usec.
f	j			
70	#	JGD	Jump on Greater and Decrement	.75/1.50
71	16	DJZ	Double Precision Zero Jump	.875/1.625
72	02	JPS	Jump on Positive and Shift	1.50/ .75
72	03	JNS	Jump on Negative and Shift	1.50/ .75
74	00	JZ	Jump on Zero	1.50/ .75
74	01	JNZ	Jump on Non Zero	1.50/ .75
74	02	JP	Jump on Positive	1.50/ .75
74	03	JN	Jump on Negative	1.50/ .75
74	04	JK,J	Jump on Keys, Jump	.75
74	05	HKJ,HJ	Halt on Keys and Jump, Halt and Jump	.75
74	10	JNB	Jump on No Low Bit	1.50/ .75
74	11	JB	Jump on Low Bit	1.50/ .75
74	12	JMGI	Jump Modifier Greater and Increment	1.625 .75
74	14	JO	Jump on Overflow	.75/1.50
74	15	JNO	Jump on No Overflow	.75/1.50
74	16	JC	Jump on Carry	.75/1.50
74	17	JNC	Jump on No Carry	.75/1.50
75	02	JIC	Jump on Input Channel Busy	.75
75	06	JOC	Jump on Output Channel Busy	.75
75	12	JFC	Jump on Function in Channel	.75

The j and a designators together serve to specify any of 128 control registers.

10. Logical Instructions

Instruction Code		Mnemonic	Instruction	Execution Times + in usec.
f	j			
40	0–17	OR	Logical OR	.75
41	0–17	XOR	Logical Exclusive OR	.75
42	0–17	AND	Logical AND	.75
43	0–17	MLU	Masked Load Upper	.75

11. Test (or Skip) Instructions:

Instruction Code		Mnemonic	Instruction	Execution Times +* in usec.
f	j			
44	0-17	TEP	Test Even Parity	1.875/1.25
45	0-17	TOP	Test Odd Parity	1.875/1.25
47	0-17	TLEM,TNGM	Test Less or Equal to Modifier	1.75/1.00
50	0-17	TZ	Test for Zero	1.625/.875
51	0-17	TNZ	Test for Non Zero	1.625/.875
52	0-17	TE	Test for Equal	1.625/.875
53	0-17	TNE	Test for Not Equal	1.625/.875
54	0-17	TLE,TNG	Test for Less or Equal	1.625/.875
55	0-17	TG	Test for Greater	1.625/.875
56	0-17	TW	Test for Within Range	1.75/1.00
57	0-17	TNW	Test for Not Within Range	1.75/1.00
60	0-17	TP	Test for Positive	1.625/.875
61	0-17	TN	Test for Negative	1.625/.875
71	17	DTE	Double Precision Test Equal	2.375/1.625

12. Input/Output Instructions

Instruction Code		Mnemonic	Instruction	Execution Times + in usec.
f	j			
75	00	LIC	Load Input Channel	.75
75	01	LICM	Load Input Channel and Monitor	.75
75	02	JIC	Jump on Input Channel Busy	.75
75	03	DIC	Disconnect Input Channel	.75
75	04	LOC	Load Output Channel	.75
75	05	LOCM	Load Output Channel and Monitor	.75
75	06	JOC	Jump on Output Channel Busy	.75
75	07	DOC	Disconnect Output Channel	.75
75	10	LFC	Load Function in Channel	.75
75	11	LFCM	Load Function in Channel and Monitor	.75
75	12	JFC	Jump on Function in Channel	.75
75	13	AFC	Allow Function in Channel	.75
75	14	AACI	Allow All Channel Interrupts	.75
75	15	PACI	Prevent All Channel Interrupts	.75

13. Index Register Instructions

Instruction Code		Mnemonic	Instruction	Execution Times +* in usec.
f	j			
06	0-15	SX	Store X	.75
24	0-17	AX	Add to X	.75
25	0-17	ANX	Add Negative to X	.75
26	0-17	LXM	Load X Modifier	.75
27	0-17	LX	Load X	.75
46	0-17	LXI	Load X Increment	.875
47	0-17	TLEM,TNGM	Test Less or Equal to Modifier	1.75 /1.00
74	12	JMGI	Jump Modifier Greater and Increment	1.625/ .75
74	13	LMJ	Load Modifier and Jump	.75

14. Miscellaneous Instructions

Instruction Code		Mnemonic	Instruction	Execution Times + in usec.
f	j			
72	10	EX	Execute	.75
74	06	NOP	No Operation	.75

APPENDIX B. INSTRUCTION REPERTOIRE (By Function Code)

Instruction Code		Mnemonic	Instruction	Description	Execution Time + in usec.
f	j				
00	-	-	Illegal Code		-
01	0-15	SA	Store A	$(A) \rightarrow U$	.75
02	0-15	SN SNA	Store Negative A	$-(A) \rightarrow U$	.75
03	0-15	SM SMA	Store Magnitude A	$\|(A)\| \rightarrow U$	.75
04	0-15	SR	Store R	$(R_a) \rightarrow U$	.75
05	0-15	SZ	Store Zero	Zeroes → U	.75
06	0-15	SX	Store X	$(X_a) \rightarrow U$	.75
07	–	–	Illegal Code		–
10	0-17	LA	Load A	$(U) \rightarrow A$	.75
11	0-17	LN LNA	Load Negative A	$-(U) \rightarrow A$	.75 –
12	0-17	LM LMA	Load Magnitude A	$\|(U)\| \rightarrow A$	.75
13	0-17	LNMA	Load Negative Magnitude A	$-\|(U)\| \rightarrow A$	.75
14	0-17	AA	Add to A	$(A) + (U) \rightarrow A$	.75
15	0-17	ANA	Add Negative A	$(A) - (U) \rightarrow A$	.75
16	0-17	AM AMA	Add Magnitude to A	$(A) + \|(U)\| \rightarrow A$	.75
17	0-17	ANM ANMA	Add Negative Magnitude to A	$(A) - \|(U)\| \rightarrow A$	.75
20	0-17	AU	Add Upper	$(A) + (U) \rightarrow A+1$	.75
21	0-17	ANU	Add Negative Upper	$(A) - (U) \rightarrow A+1$	.75
22	0-17	BT	Block Transfer	$(X_x + u) \rightarrow (X_a + u)$, repeat	1.50+ 1.5K
23	0-17	LR	Load R	$(U) \rightarrow R_a$	.75
24	0-17	AX	Add to X	$(X_a) + (U) \rightarrow X_a$	.75
25	0-17	ANX	Add Negative to X	$(X_a) - (U) \rightarrow X_a$	.75
26	0-17	LXM	Load X Modifier	$(U) \rightarrow X_{a\ 17-0}$	.875
27	0-17	LX	Load X	$(U) \rightarrow X_a$	.75
30	0-17	MI	Multiply Integer	$(A) \cdot (U) \rightarrow A, A+1$	2.375
31	0-17	MSI	Multiply Single Integer	$(A) \cdot (U) \rightarrow A$	2.375
32	0-17	MF	Multiply Fractional	$(A) \cdot (U) \rightarrow A, A+1$	2.375
33	–	–	Illegal Code		–

Instruction Code		Mnemonic	Instruction	Description	Execution Time + * in usec.
f	j				
34	0-17	DI	Divide Integer	$(A,A+1) \div (U) \rightarrow A$, Remainder $\rightarrow A+1$	10.125
35	0-17	DSF	Divide Single Fractional	$(A) \div (U) \rightarrow A+1$	10.125
36	0-17	DF	Divide Fractional	$(A,A+1) \div (U) \rightarrow A$, Remainder $\rightarrow A+1$	10.125
37	–	–	Illegal Code		–
40	0-17	OR	Logical OR	$(A) \vee (U) \rightarrow A+1$	.75
41 •	0-17	XOR	Logical Exclusive OR	$(A) \not\equiv (U) \rightarrow A+1$	.75
42	0-17	AND	Logical AND	$(A) \wedge (U) \rightarrow A+1$	.75
43	0-17	MLU	Masked Load Upper	$(U) \wedge (M) \vee A \wedge (M)' \rightarrow A+1$	.75
44	0-17	TEP	Test Even Parity	Skip if $(A) \wedge (U)$ is even parity	1.875/1.25
45	0-17	TOP	Test Odd Parity	Skip if $(A) \wedge (U)$ is odd parity	1.875/1.25
46	0-17	LXI	Load X Increment	$(U) \rightarrow X_{a\ 35-18}$	1.00
47	0-17	TLEM TNGM	Test Less or Equal to Modifier Test Not Greater than Modifier	Skip if $(X_a)_{17-0} \geq (U)$	1.75 /1.00
50	0-17	TZ	Test for Zero	Skip if $(U) = \pm 0$	1.625/ .875
51	0-17	TNZ	Test for Non Zero	Skip if $(U) \neq \pm 0$	1.625/ .875
52	0-17	TE	Test for Equal	Skip if $(A) = (U)$	1.625/ .875
53	0-17	TNE	Test for Not Equal	Skip if $(A) \neq (U)$	1.625/ .875
54	0-17	TLE TNG	Test for Less or Equal Test for Not Greater	Skip if $(U) \leq (A)$	1.625/ .875
55	0-17	TG	Test for Greater	Skip if $(U) > (A)$	1.625/ .875
56	0-17	TW	Test for Within Range	Skip if $(A) < (U) \leq (A+1)$	1.75 /1.00
57	0-17	TNW	Test for Not Within Range	Skip if $(U) \leq (A)$ or $(U) > (A+1)$	1.75 /1.00
60	0-17	TP	Test for Positive	Skip if $(U)_{35} = 0$	1.625/ .875
61	0-17	TN	Test for Negative	Skip if $(U)_{35} = 1$	1.625/ .875
62	0-17	SE	Search for Equal	Skip if $(U) = A$, repeat	2.25 + .75K
63	0-17	SNE	Search for Not Equal	Skip if $(U) \neq A$, repeat	2.25 + .75K
64	0-17	SLE SNG	Search for Less or Equal Search for Not Greater	Skip if $(U) \leq (A)$, repeat	2.25 + .75K
65	0-17	SG	Search for Greater	Skip if $(U) > (A)$, repeat	2.25 + .75K

Instruction Code		Mnemonic	Instruction	Description	Execution Time + * in usec.
f	j				
66	0-17	SW	Search for Within Range	Skip if $(A) < (U) \leq (A+1)$, repeat	2.25 + .75K
67	0-17	SNW	Search for Not Within Range	Skip if $(U) \leq (A)$ or $(U) > (A+1)$, repeat	2.25 + .75K
70	†	JGD	Jump on Greater and Decrement	Jump to U if $(JA) > 0$, then $(JA) - 1 \rightarrow JA$	.75/ 1.50
71	00	MSE	Mask Search for Equal	Skip if $(U) \wedge (M) = (A) \wedge (M)$, repeat,	2.25 + .75K
71	01	MSNE	Mask Search for Not Equal	Skip if $(U) \wedge (M) \neq (A) \wedge (M)$, repeat	2.25 + .75K
71	02	MSLE MSNG MDNH	Mask Search for Less or Equal Mask Search for Not Greater	Skip if $(U) \wedge (M) \leq (A) \wedge (M)$, repeat	2.25 + .75K
71	03	MSG	Mask Search for Greater	Skip if $(U) \wedge (M) > (A) \wedge (M)$, repeat	2.25 + .75K
71	04	MSW	Masked Search for Within Range	Skip if $(A) \wedge (M) < (U) \wedge (M) \leq (A+1) \wedge (M)$, repeat	2.25 + .75K
71	05	MSNW	Masked Search for Not Within Range	Skip if $(U) \wedge (M) \leq (A) \wedge (M)$ or $(U) \wedge (M) > (A+1) \wedge (M)$, repeat	2.25 + .75K
71	06	MASL	Masked Alphanumeric Search for Less or Equal	Skip if $(U) \wedge (M) \leq (A) \wedge (M)$, repeat	2.25 + .75K
71	07	MASG	Masked Alphanumeric Search for Greater	Skip if $(U) \wedge (M) > (A) \wedge (M)$, repeat	2.25 + .75K
71	10	DA	Double Precision Fixed Point Add	$(A,A+1) + (U,U+1) \rightarrow A,A+1$	1.625
71	11	DAN	Double Precision Fixed Point Add Negative	$(A,A+1) - (U,U+1) \rightarrow A,A+1$	1.625
71	12	DS	Double Store A	$(A,A+1) \rightarrow U,U+1$	1.50
71	13	DL	Double Load A	$(U,U+1) \rightarrow A,A+1$	1.50
71	14	DLN	Double Load Negative	$-(U,U+1) \rightarrow A,A+1$	1.50
71	15	DLM	Double Load Magnitude	$\|(U,U+1)\| \rightarrow A,A+1$	1.50
71	16	DJZ	Double Precision Zero Jump	Jump to U if $(A,A+1) = \pm 0$	.875/1.625
71	17	DTE	Double Precision Test Equal	Skip if $(U,U+1) = (A,A+1)$	2.375/1.625
72	00		Illegal Code		–
72	01	SLJ	Store Location and Jump	$(P) \rightarrow U_{17-0}$, jump to U+1	2.125
72	02	JPS	Jump on Positive and Shift	if $(A)_{35} = 0$, jump to U and shift (A) left one position	1.50 / .75
72	03	JNS	Jump on Negative and Shift	if $(A)_{35} = 1$, jump to U and shift (A) left one position	1.50 / .75
72	04	AH	Add Halves	$(A)_{17-0} + (U)_{17-0} \rightarrow (A)_{17-0}$, $(A)_{35-18} + (U)_{35-18} \rightarrow A_{35-18}$	.75

† *The j and a designators together serve to specify any of the 128 control registers.*

Instruction Code		Mnemonic	Instruction	Description	Execution Time + in usec.
f	j				
72	05	ANH	Add Negative Halves	$(A)_{17-0} - (U)_{17-0} \rightarrow A_{17-0}$ $(A)_{35-18} - (U)_{35-18} \rightarrow A_{35-18}$	.75
72	06	AT	Add Thirds	$(A)_{35-24} + (U)_{35-24} \rightarrow A_{35-24}$ $(A)_{23-12} + (U)_{23-12} \rightarrow A_{23-12}$ $(A)_{11-0} + (U)_{11-0} \rightarrow A_{11-0}$	.75
72	07	ANT	Add Negative Thirds	$(A)_{35-24} - (U)_{35-24} \rightarrow A_{35-24}$ $(A)_{23-12} - (U)_{23-12} \rightarrow A_{23-12}$ $(A)_{11-0} - (U)_{11-0} \rightarrow A_{11-0}$	.75
72	10	EX	Execute	Execute the instruction at U	.75
72	11	ER	Executive Return	PSR→ X0, Guard Mode off, program lockin off, clear addressing mode, interrupt to location 162	1.375
72	12		Illegal Code		
72	13	PAIJ	Prevent All Interrupts and Jump	Disable all Interrupts and Jump to U	.75
72	14	SCN	Store Channel Number	Channel Number → U †	.75
72	15	LIF	Load Processor State Register	(U) → PSR	.75
72	16	LSL	Load Storage Limits Register	(U) → Program Lockin Register	.75
72	17		Illegal Code		
73	00	SSC	Single Shift Circular	Shift (A) right U places circularly	.75
73	01	DSC	Double Shift Circular	Shift (A,A+1), right U places circularly	.75
73	02	SSL	Single Shift Logical	Shift (A) right U places, fill zeros	.75
73	03	DSL	Double Shift Logical	Shift (A,A+1) right U places, fill zeros	.75
73	04	SSA	Single Shift Algebraic	Shift (A) right U places, sign fill	.75
73	05	DSA	Double Shift Algebraic	Shift (A,A+1) right U places, sign fill	.75
73	06	LSC	Load Shift and Count	(U) → A, Shift A left circularly until $A_{35} \neq A_{34}$ or until A has been shifted 35 times. Store the result in A and the number of shifts in A+1.	1.125
73	07	DLSC	Double Load Shift and Count	Same as single shift except except(U,U+1)→A,A+1, result in A,A+1 shift count in A+2	2.125

† U_{17-0} if main storage

U_{35-0} if control register

Instruction Code		Mnemonic	Instruction	Description	Execution Time + in usec.
f	j				
73	10	LSSC	Left Single Shift Circularly	Shift (A) left U places, circularly	.75
73	11	LDSC	Left Double Shift Circularly	Shift (A,A+1) left U places, circularly	.75
73	12	LSSL	Left Single Shift Logical	Shift (A) left U places, fill zeros	.75
73	13	LDSL	Left Double Shift Logical	Shift (A,A+1) left U places, fill zeros	.75
73	14	ISI	Initiate Synchronous Interrupt	Initiate synchronous interrupt on channel A	.75
73	15	SIL	Select Interrupt Module	If A = 0, assign interrupt locations to store module 1; if A = 1, assign.interrupt locations to store module 2 if A = 2, assign interrupt locations to store module 3 if A = 3, assign interrupt locations to store module 4	.75
73	16	LCR	Load Channel Select Register	$(U)_{3-0} \rightarrow$ CSR	.875
73	17		Illegal Code		
74	00	JZ	Jump on Zero	If (A) = ± 0, jump to U	1.50 / .75
74	01	JNZ	Jump on Non Zero	If (A) ± 0, jump to U	1.50 / .75
74	02	JP	Jump on Positive	If $(A)_{35} = 0$, jump to U	1.50 / .75
74	03	JN	Jump on Negative	If $(A)_{35} = 1$, jump to U	1.50 / .75
74	04	JK J	Jump on Keys Jump	If A = key which is set, A = 0 jump to U	.75
74	05	HKJ HJ	Halt on Keys and Jump Halt and Jump	Stop if A = 0, or if A $\wedge$ key setting = 0	.75
74	06	NOP	No Operation	Proceed to next instruction	.75
74	07	AAIJ	Allow all Interrupts and Jump	Enable all interrupts and Jump to U	.75
74	10	JNB	Jump on No Low Bit	If $(A)_0 = 0$, jump to U	1.50 / .75
74	11	JB	Jump on Low Bit	If $(A)_0 = 1$, jump to U	1.50 / .75
74	12	JMGI	Jump Modifier Greater and Increment	If $(X_a)_{17-0} > 0$, jump to U	1.625/ .75
74	13	LMJ	Load Modifier and Jump	$(P) \rightarrow X_{a17-0}$, jump to U	.75
74	14	JO	Jump on Overflow	Jump to U if o.f. designator set	.75 /1.50
74	15	JNO	Jump on No Overflow	Jump to U if o.f. designator not set	.75 /1.50
74	16	JC	Jump on Carry	Jump to U if carry designator set	.75 /1.50
74	17	JNC	Jump on No Carry	Jump to U if carry designator not set	.75 /1.50

Instruction Code		Mnemonic	Instruction	Description	Execution Time +* in usec.
f	j				
75	00	LIC	Load Input Channel	(U) → input control word A, initiate input mode on channel A	.75
75	01	LICM	Load Input Channel and Monitor	(U) → input control word A, initiate input mode on channel A with monitor	.75
75	02	JIC	Jump on Input Channel Busy	If channel A in input mode, jump to U	.75
75	03	DIC	Disconnect Input Channel	Terminate input mode on channel A	.75
75	04	LOC	Load Output Channel	(U) → output control word A, initiate output mode on channel A	.75
75	05	LOCM	Load Output Channel and Monitor	(U) → output control word A, initiate output mode on channel A with monitor	.75
75	06	JOC	Jump on Output Channel Busy	If channel A is in output mode, jump to U	.75
75	07	DOC	Disconnect Output Channel	Terminate output mode on channel A	.75
75	10	LFC	Load Function in Channel	(U) → output control word A, and initiate Function mode on channel A	.75
75	11	LFCM	Load Function in Channel and Monitor	(U) → output control word A, and initiate Function mode on channel A with monitor	.75
75	12	JFC	Jump on Function in Channel	If channel A is in function mode, jump to U	.75
75	13	AFC	Allow Function in Channel	Request external function or output word on channel A	.75
75	14	AACI	Allow All Channel Interrupts	All external interrupts are allowed	.75
75	15	PACI	Prevent All Channel Interrupts	All external interrupts are disabled	.75
76	00	FA	Floating Add	(A) + (U) → A,A+1	1.75
76	01	FAN	Floating Add Negative	(A) − (U) → A,A+1	1.75
76	02	FM	Floating Multiply	(A) · (U) → A,A+1	2.625

Instruction Code		Mnemonic	Instruction	Description	Execution Time + in usec.
f	j				
76	03	FD	Floating Divide	$(A) \div (U) \rightarrow A$, Remainder $A+1$	8.25
76	04	LUF	Load and Unpack Floating	Unpack (U), store fixed-point part in $A+1$ and store biased exponent in A_{7-0}	.75
76	05	LCF	Load and Convert to Floating	Normalize (U), pack with biased exponent from (A) and store at $A+1$	1.125
76	06	MCDU	Magnitude of Characteristic Difference to Upper	$\lvert\lvert(A)_{35-27}\rvert - \lvert(U)_{35-27}\rvert\rvert \rightarrow A+1$	.75
76	07	CDU	Characteristic Difference to Upper	$\lvert(A)_{35-27}\rvert - \lvert(U)_{35-27}\rvert \rightarrow A+1$	.75
76	10	DFA	Double Precision Floating Add	$(A,A+1) + (U,U+1) \rightarrow A,A+1$	2.625
76	11	DFAN	Double Precision Floating Add Negative	$(A,A+1) - (U,U+1) \rightarrow A,A+1$	2.625
76	12	DFM	Double Precision Floating Multiply	$(A,A+1) \cdot (U,U+1) \rightarrow A,A+1$	4.250
76	13	DFD	Double Precision Floating Divide	$(A,A+1) \div (U,U+1) \rightarrow A,A+1$	17.25
76	14	DFU	Double Load & Unpack Floating	Unpack $(U,U+1)$: fixed-point part $\rightarrow A+1,A+2$, exponent $\rightarrow A_{10-0}$	1.50
76	15	DFP	Double Load & Convert to Floating	Normalize and pack from fixed-point part in $(U,U+1)$, exponent in $(A)_{10-0}$ and store in $A+1, A+2$	2.125
76	16	FEL	Floating Expand and Load	$(U)_{1,8,27} \rightarrow A,A+1_{1,11,60}$	1.00
76	17	FCS	Floating Compress and Store	$(A,A+1)_{1,11,60} \rightarrow U_{1,8,27}$	1.625
77	0-17		Illegal Code		–

+ *Times are given for alternate bank case only. In cases where instruction and operand are procured from same bank, add .75 usec. to execution time.*

* *For all comparison instructions, the first number represents the skip or jump condition, the second number is for no skip or no jump condition. For function codes 01-67, add .375 usec. to execution times for 6-bit and 12-bit writes.*

APPENDIX C. REAL TIME PROGRAMMING FEATURES

The following section describes some of the programming features of the UNIVAC 1108 System which are used with the UNIVAC Standard Communication Subsystem. It assumes that the reader is familiar with UNIVAC 1108 programming techniques as well as with the Executive routine. Special emphasis is placed on the composition of computer words for use with the Subsystem as well as on the various arrangements of input/output buffers which may be used. Other areas such as checking and polling have not been included since they vary from one application to the next.

UNIVAC 1108 EXTERNALLY SPECIFIED INDEX (ESI) OPERATIONS

1. ESI Access-Control Word

The ESI control word has a format very similar to the normal I/O Access-Control Word.

G	H	W	V
35 34	33	32 18	17 0

V = Data Address (18 Bits)
W = Character Count (15 Bits)

H	Character Location
0	Lower half of V
1	Upper half of V

G	
00	If H = 1 when the Access-Control Word is referenced for an input/output transfer, increase V by 1.
10	If H = 0 when the Access-Control Word is referenced for an input/output transfer, decrease V by 1.
01 11	No change in V

2. ESI Data Word

SECOND CHARACTER	FIRST CHARACTER
35 18	17 0

Two data characters can be stored in each 36-bit word. Each character of an input or output message will cause the appropriate ESI Access-Control Word to be read out of main storage. If H = 0 the character will be transferred to or from the lower half of the V location and to the upper half if H = 1. The H-bit is changed and the ESI Access-Control Word is stored back in main storage. This causes H to alternate between one and zero.

3. Buffer Mode Data Transfers

A buffer mode transfer, which occurs independent of main program control, is used to transfer data betw UNIVAC 1108 storage and the Standard Communication Subsystem.

Before execution of a buffer mode transfer of data, the program must perform the following steps:

a. Load the address locations specified by the Externally Specified Index addresses with Access-Control Words.

b. Activate the channel to be used for the data transfer.

c. Send the proper Function Word to the Standard Communications Subsystem. (This step is not required to activate data transfers from Low or Medium speed CLT's).

Step b is accomplished with one of the Initiate Input/Output instructions. The Access-Control Word which is specified by this instruction should designate a one word dummy buffer, since this buffer is not used in the ESI mode.

LIC	Load Input Channel
LICM	Load Input Channel and Monitor
LOC	Load Output Channel
LOCM	Load Output Channel and Monitor

Step c is performed by the Initiate Function Transfer (f = 75, j = 10). When in the ESI mode, the Initiate Function Transfer instruction operates differently. It loads the control word for the function into the Output Access-Control Register for the channel and automatically forces one External Function transfe

Data is then transferred between the main storage and the Standard Communications Subsystem without main program intervention. When a word is transferred to or from storage, one is automatically subtract from the Character Count (W) of the Access-Control Word. When the Character Count becomes zero, the computer will generate an Internal Interrupt.

APPENDIX D. MATH - PACK ROUTINES

A. INTERPOLATION

1. Gregory-Newton Method
2. Central Differences
3. Stirling's Interpolation
4. Lagrange Interpolation
5. Spline Interpolation
6. Aitken's Method

B. SOLUTION OF EQUATIONS

1. Wegstein Iteration
2. Roots (Real and Complex) of Polynomial
3. Roots of Complex Polynomial
4. Coefficients of Polynomial given its roots
5. Newton's Method
6. Aitken's Method

C. DIFFERENTIATION

1. First derivative approximation
2. Second Derivative approximation
3. Nth derivative of a polynomial

D. MATRIX MANIPULATION

1. Addition (Real)
2. Addition (Complex)
3. Subtraction (Real)
4. Subtraction (Complex)
5. Multiplication (Real)
6. Multiplication (Complex)
7. Transpose (Real)
8. Transpose (Complex)
9. Inversion (Real)
10. Inversion (Complex)
11. Multiplication by Scalar (Real)
12. Multiplication by Scalar (Complex)
13. Inversion (Double-Precision)
14. Diagonalization
15. Polynomial matrix triangularization
16. Determinant (Real)
17. Determinant (Complex)
18. Eigenvector of complex matrix
19. Reduction of symmetric matrix to tridiagonal form
20. Eigenvalues of symmetric matrix (Sturm Sequences)
21. Eigenvalues of symmetric matrix (Wilkenson's method)
22. Matrix rotation
23. Complex eigenvalues of real, complex matrix
24. Multiplication by diagonal matrix stored as vector
    Rank (Real)
    Rank (Complex)

E. NUMERICAL INTEGRATION

1. Maximum step trapezoidal rule
2. Maximum step Simpson's 1/3 rule
3. Maximum step Simpson's 3/8 rule
4. Variable step
5. Double integration

F. ORDINARY DIFFERENTIAL EQUATIONS

1. Runge-Kutta scheme
2. Modified Euler method
3. Milne's method
4. Nth order system expressed as system of N first order equations
5. Second order equations

G. SYSTEMS OF EQUATIONS

1. Iterative solution (Jacobi)
2. Gauss-Jordon Solution
3. Iteration to improve accuracy of root of set of linear equations
4. Functional iteration for sets of non-linear equations

H. MISCELLANEOUS

1. Fourier series
2. Fourier transform
3. Random number generator
4. Arctangent subroutine
5. Bessel function
6. Legendre function
7. Gamma function
8. Orthogonal polynomial least squares curve fitting
9. Polynomial evaluation
10. Multiplication of two polynomials (Real)
11. Multiplication of two polynomials (Complex)

APPENDIX E. BEEF DATA PROCESSING ROUTINES

A. WHOLE WORD DATA MOVEMENT

1. MOVEKA — Moves the contents of a word into an array.
2. MOVEKL — Moves the contents of a word into a list specified in the calling sequence.
3. MOVEAA — Moves whole words from one array to another.
4. MOVEAL — Moves whole words from an array to a list specified in the calling sequence.
5. MOVELA — Moves whole words from a list specified in the calling sequence to an array.
6. MOVELL — Moves one list of words to another.

B. CHARACTER AND FIELD MOVEMENT

1. IXTRAC — (Function) Selects a character from an array and creates a variable FORTRAN integer.
2. INSERT — Replaces a character with the low order bits of a FORTRAN integer variable.
3. MOVECH — Transfers a field of characters from one array to another.
4. EXTRAC — (Function) Selects a character from an array and creates a Hollerith variable.
5. ENSERT — Places a single Hollerith character into an array.

C. FORMATTING

1. MOVMCH — Transfers several fields of characters from one array to another.
2. MOVECC — Packs or unpacks several fields of characters from one array to another.
3. MOVEHA — Defines Hollerith constants which may be used as FORTRAN variables.
4. HEDSET — Sets up to allow automatic writing of headings on reports.
5. HEDING — Allows the automatic writing of headings on reports.
6. CLKNDT — Transfers the date and or time as FORTRAN integers.
7. DATBCD — Transfers the date as Hollerith characters.

D. DECISION MAKING

1. CMPAA – Compares logically or arithmetically a number of words of two arrays.
2. CMPAL – Compares logically or arithmetically a number of words of an array and a list.
3. CMPLL – Compares logically or arithmetically a number of words of two lists.
4. CMPKA – Determines if a word exists in a number of words of an array.
5. CMPKL – Determines if a word exists in a list.
6. FINDKA – Searches an array for a one word match.
7. FINDKL – Searches a list for a one word match.

E. DATA CONVERSION

1. BCD21 – Converts a Hollerith number into a FORTRAN integer.
2. DECIN – Creates a FORTRAN integer from a Hollerith number (two word)
3. DECN – Creates a FORTRAN integer from a Hollerith number (one word)
4. AAADEC – Creates a signed Hollerith field from a FORTRAN integer.
5. AADEC – Creates a signed, edited Hollerith field from a FORTRAN integer.
6. LEDZER – Converts leading of a word to blanks.

F. REPORT CONTROL

1. RPTFLT – Real variable report generator control.
2. RPTFIX – Integer variable report generator control.

G. INPUT/OUTPUT

1. BSFILE – Backspace a file
2. UNLOAD – Rewind FORTRAN unit with interlock.
3. EOFBIN – End-of-file, binary
4. EOFILE – End-of-file, Hollerith
5. IOPACK – Blocking and Buffering routine for fixed and variable tape records.
6. COPY – Copy a print file to the printer

H. SORTING

1. BCDSR – Internal commercial and arithmetic sort

I. COMPATIBILITY

1. IXTRAK – (Function) Selects a character from an array, converts it to scientific sequence, and creates a variable FORTRAN integer.

2. INSERK – Replaces a character with the low order bits of a FORTRAN integer variable after converting it to Field Data from scientific sequence.

3. EAM – Creates a word containing the positional representation of the commercial sequence of the characters of a word.

4. EDP – Restores the word of the EAM routine.

5. CMPEAM – Compares two arrays for commercial sequence.

6. KMPWL – Compares word fields in two arrays for Field Data sequence.

7. KMPWA – Compares word fields in two arrays for Algebraic sequence.

8. KMPCL – Compares character fields in two arrays for Field Data sequence.

9. KMPCA – Compares character fields in two arrays for Algebraic sequence.

10. KMPWC – Compare word fields in two arrays for commercial sequence.

11. KMPWS – Compares word fields in two arrays for scientific sequence.

12. KMPCC – Compares character fields in two arrays for commercial sequence.

13. KMPCS – Compares character fields in two arrays for scientific sequence.

14. LOGSET } – Compares (EAM) two strings of Hollerith
    LAS characters.

15. JC – Defines a FORTRAN constant.

16. JD – Divides one FORTRAN constant by another.

17. JM – Multiplies one FORTRAN constant by another.

18. ENDFIL – Mark end-of-file on FORTRAN unit.

19. REWIND – Rewinds FORTRAN unit.

20. XFER – Alternate call to MOVECH

APPENDIX F. STAT-PACK ROUTINES

I. DESCRIPTIVE STATISTICS

A. FREQUENCY DISTRIBUTIONS

1. Frequency Polygon	(FREQP)
2. Histogram	(HIST)
3. Multivariate Histogram	(MHIST)
4. Grouping Data	(GROUP)

B. MEASURES OF LOCATION AND DISPERSION

1. Arithmetic Mean	(AMEAN)
2. Geometric Mean	(GMEAN)
3. Harmonic Mean	(HMEAN)
4. Median	(MEDIAN)
5. Mode	(MODE)
6. Quantiles	(QUANT)
7. Distribution Curve	(OGIVE)
8a. Interpercentile Range	(IQRNG)
8b. Range	(RANGE)
9. Mean Deviation	(MNDEV)
10. Standard Deviation	(STDEV)
11. Coefficient of Variation	(CVAR)
12. Ordering and Ranking	(ORDER)

C. MOMENTS AND CUMULANTS

1. Moments	(CMONT)
2. Absolute Moments	(AMOMT)
3a. Cumulants from the Means	(CUMLT)
3b. Cumulants from Data	(CUM1)
4. Corrections for Groupings (Sheppard's Corrections)	(SHPCOR)
5a. Skewness and Kurtosis (from moments)	(KURSK)
5b. Skewness and Kurtosis (from data)	(SKUR)

D. STANDARD DISTRIBUTIONS

1. Binomial	(BINOM)
2. Poisson	(POISON)
3. Hypergeometric	(HYPER)
4. Normal	(PNORM)
5. Pearson type Distributions	(PDIST)
6. Gram-Charlier Series	(GRACH)
7. Edgeworth Series	(EDSER)

II. TESTS ON STATISTICAL PARAMETERS

A. CHI-SQUARED TEST

1. Test of Estimate of the Probability of an event from one sample (CHI21S)
2. Test of Estimate of the Probability of an event from many samples (CHF2JS)
3. Test of Fit to Poisson Distribution (CHI2P)
4. Test of Fit to Normal Distribution (CHI2N)
5. Test of Whether a Number of Samples come from the same population (CHISAM)
6. Tests of Contingency Tables (CHICNT)
7. Generalized Goodness of Fit Test (GENGOF)

B. SIGNIFICANCE TESTS

1. Significance Test for Population Proportion (SIGPRP)
2. Significance Test for Mean of Normal Population (SIGMN)
3. Significance Test for the Difference in Means of Two Samples (SIGDMN)
4. Significance Test for the Difference in Variance of Normal Population (SIGDVR)

C. CONFIDENCE INTERVALS

1. Confidence Interval for Mean (Known Variance) (CFDMKV)
2. Confidence Interval for Mean (Unknown Variance) (CRDMUV)
3. Confidence Interval for Means (Same Unknown Variance) (CFDMSU)
4. Confidence Interval for Variance (Normal Population) (CFDVAR)
5. Tolerance Interval (TOLINT)

III. ANALYSIS OF VARIANCE

A. ARRAYS

1. One-Way Arrays (ANOV1)
2. Two-Way Arrays (ANOV2)
3. Three and Higher Order Arrays (ANOV3)

B. MISSING DATA (MISDAT)

C. VARIATE TRANSFORMATIONS (VTRANS)

D. RANDOMIZED BLOCKS (ANOVRB)

E. LATIN SQUARES (ANOVLS)

F. SPLIT PLOT (ANOVSP)

G. SPLIT SPLIT PLOT (ANOVSS)

H. NESTED DESIGNS (ANOVN)

I. FACTORIAL DESIGNS (ANOVFC)

J. ANALYSIS OF COVARIANCE (ANOCO)

K. GENERAL LINEAR HYPOTHESIS (GLH)

IV. REGRESSION AND CORRELATION ANALYSIS

A. LINEAR REGRESSION

1. Stepwise Multiple Regression (RESTEM)
2. Back Solution Method (REBSOM)

B. CORRELATION

1. Correlation Analysis (CORAN)

V. ANALYSIS (FACTAN)

VI. TIME SERIES ANALYSIS

A. MOVING AVERAGES (MOVAVG)

B. DECOMPOSITION OF TIME SERIES

1. Cycle Eliminations (Shiskin Method) (SEASHI)
2. Trend Elimination
   a. Moving Average (WEMAV)
   b. Least Squares (TRELS)
2. Analysis of the Random Component
   a. Variate Differences (VADIME)
   b. Probabilistic Method (RCPROB)

C. FORECASTING TECHNIQUES

1. Autoregressive Model (TSFARG)
2. Trend-Cycle-Least Squared Method (Shiskin Method) (FORSKI)
3. Generalized Exponential Smoothing (GEXSMO)
4. Autocorrelation and Cross (Correlation Analysis) (AUXCOR)
5. Power Density Function (POWDEN)

VII. MULTIVARIATE ANALYSIS

A.	GENERALIZED VARIANCE	(GENVAR)
B.	HOTELLING'S DISTRIBUTION	(DISHOT)
C.	SIGNIFICANCE OF A SET OF MEANS	(SIGMNS)
D.	MAHALANOBOIS'S GENERALIZED D^2-	(SIGTMN)
E.	DISCRIMINANT ANALYSIS	

VIII. DISTRIBUTION FUNCTIONS

A.	NORMAL DISTRIBUTION	(NORM)
B.	INVERSE NORMAL DISTRIBUTION	(NORMIN)
C.	CHI-SQUARED	(CHI)
D.	STUDENT'S DISTRIBUTION	(STUD)
E.	FISHER'S F-DISTRIBUTION	(FISH)
F.	POISSON DISTRIBUTION	(POIS)
G.	BINOMIAL DISTRIBUTION	(BIN)
H.	HYPERGEOMETRIC DISTRIBUTION	(HYGEO)
I.	BETA FUNCTION	(BETA)
J.	GAMMA FUNCTION	(GAMMA)

IX. PLOTTING ROUTINES

A.	PLOT SINGLE LINE	(PLOT 1)