Standard

Standard Computer Corporation

IC-9000 PROCESSOR

PRELIMINARY FUNCTIONAL DESCRIPTION

Form 9001-2
Preliminary Draft

0.2 TABLE OF CONTENTS

TABLE OF CONTENTS

0.3 Introduction

The IC-9000 is a high-speed microprogrammed data processor possessing a high degree of versatility and computing power. MINIFLOW instructions (Ministeps) are considerably more powerful than order codes in most microprogrammed processors. The IC-9000 is designed to allow the systems programmer hands-on access at the microprogram level. Target language and target system translator logic allows the IC-9000 to efficiently emulate a broad spectrum of processing systems. High level statements in existing languages can be executed directly or complex macros can be designed for problem-oriented target languages. The IC-9000 processor is divided into two major functional elements. The Control Engine interprets target language instructions and performs logical manipulation and arithmetic operations on data in the Operating Engine. Figure 1.0 is a block diagram of the overall IC-9000 processor. Outstanding features of IC-9000 processor design include:

- o Two Ministeps can be executed simultaneously.
- o Hands-on microprogram capability using Control Memory read-write storage modules.
- o Flexible microprogram branch features, including relative branching for subroutine overlay and relocation in read-write store.
- o Extensive emulation capability with a Ministep repertoire designed for efficient execution of many target languages.
- o Multiple accumulators (32 general registers).
- o Automatic subroutine entry, stacking and return.
- o Character serial and BCD arithmetic capability.
- o Four input/output busses designed to interface with asynchronous peripheral devices.
- o Data masking and field extraction during arithmetic and logical operations.
- o Extensive shift capability, partially available during arithmetic and logical operations.
- o Program access to all registers and storage elements in the processor.

IC-9000 MINIFLOW instruction words are divided into two types, each 32 bits long. To effectively increase the processor throughput without sacrificing Control Memory storage efficiency, two Ministeps are read out of Control Memory at each clock time. If the two Ministeps are an Operating and a Control type, in that order, both will be executed simultaneously.

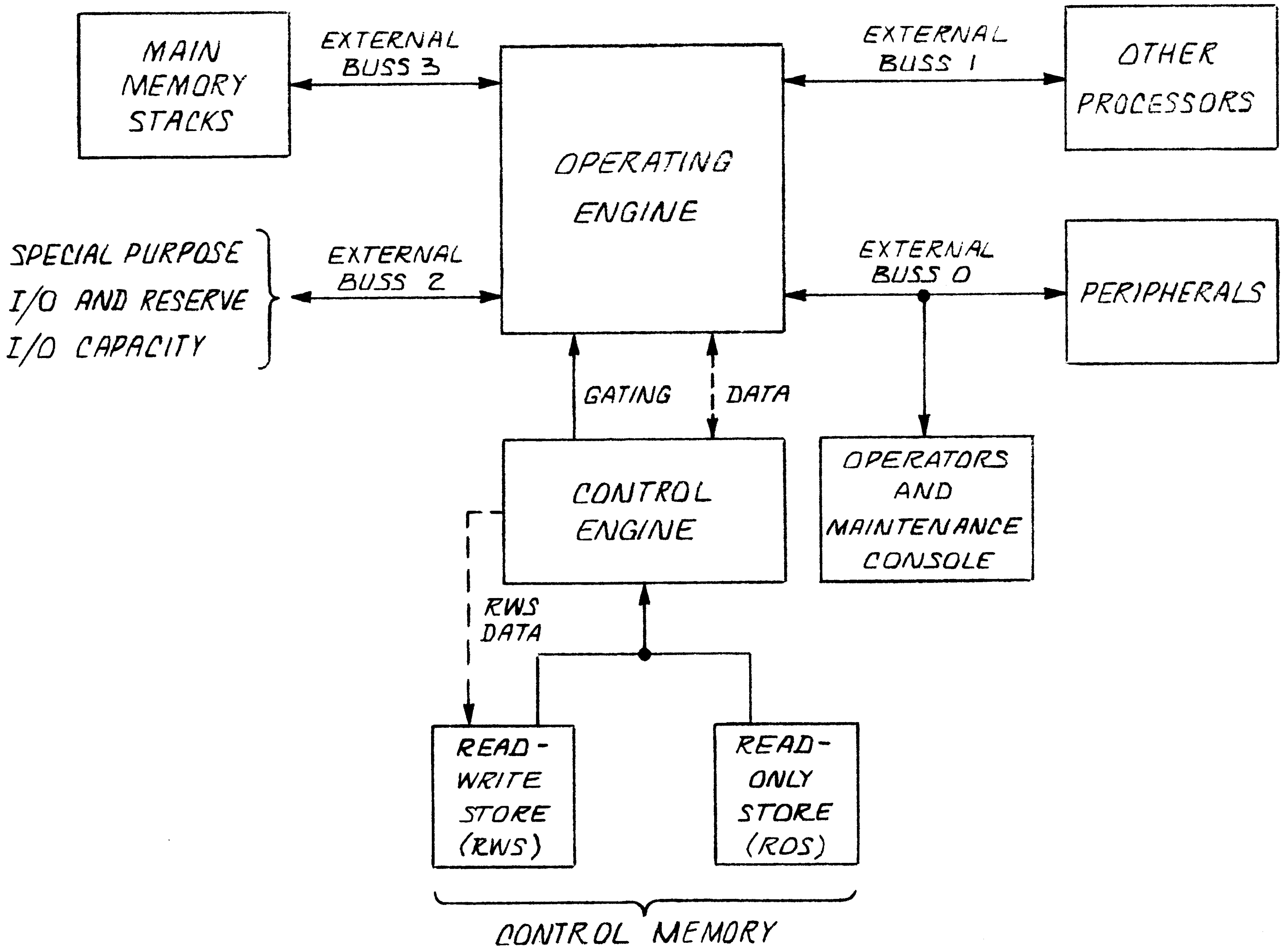

FIGURE 1.0 - IC-9000 OVERALL BLOCK DIAGRAM

1.0 IC-9000 PROCESSOR DESIGN

1.1 Operating Engine Design

Figure 1.1 is the functional block diagram of the IC-9000 Operating Engine. This Engine is organized as a two-address machine (except for data exchange operations, which specify data transfers into and out of the External Buss Registers). The Arithmetic Unit consists of two separate processing structures. The Parallel Arithmetic Unit operates on two 36-bit inputs from the General Registers in an arithmetic or logical capacity. A Decimal Arithmetic Unit performs similar functions serially on 8-bit bytes or decimal (BCD) characters. A high-speed multiply/divide unit and a double-word-length parallel arithmetic unit are available with the IC-9000 as options.

The Address Modification Unit (AMU) participates in data exchange operations between External Buss Registers, the General Register Stack and the Control Engine. Addresses and commands to Main Memory and other peripheral devices are formatted in the AMU and transmitted over the External Busses using the same type of logical operations and elements.

The Auxiliary Registers are intended for use as a high-speed buffer and for storing intermediate results or temporary data. Some Auxiliary Register addresses are assigned by hardware to facilitate communication with various registers and functional elements of the IC-9000 processor.

A double-word length arithmetic unit is available for more efficient handling of computations which require data fields greater than the 36-bit machine word length. Processing of data in fields less than the 36-bit machine word length is efficiently handled by MINIFLOW routines using the data mask feature and character serial arithmetic.

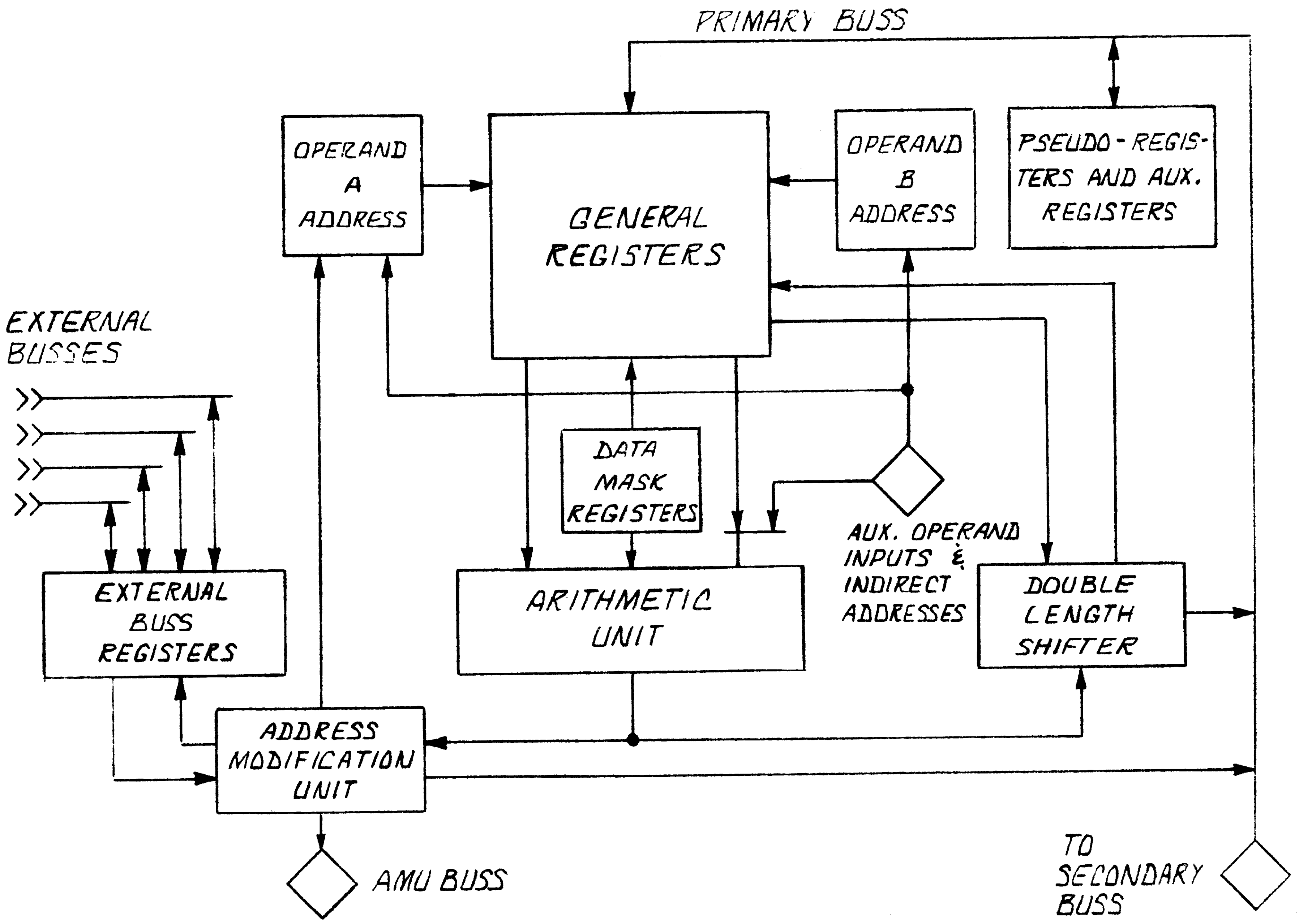

FIGURE 1.1 - OPERATING ENGINE BLOCK DIAGRAM

1.1.1 Arithmetic Unit

The Arithmetic Unit is the central focus of processing functions in the Operating Engine. It is composed of two functional elements, a high-speed Parallel Arithmetic Unit and a character serial Decimal Arithmetic Unit.

The Parallel Arithmetic Unit has 16 arithmetic and logical operations. By specifying "Long" and "Short" Immediate Data inputs (literal operands), a single register can be independently manipulated. The Parallel Arithmetic Unit is 36 bits wide and provides the following:

Table 1.1.1 Parallel Arithmetic Unit Operations

Arithmetic		Logical	
$A \leftarrow A+B$		$A \leftarrow A \bullet B$	(Logical Product)
$A \leftarrow A+\bar{B}+1$	(2's complement	$A \leftarrow A \bullet \bar{B}$	
$A \leftarrow \bar{A}+B+1$	subtract)	$A \leftarrow \bar{A} \bullet B$	
$A \leftarrow A+B+C_1$	(Conditional carry-	$A \leftarrow A \mathrm{U} B$	(OR)
$A \leftarrow A+\bar{B}+C_1$	in)	$A \leftarrow A \mathrm{U} \bar{B}$	
$A \leftarrow \bar{A}+B+C_1$		$A \leftarrow \bar{A} \mathrm{U} B$	
$A \leftarrow B$	(Clear and Add)	$A \leftarrow A \oplus B$	(Exclusive OR)
$A \leftarrow \bar{B}$	(1's Complement)	$A \leftarrow A \oplus \bar{B}$	(Compare)

The Decimal Arithmetic Unit operates on two pairs of packed BCD digits from the General Registers or from one of the Operand B auxiliary inputs. Two 8-bit bytes are independently specified from one of four positions in the lower 32 bits of a 36-bit word. After processing, the result is loaded into the Operand A address. The Decimal Arithmetic Unit checks both inputs for validity, adds or subtracts the operands, provides carry correction and stores the result. Decimal multiplication and division are performed by a MINIFLOW routine of iterated additions and subtractions.

Additionally, the full 8-bit character width may be used for binary operations, such as comparision and binary addition or subtraction. Other operations include two's complement addition and subtraction (two cases, A-B and B-A), replace (clear and add) and replace complement, AND (•), OR (U), and Exclusive OR (⊕). Characters less than eight bits wide may be processed using the data masking capability.

1.1.2 General Registers and Operand Addressing

The IC-9000 has 32 General Registers, each 36 bits wide. These registers interface with the Parallel and Decimal Arithmetic Units. The General Registers have two independent address structures. The Operand A register specified by a Ministep is gated to one input of the Arithmetic Unit and the Operand B input goes to the other. There are also several auxiliary inputs to the Operand B side. Both Operand A and Operand B addresses may be obtained by indirection, using Control Engine Index Registers.

Some of the General Registers are also gated to the Shifter. One half of the Shifter operates on the output of the Arithmetic Unit and the other with odd-numbered General Registers to provide a double-word-length shift capability.

In the "Non-clear" mode of data masking operations, Data Mask logic prevents "masked out" bits from being read into the General Registers. This feature allows a considerable degree of flexibility in the manipulation of partial word fields in the IC-9000. An inhibit function is also provided by the TEST bit of the GEneral ARithmetic (GEAR) Ministep. When this bit is a one, the result of the operation goes to the Primary Buss but is not read into the Operand A register. This feature allows testing Arithmetic Unit status outputs without modifying data in the General Registers.

1.1.3 Auxiliary Registers and Pseudo-registers

In addition to the General Registers, the Operating Engine has a group of up to 112 full-word-length Auxiliary Registers. These registers may be used as a high-speed buffer or for temporary storage.

16 other locations in the same addressing structure are used to access other storage elements and data sources in the IC-9000 Operating Engine. Dedicated locations are called Pseudo-registers. They provide for:

- o Loading and reading from the Data Mask Registers.
- o Changing protection keys and memory mapping data in the Address Modification Unit.
- o Direct loading and readout of the External Buss Registers.
- o Reading data from the 16-bit Secondary Buss (into either bits 4-19 or 20-35 of the Operating Engine).

1.1.4 Data Mask Registers

Data Masks are used to facilitate selective modification of specific bits or fields of an Operating Engine word. One Data Mask Register is accessed at each execution of an Operating Ministep. Bits in the Mask Register which contain a one are defined as masked-in bits. Bits containing a zero are masked-out bits.

Data Masking operates in one of two modes. In the "Clear" mode, all masked-out bits of the result word are replaced by a zero when the result is read into the Operand A register; masked-in bits are read in unchanged. If the "Clear" mode is not active, masked-out bits cause a write inhibit on the corresponding bit of the Operand A register. The masked-off bits in the result (Operand A) register are unaffected by the operation, but masked-in bits will be loaded normally. In the Arithmetic Unit, Data Mask logic causes the suppression of carry generate out of masked-out bits, but carries are propagated across masked-out fields.

There are 16 full-word Data Mask Registers. They will normally be loaded with the most commonly used masks for processing target language instructions efficiently. When the GENeral data Transfer (GENT) Ministep specifies a transfer to or from the Data Mask Registers, the MASK field is used as an address to read into or out of a particular Mask Register. When the Data Mask Registers are not used as a transfer source or destination, the MASK field specifies the Data Mask which is effective on the data transfer.

1.1.5 Shift Operations

The Operating Engine has two identical Shifter elements (shown in Figure 1.1 as the Double Length Shifter). The Result Shifter operates on the output of the Parallel Arithmetic Unit. The Extension Shifter works in conjunction with a second register of the General Registers. Two of the Operating Ministeps specify shifts. In the GEAR Ministep, only the output of the Parallel Arithmetic Unit can be shifted. In the SHift INstruction (SHIN), other shift paths can be specified. This includes connected or independent two-word shifts, indirectly specified shift amounts, and register gating for the Extension Shifter. Shift paths and arithmetic operations used for divide and multiply algorithms are controlled by the SHIN Ministep. The following direct shifts can be specified; left or right 0,1,2,3,4,6,8, 16 bits and an 18 bit exchange.

1.1.6 External Busses and Registers

The principal means of communication between the IC-9000 processor and the outside world are four, full-word External Busses and their associated External Buss Registers. The four busses are functionally identical and information transfer is bi-directional. Control lines provided with each buss identify the type of information being transmitted. Main Memory, peripheral equipment, data terminal devices or other processors may mutually share a single buss or be assigned to separate busses, depending on system requirements. Main Memory is accessed in the same manner as other peripherals and is not directly tied to processor logic.

Interfacing with each buss structure is a dual-purpose I/O register which is controlled by both the IC-9000 clock and an External Buss Clock. The external clock is generated by peripheral devices attached to the buss. The use of dual-clocked I/O registers gives the IC-9000 a versatile interface for communicating with peripherals which are asynchronous to the IC-9000 clock or which require multiple clock cycles for access.

1.1.7 Address Modification Unit

The Address Modification Unit (AMU) performs a wide variety of tasks related to the manipulation of input/output data and address words in the External Buss Registers. In conjunction with the Control Engine Language Boards, the AMU provides a means of adapting the IC-9000 to a desired system configuration, emulation mode or target language processing capability. During execution of the Conditional External Data Exchange (CEDE) Ministep, the AMU provides General Register addresses for address modification and controls the format of external address words which are sent to the IC-9000 Main Memory and other peripheral devices.

The Address Modification Unit may optionally be used for memory mapping and storage protection. In conjunction with MINIFLOW routines, the AMU can be used for program storage allocation and segmentation. The decision on which features to incorporate into the Address Modification Unit should be based on user requirements. The size and configuration of main memory, overall system configuration and characteristics of the object program have a direct impact on the design of this portion of the AMU.

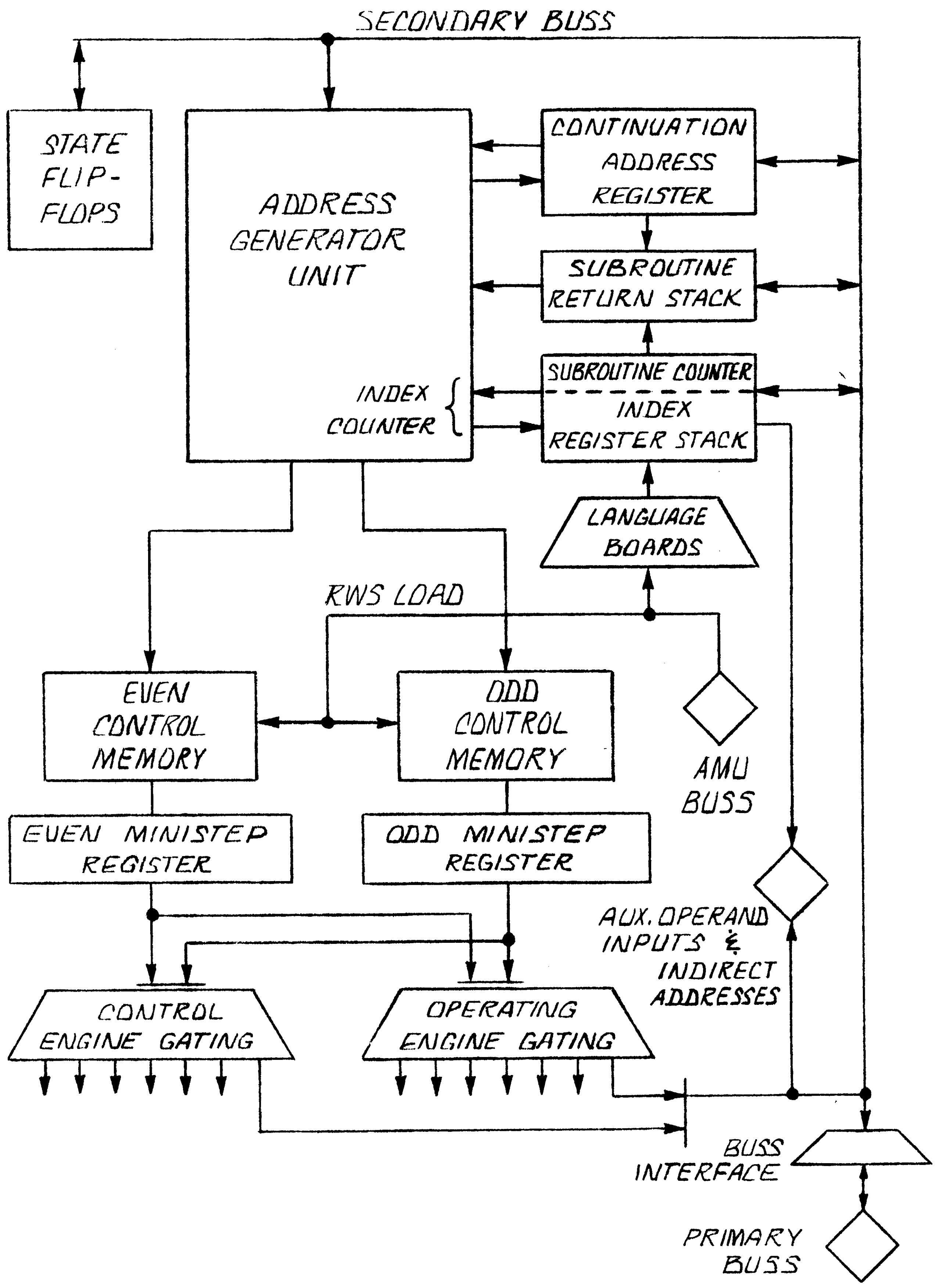

FIGURE 1.2 - CONTROL ENGINE BLOCK DIAGRAM

1.2 Control Engine Design

The Control Engine accesses two MINIFLOW words at a time. These two Ministeps will be executed simultaneously if they fall in the sequence of an Operating Ministep followed by a Control Ministep. The Control Engine has extensive indexing, subroutine and branch test capability and communicates with the Operating Engine, in addition to its control and gating functions.

The Address Generator Unit develops Control Memory address pairs from various sources of address data. Addresses are generated after each clock pulse and access to Control Memory takes place before the next clock. Control Memory is organized into odd and even banks of locations so that two Ministeps can be read out simultaneously. Control Memory locations may be implemented in Read-Write Storage or Read-Only Storage,or both, at the users option. MINIFLOW routines in Read-Write Storage can be altered or exchanged under program control. MINIFLOW executive routines and commonly used subroutines which are never changed will normally reside in Read-Only Storage. Read-Only Storage is available at a lower cost than an equivalent amount of Read-Write Storage.

Control Ministep sequencing functions include branching, subroutine entry and return and indexed addressing. It is feasible to transfer out of a program sequence at almost every clock cycle. There are no restrictions on ordering different types of Ministeps sequentially, so it is not necessary to use the simultaneous execution feature where no advantage is gained. This approach results in saving some storage locations in the Control Memory and the Ministep word length was kept to 32 bits.

Conditional tests for branching, indexing, etc., are accomplished within a single clock cycle with some exceptions. Arithmetic Unit status outputs such as Zero Result, Carry, etc., are not available sufficiently early in the clock cycle to guarantee an absolute branch test. In this event, a tentative address is generated and and the Control Memory is accessed as if the test condition were true. If the test condition is false at clock time, the tentative access is discarded and an extra clock cycle is taken to access the correct Ministep pair. Branch and index Ministeps can test for either the false or the true condition of State Flip-flops. Some branch Ministeps can test for a specific logical combination of two State Flip-flops.

1.2.1 Address Generator Unit and Continuation Address Register

Addresses to the Control Memory are generated by a group of adders and incrementers, designated collectively as the Address Generator Unit (AGU). Another function of the AGU is to act as an index counter for incrementing the Index Registers. The Continuation Address Register is loaded with the next sequential continuation address in preparation for the next instruction execution cycle. When a subroutine entry occurs, the continuation address is also loaded into the Subroutine Return Stack.

In the process of generating addresses to the Control Memory, several alternate sources of input data are available. The selection of the inputs is dependent on the type of Control Ministep which is being executed and the logical condition of one or two State Flip-flops under test. Address input data possibilities are:

- o Contents of the Continuation Address Register
- o Branch Address data from the Control Ministep Register
- o Index Register contents (one of 16)
- o Contents of the active Subroutine Return Register
- o Control Memory access inhibit signal
- o Miscellaneous forcing inputs for power failure, start-up, etc.

Several modes of sequencing MINIFLOW routines are synthesized from these inputs. Departing from the nominally sequential, i.e., Continuation Address and Continuation plus one, we might have (neglecting the incremented address for the second Ministep) branch relative, branch absolute indexed, branch and subroutine entry, branch and increment index, forced branch, continuation plus index, subroutine return and "wait". The "wait" mode can be caused by Data Exchange Ministeps which inhibit further Control Memory accesses until a particular test input goes to the true state.

1.2.2 Control Memory Design

In addition to being organized into odd and even locations for accessing any two sequential Ministeps each clock period, the Control Memory can also be implemented using two distinct types of memory elements — read-only or read-write.

The Read-Only Store (ROS) is a high-speed, 32-bit-word memory with contents which are permanently fixed when the module is built. Data in the ROS cannot be altered by overwriting, power loss, or transient conditions. A significant restriction on the use of ROS modules is that the contents must be completely specified at the time hardware is ordered. It is possible, however, to alter the amount of ROS in Control Memory in increments of 2,048 words.

Read-Write Store (RWS) modules are functionally similar to ROS, except that they can be loaded from an external source. Read-Write storage is considerably more expensive than Read-only storage on a per-location basis. Read-Write storage might be used for overlaying compiler subroutines, highly specialized macros or rarely used Ministep sequences where it is not cost-effective to fabricate and check out the routine in Read-Only Store. RWS is also useful for initial program checkout and debugging prior to specification of ROS memory modules. Because integrated circuit registers are used in the RWS, only 1,024, words can be accommodated on a module. Thus, RWS storage can be specified by the user in increments one-half the size of ROS modules. The ration of fixed, (read-only) storage to alterable (read-write) storage may be varied to suit the processing application.

1.2.3 Ministep Registers and Gating Elements

The two, 32-bit, Even Ministep and Odd Ministep Registers are central to Control Engine operations. Ministeps in these registers are decoded by two gating structures, one for the Operating Engine and one for the Control Engine. Outputs from the gates control all programmable IC-9000 data transfers, processing, testing and sequencing.

After each access of the Control Memory these two registers hold the initially addressed MINIFLOW word and its next sequential successor Ministep. The current Ministep is extracted from the Control Memory location with the numerically smaller address of the two Ministeps. It may be located at either an odd or an even address. The current Ministep is routed to the Operating or Control Engine Gating structure, as specified by the most significant bit in its OP CODE field. If the current MINIFLOW word is an Operating type Ministep and its successor (current Ministep address plus one) is a Control type Ministep, both will be executed simultaneously.

It is possible to program two or more Control Ministeps sequentially and end up with a Control Ministep as the current Ministep. When this happens, the current Ministep has priority of execution over the successor Ministep. At this time the current Ministep word is routed to the Control Engine gating elements and the successor Ministep is not executed. In one instance, the word stored in the successor location will be neither an Operating nor a Control Ministep. This situation occurs when a Long Immediate Operand (literal) is specified by an Operating instruction as an input to the Arithmetic Unit.

1.2.4 Subroutine Return Stack and Counter

This group of 16 storage registers is loaded with a subroutine return address each time a subroutine entry is executed. Under the control of the Subroutine Counter, these registers function as a push-down stack. Destacking occurs automatically when a subroutine return operation is executed by the program.

Although the Subroutine Counter is also one of the Index Registers, it is covered here for continuity. The Subroutine Counter performs two functions in controlling the loading of the Subroutine Return Stack. First, it acts as a pointer to specify the subroutine return address currently active and the location in which the next return address is to be stored in the event a subroutine entry is executed. Secondly, it indicates, by setting a State Flip-Flop, that the stack is full and requires further servicing before another level of subroutine nesting can be accommodated. It is an up/down counter, incrementing at each subroutine entry and decrementing when a return is executed.

1.2.5 Index Registers

The IC-9000 has an assigned group of 16 registers to perform various functions including indexing. Some members of this group are not true index registers in the commonly accepted sense of the term. Others are general registers which can be used to perform various addressing, control or indexing functions, depending on the program context. Special purpose registers include the Subroutine Counter (discussed in Section 1.2.4) and the Indirect Shift Control Register. Other Index Register locations will be assigned special functions if needed to improve processing capability. An important function of the Index Registers is to receive data which is "unpacked" and translated from target language instructions by way of the IC-9000 Language Boards.

Each of the General and special purpose index registers has at least one associated pseudo-flip-flop which is treated for decision making purposes as a State Flip-flop. For the general Index Registers, the output of the associated pseudo-flip-flop is zero if the index register has a count other than zero. The output becomes one when the register count passes through zero.

1.2.6 Language Boards

The IC-9000 Language Boards translate target language order codes into addresses which specify MINIFLOW subroutine entry points and unpack and translate other fields in target instructions into pre-assigned Index Registers and State Flip-flops. The functions of the Language Boards include:

- o Generate subroutine entry locations for accessing Control Memory.
- o Set State Flip-flops to indicate status and provide control functions during subroutine execution.
- o Collate inputs to the Index Registers for indirect addressing, shift control, indexing, etc.

Inputs to the Language Boards are routed from the External Buss Registers via the AMU Buss.

1.2.7 State Flip-flops

The IC-9000 has a group of mode control and mode indicating elements, called State Flip-flops. State flip-flops (and pseudo-flip-flops) respond to a variety of logical conditions and status inputs, which reflect the IC-9000 processor internal condition and monitor its external environment. Some are assigned to signal interrupt conditions such as I/O requests, power failure, etc. Other State Flip-flops respond to parity errors, carry overflow, register zero, and similar conditions in the Operating Engine which may require intervention and servicing by the program. For addressing purposes, the State Flip-flop array may contain up to 256 elements.

In addition to such obvious functions, all mode control elements which define the current status of the processor are included in the array of State Flip-flops. Various Control Ministeps are available; to test and manipulate the State Flip-flops, to load data into those which are not pseudo-flip-flops or privileged elements and to read out their current status into other registers in the Control and Operating Engine.

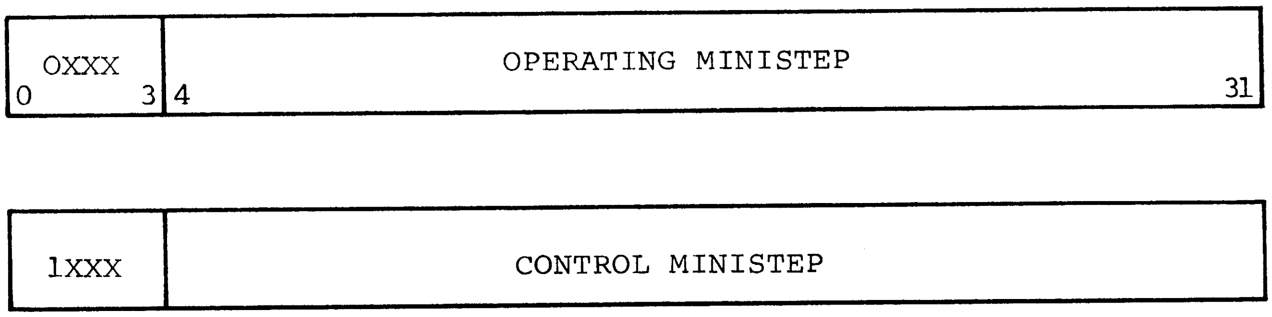

Figure 2.0 Basic Ministep Formats

2.0 Ministep Formats and Conventions

IC-9000 Ministeps are 32 bits long and are classed as Operating or Control Ministeps. One exception occurs where the 32-bit word immediately following a current Operating Ministep is treated as data instead of an instruction. This occurs when Long Immediate data is called as an input to the Arithmetic Unit. A 36-bit data word is synthesized using the LONG IMMEDIATE EXTEND field in the Operating Ministep to supply the other 4 bits of the 36-bit operand.

Bits are numbered 0 through 31, left to right. The leftmost bit in any field, sub-field or word is the most significant bit (MSB) for the purpose of addressing, decoding, use as an operand, etc. The least significant bit (LSB) is to the right. Ministep names are generally acronyms made up from initial letters of words describing their function. All acronyms are four letters long and form a real English word. The single exception to the use of acronyms is the MOVE Ministep, which transfers (moves) data between Control Engine locations via the Secondary Buss. The names of all Ministeps and field titles within a Ministep are capitalized.

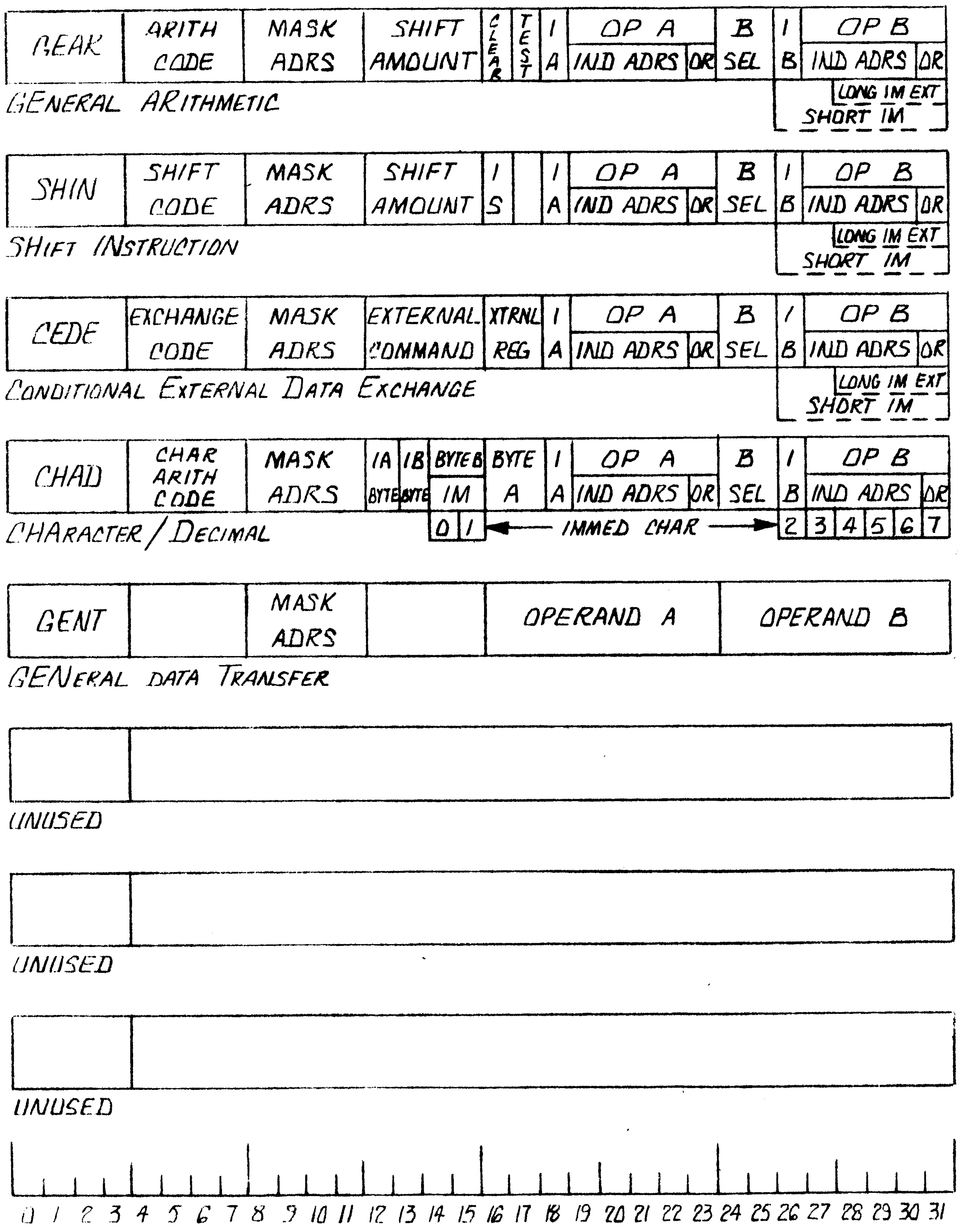

FIGURE 2.1 - OPERATING MINISTEPS

2.1 Operating Instructions

2.1.1 GEAR - GEneral ARithmetic

The GEAR Ministep provides a generalized instruction which specifies arithmetic and logic operations to be performed by the Parallel Arithmetic Unit upon operands from the General Registers and other sources. Table 2.1.1 lists these arithmetic and logical operations. If the TEST bit is one, the result goes to the Primary Buss, but the Operand A Register is unchanged. Outputs from the Arithmetic Unit set State Flip-flops for Control Ministep testing.

Bits shifted out of the result are lost and bits shifted in from either the low order or the high order end of the word are zeros or ones as controlled by a dedicated State Flip-flop. Allowable shift amounts are 0, 18 bit exchange, and left or right 1,2,3,4, 6,8 and 16 bits.

Masking is always enabled. Mask Registers (M0-M15), previously loaded by the GENT Ministep, function in two modes. Masked out bits of the result are either cleared or unchanged as determined by the CLEAR bit. When a masked out bit is shifted into a masked in area, the masked out, shifted bit will be zero in the result.

OPERAND A - If the INDIRECT A OPERAND bit is zero, bits 27-31 address the General Registers (G0-G31). If the A bit is one, the INDIRECT ADDRESS field specifies one of the Control Engine Index Registers (I0-I16). The five low order bits of the Index Register contents, including a logical OR of the low order bit from the Index Register and the low order bit from the OPERAND A field, are combined to obtain the General Register address desired.

OPERAND B - The B SELECT field specifies one of four OPERAND B inputs:

- o General Registers
- o Index Registers
- o Short Immediate Data
- o Long Immediate Data

If the General Register input is selected, I A bit and the remaining five address bits have the same format as the OPERAND A field. If an Index Register address is selected, bits 27-30 address the Index to be used as a data input.

The Short Immediate specification causes the low order six bits of the OPERAND B field to be combined with thirty, high order zeros of the B operand literal input. The Long Immediate code gates the successor Ministep word (32 bits) as the low order B operand bits. The most significant four bits of the literal operand come from the low order four bits of the OPERAND B field.

Table 2.1.1 Arithmetic Operations

Operation	A is replaced by:
$A \leftarrow B$	B
$A \leftarrow \overline{B}$	the one's complement of B
$A \leftarrow A U B$	A logically OR'ed with B
$A \leftarrow A U \overline{B}$	A logically OR'ed with the complement of B
$A \leftarrow \overline{A} U B$	the complement of A logically OR'ed with B
$A \leftarrow A \cdot B$	A logically AND'ed with B
$A \leftarrow A \cdot \overline{B}$	A logically AND'ed with the complement of B
$A \leftarrow \overline{A} \cdot B$	the complement of A logically AND'ed with B
$A \leftarrow A E B$	the exclusive OR of A and B
$A \leftarrow A E \overline{B}$	the exclusive NOR of A and B
$A \leftarrow A+B$	the sum of A and B
$A \leftarrow A+\overline{B}+1$	A plus the 2's complement of B
$A \leftarrow \overline{A}+B+1$	B plus the 2's complement of A
$A \leftarrow A+\overline{B}+C_I$	A plus the complement of B with a conditional carry-in
$A \leftarrow \overline{A}+B+C_I$	B plus the complement of A with a conditional carry-in
$A \leftarrow A+B+C_I$	A plus B with a conditional carry-in

Notes: "A" refers to OPERAND A, "B" refers to OPERAND B. "$\overline{A}$" refers to the one's complement of A. "+" means add; if there is a carry out, a State Flip-flop will be set. "E" means Exclusive OR. "U" means OR. "$\cdot$" means AND. "C_I" designates a conditional carry-in, as controlled by a State Flip-flop.

2.1.2 SHIN - SHift INstruction

SHIN controls single register shifts, even/odd register pair shifts, and a dedicated Shift Extension Register (G31) which can be combined with any General Register to provide a shift register pair. A partial list of SHIFT CODE assignments is given in Table 2.1.2. For even/odd register pair shifts, the A OPERAND field must address the even register. The odd address is implied. For SHIFT CODES which specify the Shift Extension Register, the A OPERAND addresses any General Register. The B OPERAND field is used only with special shift codes, such as Multiply and Divide, where auxiliary functions are needed.

Masking affects only the specifically addressed register (OPERAND A), never the implied register. Masked out bits in the OPERAND A register are unchanged, since there is no CLEAR option. The other functions of masking are the same as for GEAR. If the INDIRECT SHIFT bit is zero, the SHIFT AMOUNT field indicates shift direction, while the amount is specified by the Indirect Shift Control Register. Shift amounts which are a power of two are shifted in one clock cycle. The bit specifying the highest power of two takes precedence. A shift of 31 would occur in this order; 16,8, 4,2,1, taking five clock cycles (using a two Ministep loop). After each clock cycle the Indirect Shift Control Register contains the shift amount remaining (residue).

Table 2.1.2 Shift Operations

Function	Type
A	Single Shift
A (even) / A (odd)	Disconnected Long Shift
A (even) $\rightarrow$ A (odd)	Connected Long Shift
A (odd) $\rightarrow$ A (even)	Connected Long Shift
A (even) $\rightarrow$ A (odd) $\rightarrow$ A (even)	Long Circle Shift
A $\rightarrow$ Shift Extension	Connected Long Shift
Shift Extension $\rightarrow$ A	Connected Long Shift
A $\rightarrow$ Shift Extension $\rightarrow$ A	Long Circle Shift

"A" refers to OPERAND A. "A (even)" and "A (odd)" are a register pair where the address of A (odd) is A (even)+1. A right arrow ($\rightarrow$) denotes that shifted out bits go to the register to the right of the arrow. A slash (/) specifies that shifted out bits are lost.

2.1.3 CEDE - Conditional External Data Exchange

This instruction permits the exchange of data and address/control codes between I/O devices, Main Memory and the processor. Various EXCHANGE CODES are provided (see Table 2.1.3 for partial listing) to generate and format the address of desired data, receive the requested data, format the address of a data receiver, and transmit the data word to the receiver. Information coming in can be interpreted in one mode as a target language instruction. Unpacking of instructions is performed by the Language Boards which are tailored to the target language being used. Exchange codes are provided to accomplish the routing of instructions to the Language Boards.

Table 2.1.3 Exchange Codes

Mnemonic	Operation	Function
FIN*	$(A+B)_{ADR} \rightarrow ER$	Fetch next INstruction
WIN*	$(ER) \rightarrow I$, CM $(ER)_{ADR}+B \rightarrow A$, $(ER)_{ADR}+B \rightarrow ER^{**}$	Wait for next INstruction, load and start execution, perform address modification and save and request operand if needed.
FOP	$(A+B)_{ADR} \rightarrow ER$	Fetch an OPerand
WOP	$(ER) \rightarrow A$	Wait for OPerand and load
WOF	$(ER) \rightarrow A$, $(B)_{ADR} \rightarrow ER$	Wait for Operand, load and Fetch next operand
SOP	$(A+B)_{ADR} \rightarrow ER$	Store OPerand (request a write cycle)
WAS	$(B) \rightarrow ER$	Wait for Address acknowledge and Store operand
GAD	$(ER)_{ADR}+B \rightarrow A$	Generate ADdress from external register and save
RMW	$(A+B)_{ADR} \rightarrow ER$	Request a Read-Modify-Write cycle
WOM	$A \bullet M \; U (ER) \bullet \overline{M} \rightarrow ER$	Wait for Operand, Modify and store
LCM	$(ER) \rightarrow CM$ $(A+B)_{ADR} \rightarrow ER$	Wait for operand, Load in Control Memory and request next operand. Exit when (B)=0

* Inhibited if an interrupt is pending

** Inhibited if instruction is not an operand fetch type

Notes: A refers to word designated by OPERAND A address, B refers to word designated by OPERAND B address and ER refers to the EXTERNAL Register designated by the XTRNL REG field. I refers to Control Engine Index Registers and CM refers to Control Memory. M refers to Mask.

2.1.4 CHAD - CHAracter and Decimal Operations

This Ministep operates on the 32 low order bits of an Operating Engine word. Four 8-bit bytes are addressable in both operands. The 8-bit-wide Decimal Arithmetic Unit performs arithmetic and logical operations as defined by the CHARACTER ARITHMETIC CODE field. Some operations performed by this Ministep are Add and Subtract (both for an 8-bit binary character and for two, 4-bit, binary-coded-decimal digits), 8-bit logic operations; and an 8-bit move. Used in conjunction with the processor masking ability, this instruction gives great flexibility in handling odd sized fields and skewed bytes. Bytes are numbered 0-3 and may be addressed directly or indirectly as specified by the INDIRECT A and INDIRECT B BYTE fields. If the B SELECT field indicates that the B OPERAND is to be immediate, the next 6 bits are concatenated with the 2-bit BYTE B field to form an 8-bit immediate input.

2.1.5 GENT - GENeral data Transfer

The GENT Ministep allows complete flexibility in data movement in the Operating Engine. OPERAND B replaces OPERAND A. Masking is enabled unless the Data Mask Stack is specified as a data source or destination. In this event, the Mask Register addressed in the MASK ADDRESS field is either loaded or read out to the Buss.

2.2 Control Instructions

2.2.1 BRAT - BRAnch Test

If the tested condition is true, the Continuation Address (the address of the next sequential instruction) is added to the RELATIVE ADDRESS to obtain the resulting Branch Address. The addition is 2's complement, allowing branching both forward and backward. TEST BIT A and TEST BIT B are addresses of State Flip-flops. The TEST MODE field determines how these flip-flops are to be tested.

There are four main TEST MODES.

- o Test A and move A to B
- o Test AUB (OR)
- o Test A·B (AND)
- o Test AEB (EXCLUSIVE OR)

In addition, the complement of A and/or B may be specified.

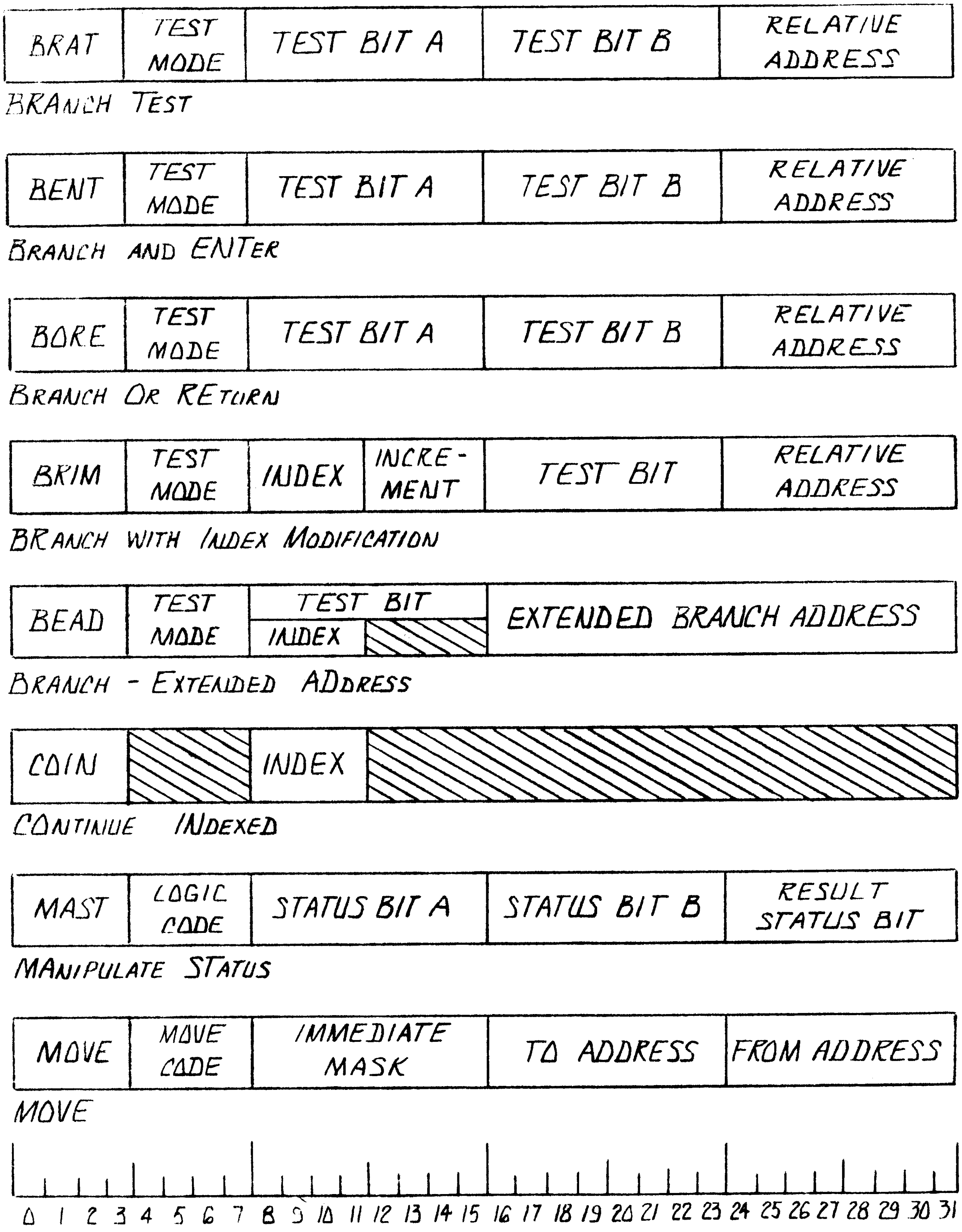

Figure 2.2 - CONTROL MINISTERS

2.2.2 BENT - Branch and ENTer

If the test is satisfied, the branch is taken and a subroutine entry is effected by storing the Continuation Address into the next higher address of the Subroutine Return Register (pointed to by the Subroutine Counter) and incrementing the Subroutine Counter by one. If the test fails, no branch occurs and the continuation Ministep is executed. TEST MODE and RELATIVE ADDRESS are specified in the same manner as BRAT.

2.2.3 BORE - Branch Or Return

If the test is satisfied the branch is taken. Otherwise, a subroutine return is effected by obtaining the next Ministep address from the active Subroutine Return Register (pointed to by the Subroutine Counter) and decrementing the Subroutine Counter by one. The TEST MODE and RELATIVE ADDRESS are specified in the same manner as BRAT.

2.2.4 BRIM - BRanch with Index Modification

TEST MODE determines whether A or $\overline{A}$ is to be tested. If the test is true the branch is taken, and the address INDEX Register is incremented the amount specified in the INCREMENT field. The addition is 2's complement, giving a span of plus 7 to minus 8. If the State Flip-flop which indicates the zero condition of the addressed INDEX is tested, this Ministep functions as a loop control.

2.2.5 BEAD - Branch Extended ADdress

The TEST MODE field determines whether a State Flip-flop, or its complement, is to be tested (resulting in a conditional branch), or if an unconditional, INDEXed branch is to be executed. The EXTENDED BRANCH ADDRESS is not relative and contains 16 bits. The contents of the selected Index Register are added, as an 8-bit positive number, to the least significant eight bits of the EXTENDED BRANCH ADDRESS.

2.2.6 COIN - COntinue INdexed

The continuation Ministep address is added to the indicated Index Register to obtain the address of the next Ministep, giving the effect of a multiple-address branch, or "tabling" function.

2.2.7 MAST - MAnipulate STatus

STATUS BIT A is logically combined with STATUS BIT B and the result is placed in the RESULT STATUS BIT. The LOGIC CODE allows the RESULT STATUS BIT to be replaced by ABU, A·B, AEB (OR, AND, Exclusive OR). Also the LOGIC CODE allows the complement of A and/or B to be selected as inputs.

2.2.8 MOVE - Control Engine MOVE

Data in the Control Engine register addressed by the FROM ADDRESS is moved, as determined by the MOVE CODE, to the TO ADDRESS. An IMMEDIATE MASK provides for the transfer of selected fields of Control Engine registers. Mask bits of one allow the field to be moved as specified by the MASK CODE. Mask bits of zero leave this field unchanged in the TO ADDRESS register.

Standard

Standard Computer Corporation
1411 West Olympic Boulevard
Los Angeles, California 90015
(213) 387-5267

777 North 1st Street
San Jose, California 95112
(408) 294-7150

7100 France Avenue South
Minneapolis, Minnesota 55435
(612) 926-0706

2 West 45th Street
New York, New York 10036
(212) 661-1834