IC-9000 CENTRAL PROCESSING UNIT

PRINCIPLES OF OPERATION

PRELIMINARY

FORM 809001-4

TABLE OF CONTENTS

TABLE OF CONTENTS (continued)

LIST OF ILLUSTRATIONS

LIST OF TABLES

PREFACE

This document is an introduction to the functional design of the IC-9000 Central Processing Unit. It includes a description of system architecture, processing facilities and IC-9000 order codes.

The IC-9000 is a microprogrammed processor with an on-line alterable Control Memory. It possesses a designed-in capability for efficient interpretive execution of instructions in formats and languages other than its own internal order code set. On-line alterable microprogramming is not the only capability needed to make a processor efficient in multi-lingual interpretive applications. Three design principles were followed to develop a processor capable of handling a broad spectrum of target languages:

- The system architecture is generalized and provides direct control of logic functions and access to all storage elements.

- Order codes are well-structured and are similar in function to typical machine language instruction sets.

- The processor uses language-dependent hardware with complementary microprograms to efficiently adapt to widely varying target instruction and operand formats, I/O structures and memory addressing modes.

The language-dependent, plug-in hardware is used to perform a variety of translation and formatting functions more efficiently than unassisted software. Hardware aids can provide:

- High speed in executing repetitive functions; such as target instruction decomposition, order code translation and microprogram execution routine entry.

- Formatting of memory addresses and translation functions, such as page or byte addressing, memory protection, relocation, etc.

- Reduction in Control Memory storage requirements.

STANDARD COMPUTER CORPORATION Representatives are available to aid you in establishing requirements, specifications and performance goals for complementary firmware and hardware application packages for the IC-9000.

GLOSSARY OF COMMONLY USED TERMS

TARGET LEVEL TERM	COMPARABLE MICROPROGRAM LEVEL TERM
Software	Firmware (Microprogram)
Instruction Set (Order Code)	MINIFLOW Language
Instruction	MINISTEP
Main Memory	Control (Microprogram) Memory
Program Execution	Emulation or Interpretive Execution
Interrupt	Action Request
Program Status Word	Miniflow Status Word
Instruction (Program) Counter	Current Address Register
Index Registers	Pointer Registers
Mode Control (Condition Code) Elements	State Flip-Flops
Memory Fetch/Store Functions	Data Exchange Operations
I/O Operations	Transfer External Ministep
Memory Protection and Relocation	Operating Engine Language Board (Address Modification)
High Speed Buffer	Auxiliary Registers

SECTION 1
IC-9000 PROCESSOR DESIGN

1.1 GENERAL

The STANDARD IC-9000 is a microprogrammed data processor designed to provide a high degree of versatility and computing power. One of the most important features is the high-speed, on-line alterable microprogram memory. The order code is designed for ease of use. The Processor language is called MINIFLOW*, individual instructions are Ministeps. MINIFLOW programming resembles certain aspects of machine language programming on earlier processors. The IC-9000 is designed for direct access at the microprogram level. The MINIFLOW programmer can "get at" all mode control flip-flops and machine registers. MINIFLOW routines and complementary hardware tailor the IC-9000 for efficient execution of a broad spectrum of target languages and systems. High level procedures in existing languages can be executed in MINIFLOW directly. Complex macros and subroutines can be synthesized in MINIFLOW and execution can be initiated by the fetch of a single, problem-oriented, target language instruction.

In a typical processor, the instruction set generally can be separated by function into three categories:

a. Manipulation of arithmetic and logical operands.

b. Control and sequencing of instruction execution.

c. Input/Output operations (which often combine both arithmetic and control functions).

In the STANDARD IC-9000 Processor, the facilities for performing arithmetic and logical transformations on data have been separated from the control and sequencing functions to a large degree. Control signal and data transfer interfaces provide communication between the data handling section, or Operating Engine, and the sequencer, or Control Engine. The Control Engine contains the microprogram (control) memory. Input/Output operations, as in other processors, require considerable interaction between the sequencing and the data handling elements of the IC-9000.

MINIFLOW instruction words are similarly divided into two types, called Operating and Control Ministeps, which sequence the elements of the Operating and Control Engines, respectively. If an Operating Ministep is followed in sequence by a Control Ministep, the two will be executed simultaneously. Otherwise, a single Control Ministep or one Operating Ministep is executed.

Some processor tasks, which are highly repetitive in nature, are relatively inefficient when executed by an unassisted firmware routine. The IC-9000 has provision for up to four sets of "Language Boards". The principal functions of Language Boards

**MINIFLOW is a trademark of STANDARD Computer Corporation.*

are to generate addresses and sequence accesses to Main Memory, extract data from target language instructions, and generate MINIFLOW entry addresses (branch vectors).

MINIFLOW language design and the IC-9000 hardware provide the following capabilities:

- On-line alterable microprogram (Read/Write Control Memory).
- Expandable Control Memory.
- Interpretive execution of a wide range of target languages.
- Multiple general-purpose registers.
- Up to 1024 high-speed data registers (optional).
- BCD and byte arithmetic.
- Multiple, independent, asynchronous, Input/Output busses.
- Variable data field manipulation.
- Direct access to all registers and control flip-flops.
- Indirect register addressing at the microprogram level.
- Flexible microprogram branching and transfer of control.
- Microprogram subroutine entry and return.
- Order code organized for efficient use of Control Memory.

Figure 1-1 is a block diagram of the overall IC-9000 Processor design.

1.2 IC-9000 SUMMARY SPECIFICATIONS

- 128 nanosecond clock cycle
- 32 General Registers; 36 bits plus 4 parity
- Up to 1024 Auxiliary Registers*; 36 bits plus 4 parity
- Expandable Control Memory up to 64K (K = 1024) locations; 32 bits plus 1 parity
- Both Read-Write and Read-Only Control Memory available in 512-word modular increments
- 32 Data Mask Registers; 36 bits plus 4 parity

*Optional in increments of 256 words.

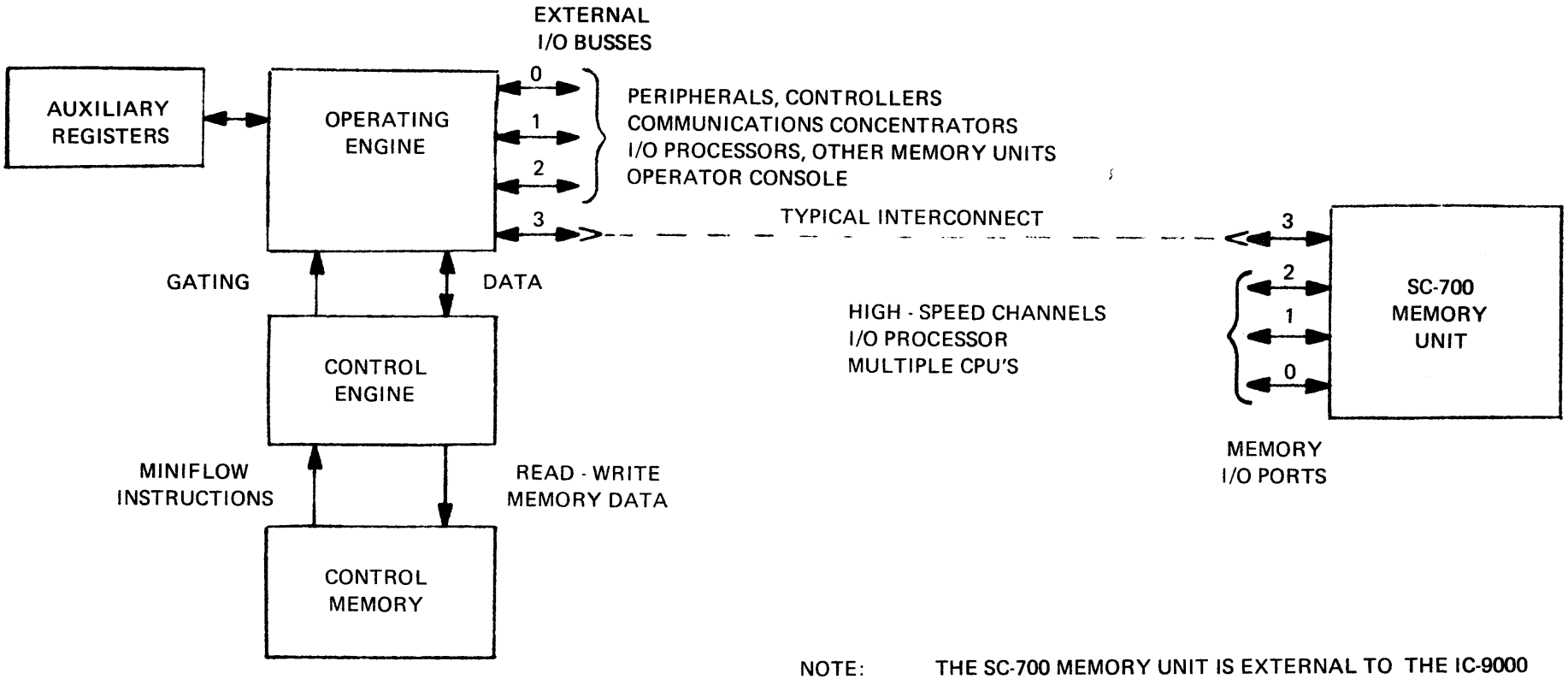


Figure 1-1. IC-9000 Overall Block Diagram

NOTE: THE SC-700 MEMORY UNIT IS EXTERNAL TO THE IC-9000 PROCESSOR. ANY IC-9000 BUSS CAN BE CONNECTED TO ANY SC-700 PORT. UP TO EIGHT, 32K-WORD, SC-700 MEMORY MODULES FIT INTO A SINGLE SC-700 CABINET. UP TO SIXTEEN CABINETS CAN BE ACCOMMODATED ON ONE BUSS.

- 4 independent, asynchronous, half-duplex, External (Input/Output) Busses; 36 bits plus 4 parity
- Automatic microprogram subroutine stacking to 15 levels
- Up to 4 target languages (independent order code sets) per processor

1.3 OPERATING ENGINE DESIGN

Figure 1-2 is the functional block diagram of the Operating Engine. This portion of the IC-9000 Processor contains registers, shifters and arithmetic and logical elements used to manipulate operands. The Operating Engine External Busses interface with Main Memory, peripherals and other external devices. Several interfaces provide communication with the Control Engine. The Operating Engine can execute six Operating Ministep types:

a. General Arithmetic (GEAR)--Performs binary arithmetic and logical operations.

b. Character/Decimal (CHAD)--Byte and BCD arithmetic and logical operations.

c. Conditional External Data Exchange (CEDE)--Transfers addresses, target instructions and data between the IC-9000 and Main Memory.

d. Transfer External (TEXT)--Transfers addresses and data between the IC-9000 and devices on the External Busses.

e. Shift Instruction (SHIN)--Executes complex shift operations.

f. General Data Transfer (GENT)--Transfers data between Operating Engine Registers and to and from the Control Engine interface.

The nominal Operating Engine register length is 36 bits. The Primary Adder operates on two, 36-bit inputs and the Byte/Decimal Adder manipulates two, 8-bit bytes. Parity is maintained on 9-bit bytes during data transfers in the Operating Engine. Logical transformations which do not maintain parity, such as arithmetic and shifting operations, are performed by two identical, independent, logical structures. The outputs of the two elements are compared at clock time for identity. If they are identical, parity is regenerated; otherwise, an Action Request (microprogram interrupt) is forced. Byte parity (9-bit) is sent out over the External Busses and incoming parity is checked when data is moved from the External Buss Registers.

The Operating Engine accommodates up to four Language Boards. These provide a hardware assist to data exchange operations (CEDE Ministeps) which process target language instructions and format Main Memory addresses. Each Language Board in the Operating Engine is matched with a corresponding Control Engine (CE) Language Board. Operating Engine (OE) Language Boards format addresses and commands to Main Memory. They also gate indirect addresses to General Registers, control the Primary Adder and transfer data to External Buss Registers. These functions allow

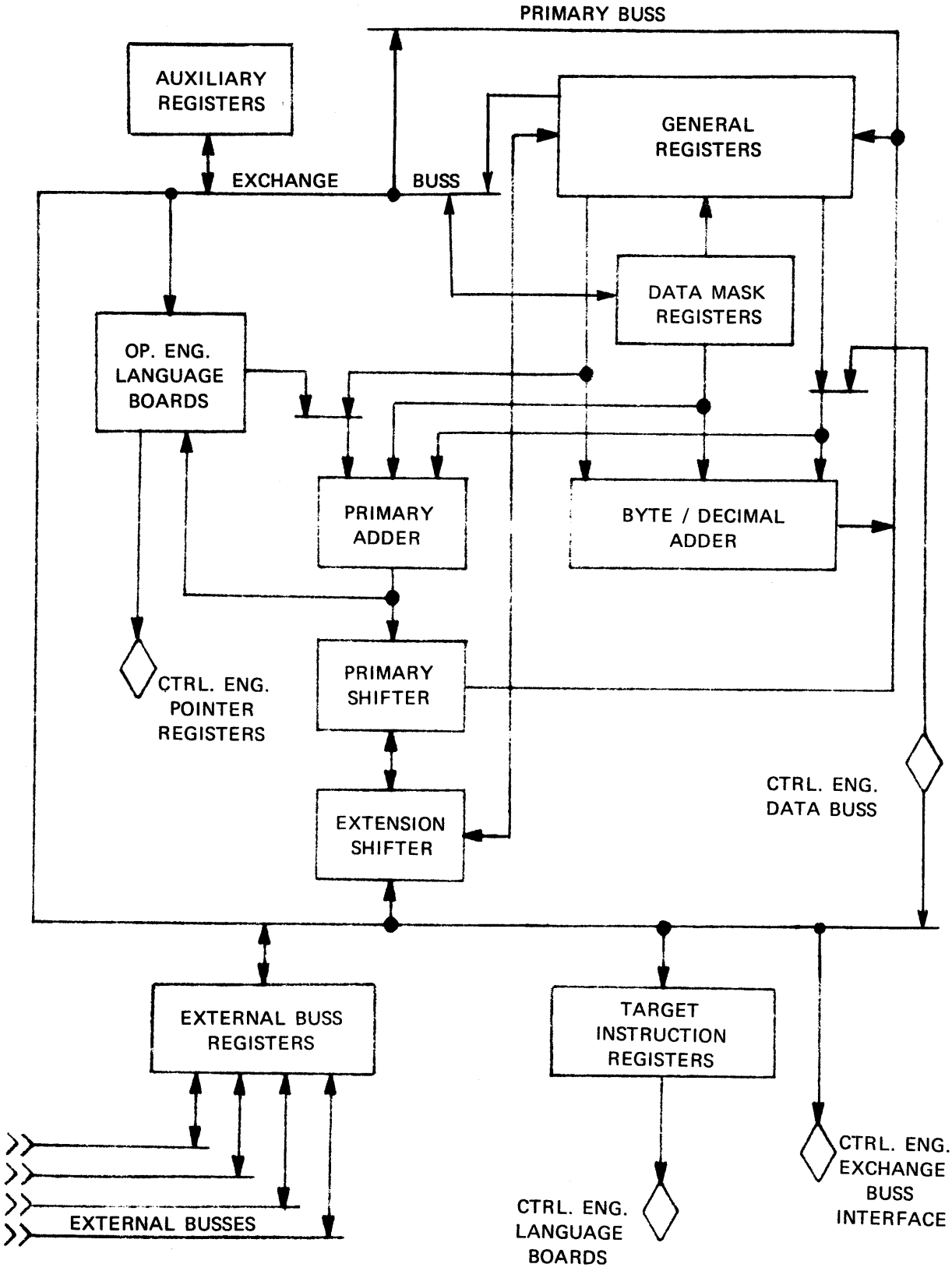


Figure 1-2. Operation Engine Block Diagram

certain types of fetches from memory to be addressed by data in an incoming word. The third major function of the OE Language Boards is to translate target language order codes into initial MINIFLOW entry addresses. Memory protection, certain types of indexing, page searching and other special-purpose address modification functions can be performed by OE Language Boards. Language Boards translate order codes and extract data and address fields faster than unaided MINIFLOW. Any function provided by either the OE or CE Language Boards can be accomplished by a MINIFLOW routine at the cost of processing speed and increased Control Memory space.

The IC-9000 Processor accommodates up to 1024 full-word Auxiliary Registers. These registers may be used as high-speed (one-clock-time access) buffers, temporary data storage, etc. Auxiliary Registers are optionally available in increments of 256 words.

A "data masking" capability allows manipulation of fields shorter than the full Operating Engine register length. Thirty-two Data Mask Registers, 36 bits wide, are preloaded with mask words via MINIFLOW initialization routines. Mask words, or Masks, modify the operation of both the Primary and Byte/Decimal Adders and data transfers into the result register from the Primary Buss. One data masking mode allows loading a field into a register without changing masked-out bits. In the General Arithmetic (GEAR) Ministep CLEAR mode, masked-out bits are zero-set in the result register. Only masked-in data is transferred in either mode. Arithmetic and logical operations apply to masked-in fields and ignore masked-out bits.

A number of register addresses other than General Registers are addressable during CEDE, TEXT and GENT execution. 1024 addresses are reserved for communicating with the OE Language Board Registers. Other addresses are reserved for the Target Instruction Registers (Primary and Secondary), the CE Data Buss, Data Mask Registers, Auxiliary Registers and External Registers.

Ministeps are obtained two at a time from Control Memory each clock cycle. Operating Ministeps are executed out of the OE Ministep Register (refer to Figure 1-4) in the Control Engine. Decoded control signals, from the OE Ministep Register, sequence the logical elements of the Operating Engine.

1.3.1 General Registers

The IC-9000 has 32 General Registers, each 36 data bits wide. Four parity bits, one for each 9-bit byte, are maintained with each register. All 32 registers are addressable as inputs to the Primary and Byte/Decimal Adders. Except for General Register 31, the Shift Extension Register, none of the General Registers has a dedicated function. General Registers have two independent address structures. The Operand A register specified by an Operating Ministep is gated to one input of the Adders and the Operand B input goes to the other, depending on the addressing options of the Ministep. Any of the General Registers may be specified as either, or both, Operand A or B inputs to the Adders by most operating Ministeps. Both Operand A and B registers can also be addressed indirectly. The indirect address mode of an Operating Ministep uses the

five low-order bits in one of the 16 CE Pointer Registers (refer to paragraph 1.4.5) to specify the address of the corresponding operand in the General Registers.

Transfers of data to the General Registers from the Primary Buss (GEAR and CHAD Ministeps) are modified by the contents of one of 32 Data Mask Registers. In the "Non-clear" or replacement mode of data masking operations, logic prevents "masked-out" bits (corresponding to zeros in the mask word) from being changed in the General Registers. In the "Clear" mode of the GEAR Ministep, masked-out bits are replaced with zeros. When the "Test" function of the GEAR and CHAD Ministeps is enabled, the Operand A General Register is not loaded with the result of the operation, regardless of masking functions.

Odd-numbered General Registers are also bussed to the Extension Shifter. During double-register-length shift operations (SHIN Ministep), the Extension Shifter is paired with the Primary Shifter. Even-odd pairs of General Registers can be specified. General Register 31, the Shift Extension Register, can be paired with any General Register as an operand for double-length shifts. Refer to paragraph 2.2.3 for a description of the Shift Instruction, shift paths and functions.

1.3.2 Primary Adder

The Primary Adder is a 36-bit, parallel binary, arithmetic and logical processor. GEAR Ministeps execute the 16 operations listed in Table 1-1. The result of operating on the Operand A and B inputs is placed in the Operand A register, except when the TEST bit true. The Primary Adder is also used with several CEDE, TEXT and SHIN Ministep types, but operations are limited to binary addition and subtraction.

Operand B inputs to the Primary Adder include General Registers, "Long" and "Short" Immediate (literal) operands and the contents of any one of 16 Control Engine Pointer Registers. Operand A inputs are the General Registers or External Buss Input Registers. Inputs from the External Buss Registers are gated to the Primary Adder by Operating Engine Language Boards to speed up memory fetches during some types of CEDE Ministeps. The result of this operation goes to the associated External Buss Output Register as an External Command Word.

Data Mask words affect the Primary Adder operation. Masking allows the MINIFLOW programmer to perform arithmetic, logical manipulation and testing on variable fields. Masking functions are:

- Adder outputs are forced to zero in masked-out bit positions.
- Arithmetic carries are not generated in masked-out bits.
- Arithmetic carries are propagated over masked-out bits.

Table 1-1. Primary Adder Operations

Arithmetic		Logical	
$A \leftarrow A+B$		$A \leftarrow A \cdot B$	(Logical AND)
$A \leftarrow A+\overline{B}+1$	(A-B; 2's Complement)	$A \leftarrow A \cdot \overline{B}$	
$A \leftarrow \overline{A}+B+1$	(B-A)	$A \leftarrow \overline{A} \cdot B$	
$A \leftarrow A+B+COF1$	(Conditional Carry-in)	$A \leftarrow A \cup B$	(Logical OR)
$A \leftarrow A+\overline{B}+COF1$		$A \leftarrow A \cup \overline{B}$	
$A \leftarrow \overline{A}+B+COF1$		$A \leftarrow \overline{A} \cup B$	
$A \leftarrow B$	(Clear and Add)	$A \leftarrow A \mathrm{E} B$	(Exclusive OR)
$A \leftarrow \overline{B}$	(1's Complement)	$A \leftarrow A \mathrm{E} \overline{B}$	(Compare)

"A" is Operand A. "B" is Operand B. "$\overline{A}$" is the 1's complement of A. "COF1" is the Carry-out State flip-flop used as initial carry-in. "$\cdot$" represents the logical product (AND) operation. "U" represents the logical union (OR) operation. "E" represents the Exclusive OR function. " $\leftarrow$ " (Left Arrow) denotes that the result of the operation on the right is transferred to the location specified on the left.

1.3.3 Byte/Decimal Adder

The IC-9000 Byte/Decimal Adder operates on 8-bit bytes. During the Character/Decimal (CHAD) Ministep, the Byte/Decimal Adder combines an 8-bit Operand A with an 8-bit Operand B input. The result goes to the Operand A location unless the TEST bit is on. Operand A inputs are General Registers. Operand B inputs are Generator Registers and auxiliary Operand B inputs (refer to paragraph 1.3.2). For full-word operands, the byte location is also specified. Figure 1-3 shows byte boundaries within a 36-bit register word.

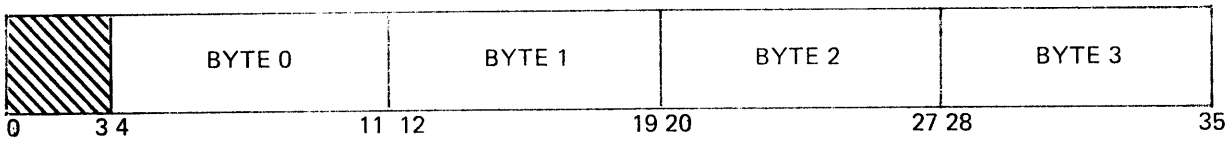


Figure 1-3. Byte Data Word Format

Four CE Pointer Registers (refer to paragraph 1.4.5) are available to tally byte and word locations in two operand character strings. Two of these four Pointers (P02 and P03) are used to indirectly address the A and B byte locations. The other two (P00 and P01) are used to hold character counters. Pointers 00 and 02 are decremented if the A byte address is obtained indirectly. Pointers 01 and 03 are decremented if the B byte is indirectly addressed. "Zero Sense" and "One Sense" State pseudo-flip-flop outputs, generated by the contents of the four Pointer Registers, are sampled to control sequencing.

The Byte/Decimal Adder executes decimal operations on binary-coded-decimal (BCD) characters (two per byte) and binary operations on 8-bit bytes, as shown in Table 1-2. Decimal arithmetic combines the two-BCD-digit A and B bytes and generates a decimal sum or difference. Both inputs are checked for validity and the result is corrected for carries. Except when masked, binary operations affect the entire 8-bit operand field. CHAD Masks are the eight least significant bits of the addressed Data Mask word.

Table 1-2. Byte/Decimal Adder Operations

Decimal	Binary		Logical
$A \leftarrow A+B$	$A \leftarrow A+B$	$A \leftarrow A+B+COF1$	$A \leftarrow B$
$A \leftarrow A+\hat{B}$	$A \leftarrow A+\overline{B}+1$	$A \leftarrow A+\overline{B}+COF1$	$A \leftarrow A \cdot B$
$A \leftarrow A+B+COF1$	$A \leftarrow \overline{A}+B+1$	$A \leftarrow \overline{A}+B+COF1$	$A \leftarrow AUB$
$A \leftarrow A+\hat{B}+COF1$			$A \leftarrow AEB$

"$\hat{B}$" represents the 9's complement of two BCD characters in the B byte.

1.3.4 Data Mask Registers and Functions

The IC-9000 Processor has 32 Data Mask Registers, each 36 bits wide. Mask Registers are divided into two banks of 16 registers. Bank selection is controlled by the "Bank Select" State flip-flop (refer to paragraph 1.4.8). Data Masking allows selective operation on bits and fields within a word. Masking is effective during GEAR, CHAD and SHIN Ministeps. A Mask Register is addressed each execution of these Ministeps. The eight, low-order bits (bits 28-35) in the selected Mask Register are active for CHAD. During SHIN, masking is effective on data in the Primary Shifter. Bit positions in the selected Mask Register which contain a one (1) are defined as "masked-in" bits. Bits containing a zero (0) are masked-out bits.

During GEAR, Masking can operate in one of two modes. In the "CLEAR" mode, all masked-out bits of the result word transferred into the Operand A register are replaced by zeros; masked-in bits are read into the register. If the CLEAR bit is off, masked-out bits inhibit alteration of the contents of corresponding bits of the receiving register. Masked-in bits will be loaded normally. The CLEAR mode is equivalent to the generation of the logical product of the Mask word and the result word. When the CLEAR bit is false (zero), masked-on bits are loaded into the result register without disturbing data in masked-off bit positions, which is equivalent to field replacement.

Data Masks also modify the operating modes of the Primary and Byte/Decimal Adders, as described in paragraphs 1.3.2 and 1.3.3, respectively. In all masked-out bit positions, the Adder output is forced to zero. This makes it possible to test the result of the operation on masked-in fields. In the Adders, Mask logic enables the propagation of carries through masked-out bits and suppresses carry generation. Arithmetic operations are executed correctly on fields shorter than the full register

word (or full byte width during the CHAD Ministep). If a carry-out signal is generated in a masked-in field it will propagate through masked-out bits. If it encounters no masked-in bits, it will be sensed as a Carry-Out signal (COP), and will be loaded into the Carry-Out flip-flop (COF1) at clock time.

Zeros (0) are shifted into vacated, masked-in, bit positions of the result word when a shift is executed. In the "Clear" mode of the GEAR Ministep, data is shifted into masked-out bit positions without being lost.

1.3.5 Shift Operations

The Operating Engine has two functionally identical Shifters. Both are 36-bits wide and have some shift paths in common. The Primary Shifter operates on the output of the Primary Adder. The Extension Shifter operates on data in odd-numbered General Registers and External Buss Registers. Up to 16 bits of cross-over data is passed when a connected shift is executed. GEAR Ministeps control only single-register shifts by the Primary Shifter. Shift Instruction (SHIN) Ministeps can execute both single or double-register shifts. SHIN codes can command connected or independent double-length shifts, single-length shifts, circular shifts (double or single-length), primitives for iterated divide and multiply algorithms, normalize shifts and indirect shifts. CEDE and TEXT Ministeps are used to input and output data in byte format using the IC-9000 shift capability. The SHIFT AMOUNT field of the GEAR, SHIN, CEDE, and TEXT Ministeps can specify one-clock-cycle shifts of left or right 0, 1, 2, 4, 6, 8, 12, and 16 bits.

SHIN indirect shifts use the Shift Control Pointer Register (P07) in the Control Engine. The 6- and 12-bit shifts cannot be executed by indirection. When an indirect shift is executed, the four least significant bits and the Logical OR of the four most significant bits in 8-bit-wide Shift Control Point Register are sampled. Since the maximum single shift is 16 bits, a 16-bit shift is taken each time logic detects a count greater than 16; i.e., a one (1) somewhere in bits 0-3 of the Shift Control Pointer. On execution, the Shift Amount count is reduced by 16. If the four most significant bits are zero, the next lower bit which contains a one (1) controls the shift. If the Shift Control Pointer has a count of 11 (binary 1011), for example, and the indirect shift is repeated, shifts of 8, 2, and 1 bit would be executed consecutively. The count in the Pointer is reduced by the amount of the shift each time. By pairing a "branch" type Control Ministep to test the SHD ("Shift Done") State pseudo-flip-flop, with a SHIN Ministep, a 1-clock-time repetitive loop can be formed to execute indirect shifts until the Pointer count goes to zero.

Multiply and Divide capability are provided by repeated execution of corresponding SHIN Ministeps. The number of iterations is initially placed in the Shift Control Pointer Register. The count in the Shift Control Pointer is reduced by one each iteration and the loop is normally terminated as the count goes to zero. Normalize shifts are controlled by outputs from the CE Language Board. A tally is kept in the Shift Control Pointer during Normalize. An output of the Language Boards, the "Shift Done" State pseudo-flip-flop, is available for testing to control the process. Specifications for SHIN Ministep functions are located in paragraph 2.2.3.

1.3.6 External Busses and Registers

Except for some miscellaneous control outputs and interrupt inputs, the IC-9000 Processor I/O interface consists of four External Busses. Each is 36 data and 4 parity bits wide. Each Buss uses an associated group of control, signalling and timing lines to sequence data transfers in both directions. Busses are bi-directional, time-shared (half-duplex) communications ports, designed to send addresses and data out and receive data back with equal facility. The IC-9000 communicates with devices on the buss by way of 8 External Buss Registers. Each Buss has a register for incoming data and a register for outgoing addresses and data. Transfers to and from the External Buss Registers are sequenced by Conditional External Data Exchange (CEDE) and Transfer External (TEXT) Ministeps. CEDE Ministeps are designed for fetching target instructions and operands from Main Memory. TEXT is used to communicate with peripherals. Various fields of incoming target language instructions are processed by the IC-9000 Language Boards under the control of the CEDE Ministep. Some of the CEDE Ministeps provide semi-automatic sequencing of instruction and operand fetches. A shift capability is provided with some CEDE and TEXT Ministeps for the input and output of byte-serial data. A number of CEDE and TEXT varieties place the processor into the "wait" mode if an external device has not responded to a request when execution begins. During waits, Ministep execution is inhibited until the desired operation occurs, or an Action Request forces a transfer of control.

The IC-9000 Processor samples four parity bits on each input transfer. If bad parity is detected during a fetch sequence, the previously transmitted external address word in the Output Register allows a retry of the fetch. This greatly improves the possibility of recovering from a transient error on the input.

The IC-9000 is designed to work with the SC-700 Memory. With proper interfacing, however, the processor can be adapted to communicate with almost any type of self-clocked memory unit, since data transfers inward have few timing constraints. Appendix A of this document is a short description of the SC-700 Memory.

1.3.7 Operating Engine Language Boards

Operating Engine (OE) Language Boards perform a wide variety of tasks related to the generation of MINIFLOW entry addresses from target language instructions and formatting address (External Command) words to Main Memory. In conjunction with a corresponding Control Engine (CE) Language Board, an OE Language Board adapts the IC-9000 to a specific target language (instruction set) and target environment (system framework). Up to four pairs of Language Boards can be installed in each IC-9000 Processor. Appendix B contains a more detailed description of the IC-9000 OE and CE Language Board inputs, outputs and typical functions.

OE Language Boards have two primary functions. One is "address modification" for accesses to Main Memory. Address modification is effective when External Command words, generated by CEDE Ministeps, initiate fetches of target language instructions and operands and generate write addresses. Address modification facilities

translate target system addresses into equivalent Main Memory addresses. Typical functions include:

- Insert Memory Protect keys
- Modify memory addresses using base register inputs
- Perform address limit violation checks
- Translate relocated page addresses
- Translate character and byte addresses to word addresses

Another major function of the OE Language Boards is to generate hard-wired, initial entry addresses to initialization MINIFLOW during execution of the CEDE/WIN (Wait for Instruction) Ministep, based on target instruction type. Initialization MINIFLOW routines include such functions as operand fetching, indexing, indirection, and some complex types of address modification. After the initialization routine is completed, the execution MINIFLOW routine is entered by branching with the assistance of the Control Engine Language Boards.

Several secondary functions can be performed by the OE Language boards, such as generating operand addresses from target instruction data. Similarly, OE Language Boards can execute fetches, using incoming data, during indirect addressing modes. Various CEDE Ministep types are implemented to provide semi-automatic addressing sequences.

1.4 CONTROL ENGINE DESIGN

Figure 1-4 is a functional block diagram of the IC-9000 Control Engine. The Control Engine can be conceptually divided into two smaller functional groups. One is made up of the Control Memory Address Generators, the Control Memory itself, its Output Registers and the Operating and the Control Engine Gating structures. The second group sequences data transfers, stores data, monitors status and controls testing and sequencing. Major operations of this part of the Control Engine include:

- Generate Control Memory address inputs.
- Translate target language instructions and extract data fields.
- Provide iteration control and indirect register addressing.
- Hold "Status" data and tally external and internal interrupts.
- Interface between the Operating and Control Engine registers.

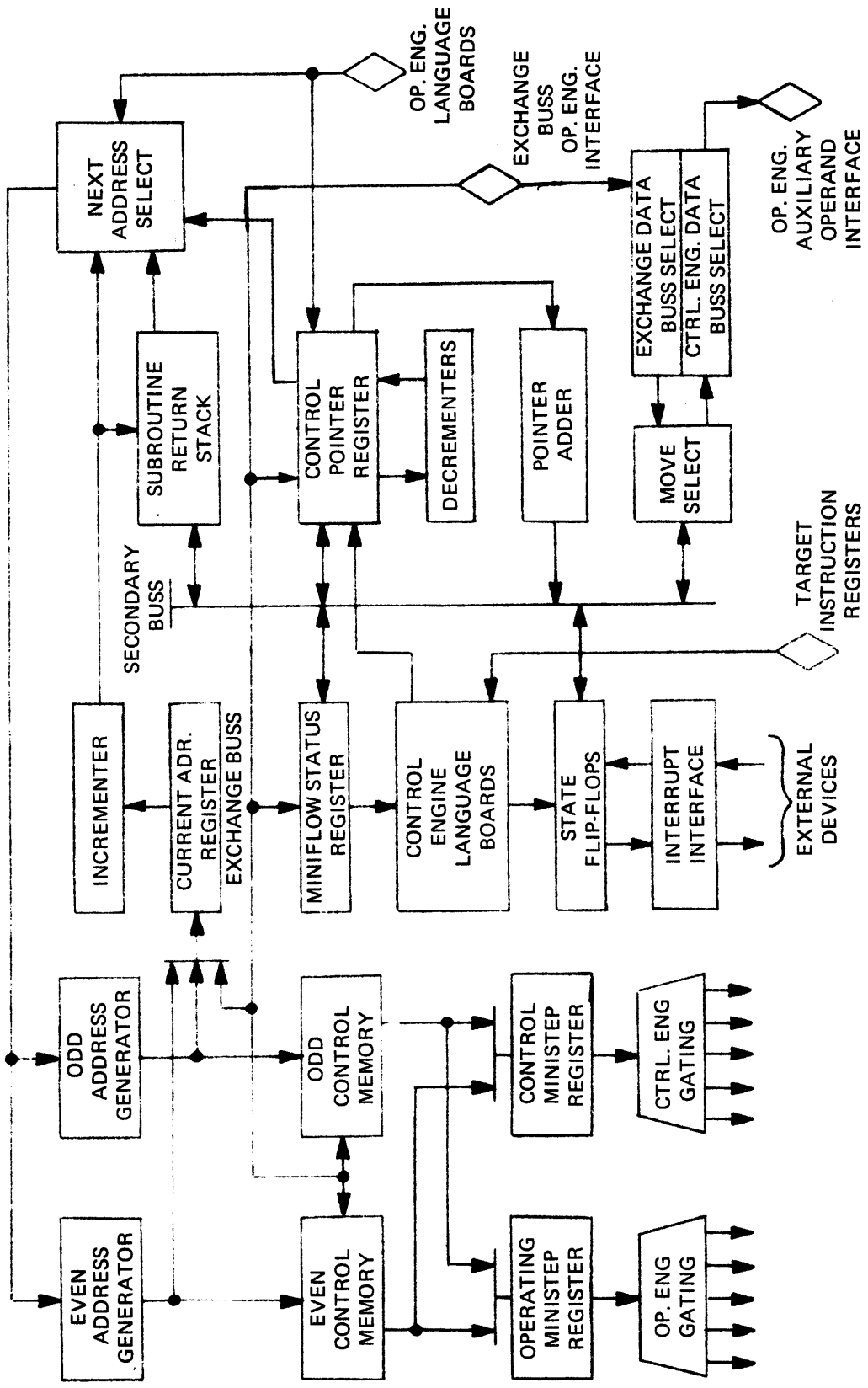


Figure 1-4. Control Engine Block Diagram

Two sequential Ministeps are fetched each clock cycle. The even and odd Control Memory banks are completely independent. Current Ministep addresses may be either even or odd without restriction. Two Ministeps will be executed simultaneously when an Operating Ministep is followed sequentially by a Control Ministep (with one exception). Only the current Ministep is executed for other sequences. Control Engine Ministeps take no additional processing time when paired. It is not necessary to pair Ministeps if there are no processing benefits. This may improve efficiency by saving some locations in Control Memory.

Control Ministeps sequence Control Engine logic just as Operating Ministeps regulate Operating Engine functions. Eight Control Ministep varieties are available. There are five Branch Ministeps, a Block Transfer Ministep for data transfers between Control Memory or the Subroutine Return Registers and the Operating Engine interface, a Ministep for manipulating the State (mode control) flip-flops and a Ministep to move data between Control Memory registers.

Branch Ministeps test State flip-flops which log the processor current status. Branch Ministeps can be used to modify the sequence of Control Memory accesses if the branch test condition is true. Branch Ministep execution can alter the sequential execution of the microprogram instruction in several ways. Least complex is the Branch Test (BRAT) Ministep, which executes relative branches based on testing one or a combination of two State flip-flops. Branch and Enter (BENT) is similar to BRAT, except taking the branch causes the return address to be loaded into the Subroutine Return Stack. The Branch Or Return (BORE) Ministep executes an automatic subroutine return if the branch is not taken. Branch and Decrement (BRAD) modifies the amount contained in one of eight Control Engine Pointer Registers by the count specified in the Ministep. One of 256 State flip-flops is tested by BRAD.

The Branch-Extended Address (BEAD) Ministep has four modes of execution. Three modes execute absolute branches, indexed and unindexed, to any location in the Control Memory. When the branch is made unindexed in one mode, a State flip-flop is tested. In the other unindexed mode, the absolute branch is taken unconditionally. The indexed absolute branch adds the contents of one of the Pointer Registers to the 16-bit branch address in the Ministep. The fourth mode adds the contents of a Control Engine Pointer Register to the continuation address and transfers. The BEAD Ministep can execute a subroutine entry in all modes.

There are three Control Ministeps other than Branches. The Manipulate Status (MAST) Ministep operates directly on State flip-flops. Any two of the State flip-flops are addressed independently and a combination of their logical states sets or resets the result flip-flop. MAST cannot affect State pseudo-flip-flops (refer to paragraph 1.4.8). Block Transfer (BLOT) loads and reads out the contents of Control Memory and the Control Engine Subroutine Stack (refer to paragraph 1.4.4) and sequences Operating Engine data transfers. BLOT is always paired with an Operating Engine CEDE, TEXT or GENT Ministep. The BLOT Ministep uses "Load Control Words" to execute chained (scatter/load) transfers into the processor. The Control Engine MOVE Ministep transfers data between Control Engine Registers and, in conjunction with the GENT, CEDE

and TEXT Ministeps, to and from the Operating Engine. Data is transferred in 8- or 16-bit bytes, depending on the MOVE mode elected. The MOVE Ministep also has a mask capability on 8-bit transfers.

1.4.1 Control Memory Addressing

Control Memory address inputs are gated from a number of sources to the Control Memory Address Generators by the Next Address Select logic. The major inputs are from the incremented Current Address Register (when no branching or transfer of control takes place), Control Engine Pointer Registers, the Subroutine Return Stack (when a subroutine is EXITED), the Branch Address fields of various Control Ministeps, and the OE Language Boards when a CEDE/WIN (Wait for Instruction) Operating Ministep is executed. The CEDE/WIN Ministep generates a starting address for MINIFLOW routines to initialize target language instruction processing.

An additional source of Control Memory addresses is the interface from the Operating Engine Exchange Buss. Another input, not shown in Figure 1-2, is a special-purpose Entry Address Generator, which provides hard-wired transfer addresses into error and interrupt MINIFLOW routines. Also not shown are Maintenance Console inputs for troubleshooting and initialization.

The Even and Odd Address Generators are adders and incrementers which form Control Memory bank addresses from the base address inputs and modifiers gated by the Next Address Select logic. Several modes of addressing are synthesized from these inputs. Departing from the nominally sequential, i.e., current address plus increment (continuation address) and current address plus increment plus one; we have (neglecting the incremented address of the second Ministep):

- Branch Relative
- Branch Absolute
- Branch Absolute plus Pointer Count
- Branch and Store Subroutine Return
- Branch and Modify Pointer Count
- Continuation plus Pointer Count
- Subroutine Return
- "Forced" Transfers

High-priority forced transfers occur immediately when internal monitoring logic detects an error Action Request interrupt condition, Lower-priority Action Requests take effect during CEDE "Wait" modes. Any of a number of lower priority Target System Interrupts can cause a forced transfer if present when a CEDE/WIN Ministep execution is attempted.

1.4.2 Control Memory Design

The Control Memory is organized into odd and even banks. Two sequential Ministeps are accessed at a time. An important feature of the IC-9000 Control Memory is that it is available in either Read-Write or Read-Only versions. The high-speed, Read-Write Storage (RWS) of the IC-9000 lets the user modify microprograms practically at will. The IC-9000 Processor with an RWS is highly adaptable and can perform processing functions in different languages (order codes) by reading in a new MINIFLOW and enabling the corresponding set of Language Boards.

Read-Write Store modules are composed of 512, 32-bit (plus parity), semiconductor registers. To change the microprogram during operation and for initialization when the processor is turned on, the RWS is loaded from an external storage element. The BLOT (Block Transfer) Ministep is used for initializing and overwriting the RWS. Due to its flexibility, RWS is highly useful for program check-out, debugging, maintenance and sequential overlay of MINIFLOW routines.

Read-Only Store (ROS) is a high-speed, 32-bit word (plus parity) memory with contents which are permanently fixed when the module is built. Data in the ROS cannot be altered by over-write, power loss or transient conditions. The contents of ROS modules must be completely specified at the time hardware is ordered. ROS modules contain 512 words (256 even and 256 odd locations).

1.4.3 Ministep Registers and Gating Functions

Even Control Memory and Odd Control Memory outputs are read into the Operating Ministep Register and Control Ministep Register, depending on sequencing and Ministep type. After the Ministep registers are loaded, their contents are further decoded and passed to gating structures which sequence the Operating and Control Engine elements. After each access of the Control Memory, the Operating and Control Ministep Registers hold the currently addressed MINIFLOW word and its next sequential successor Ministep. The current Ministep is read from the Control Memory location which has the numerically smaller address of the pair. It may be located at either an odd or an even address. The most significant bit in the OP CODE field (bit zero) determines whether the current Ministep is routed to the Operating or Control Engine Gating structure.

If the current Ministep is an Operating type and its successor is a Control type, both Ministeps will be executed simultaneously with these exceptions:

- If the current Ministep is an Operating type and it specifies a Long Immediate Data word (literal operand), the successor word is not an instruction. From the register, it is gated to the Control Engine Data Buss. From there it goes to the Operating Engine as an auxiliary Operand B input of the Primary Adder.

- CEDE/WIN Ministeps inhibit execution of any successor Ministep, since a forced branch is always taken (to an even Control Memory location).

It is possible to have a Control Ministep for the current Ministep. When this happens, the current Ministep has priority of execution over the successor Ministep. The current Ministep is routed to the Control Engine Gating when it is a Control Ministep and the successor Ministep is not executed. Likewise, if the two sequential Ministeps are both Operating Ministeps, the current Minstep has a priority of execution and it is routed to the Operating Engine Gating elements. The successor Ministep is not executed at this clock. If no branch is taken, the Current Address Register is augmented by one and the successor Ministep and its successor are read from Control Memory the next cycle.

1.4.4 Subroutine Return Stack and Stack Control

The Subroutine Return Stack is a group of 16 storage registers which hold 16-bit return addresses. A Subroutine Return Register is loaded each time a subroutine entry is executed. Entries are made when a BEAD (Branch-Extended Address), with the ENTRY bit on, or a BENT (Branch and Enter) Ministep executes a branch. Entries are also made when a breakout from the normal MINIFLOW sequence is forced by an Action Request (interrupt). Destacking occurs on a BORE (Branch Or Return) Ministep that does not take the branch.

Loading of subroutine return addresses is scheduled by the Stack Pointer Register, which addresses the top of the stack (the active return address). Pointer Register 06 is dedicated to this function. When the four least significant bits of the Stack Pointer are zero, the stack is empty. Initial entry into an empty stack is location 01. Subsequent entries go into consecutive ascending locations. A "Stack Full" Action Request occurs when location 15 is loaded. At the next clock, the now current address is entered in Stack Register 00 and a forced transfer to the Stack Full service routine is executed. A "Stack Underflow" Action Request occurs when a return is executed and the four least significant bits of the Stack Pointer Register are zero (0). The Stack Pointer does not decrement and the Action Request transfers the now current address to Stack Register 00 and goes to the Stack Underflow service routine.

Although the usable Subroutine Return stack capacity is only 15 words, with an 8-bit Stack Pointer and two MINIFLOW service routines the stack capacity can be easily extended to 240 levels (15 x 16). More complex service routines can extend the stack depth indefinitely or handle multiple stacks. However, the stack full service routine, must detect a real, upper-bound, stack overflow error, otherwise a programming error can cause subroutine entry to an infinite number of levels. Likewise, the underflow MINIFLOW routine must check for a real, lower-bound, stack limit for the same reason.

1.4.5 Pointer Registers

A group of 16 Control Engine Pointer Registers are available for counting and indirect addressing functions. Register locations 00-07 are counting Pointers. Locations 08-15 are pseudo-register (non-counting) Pointers, mechanized as sense line outputs, with functions determined by Language Boards. Most of the counting Pointers

perform special services for other Control and Operating Engine logic elements at various times. Otherwise they can be used for general processing functions. Table 1-3 is a list of Pointer assignments.

Table 1-3. Pointer Register Functions

Pointer Register	Function
00 - 03	CHAD, BLOT sequence Counters
04, 05	General Purpose (not dedicated)
06	Subroutine Stack Pointer
07	Shift Control Register
08 - 14	CE Language Boards (Pseudo-registers)
15	OE Language Boards (Pseudo-register)

Pointer Register contents indirectly address General Registers for Operating Ministeps which allow indirection on Operand A and B selection. In addition, the CHAD (Character/Decimal) Ministep (paragraph 2.2.4) allows indirect selection of the Byte A and Byte B Operands within the Operand A and B words. However, the CHAD Byte A indirect address is specified by Pointer Register 02. The Byte B indirect address is obtained from Pointer 03. Pointers 00 and 01 are normally used as character string length counters for the A and B byte operands respectively when indirect addressing occurs. These four Pointer Registers have individual decrementers. The count in the two pairs of Pointers decrements by one (1) during the execution of the CHAD Ministeps when the corresponding indirect byte address mode is specified. In addition to assisting the CHAD Ministep, Pointers 00 - 03 are also used to automatically tally locations and word counts during execution of the BLOT (Block Transfer) Ministep (refer to paragraph 2.3.6).

Pointer Registers 04 and 05 are undedicated registers and can be used for general-purpose counting or indirect addressing. Pointer 06 is the dedicated Subroutine Stack Pointer (described in paragraph 1.4.4). Pointer 07 holds and tallies indirect shift amounts for several of the Shift Instruction (SHIN) Ministep types and maintains a shift count during SHIN Normalize (refer to paragraph 2.2.3). Pointer Pseudo-registers 08 - 15 represent data translated from target instructions by the Operating and Control Engine (CE) Language Boards. Pointers 08 - 14 are driven by CE Language Boards from the Target Instruction Registers in the Operating Engine. Pointer Pseudo-register 15 is similarly driven by the OE Language Boards. Pointer Pseudo-registers can be used for indirect addressing and auxiliary Operand B inputs, but their contents must be moved to one of the counting Pointer Registers for loop control, shift operations and similar functions.

In addition to the four dedicated, single-count decrementers of Pointer Registers 00 - 03, a separate Pointer Adder can variably alter the contents of the counting Pointer specified in a BRAD (Branch and Decrement) Ministep by the amount of the

DECREMENT field. The Pointer count can be changed by any amount from plus seven (+7) to minus eight (-8) in one clock cycle.

Each Pointer Register has a State pseudo-flip-flop (ZS100-15) which goes to the one (1) logic state when the corresponding Pointer count is zero (refer to paragraph 1.4.8). These "Zero Sense" State flip-flop indicators can be tested for zero (0) logic conditions during the counting process, to maintain loop control. In addition, Pointer Registers 00 - 03 are mechanized with "One Sense" State pseudo-flip-flops (OSI00-03) which are true when the corresponding Pointer holds a one (1) count. These outputs are normally used for loop control during CHAD and BLOT Ministep execution.

Since the BRAD Ministep can increment any counting Pointer (P00-P07 by an amount other than one, it is possible to pass through a zero count from either direction. Consequently, a test of the Pointer Zero Sense pseudo-flip-flop will fail. When the Pointer Register count goes through zero, the overflow (or underflow) output of the Pointer Adder is used to drive the "Through Zero" State pseudo-flip-flop. This condition is true only during the clock cycle in which the overflow or underflow occurs. Through Zero must be sampled during the execution of BRAD to be valid. Since this output is developed relatively late in the clock cycle, a false branch condition resulting from a test of Through Zero causes a clock inhibit and a "hiccup" or one-clock-cycle delay. When testing a Zero Sense or One Sense pseudo-flip-flop of one of the Pointers, the indication is present at the start of the next clock cycle after the counting operation is complete and testing cannot cause a hiccup.

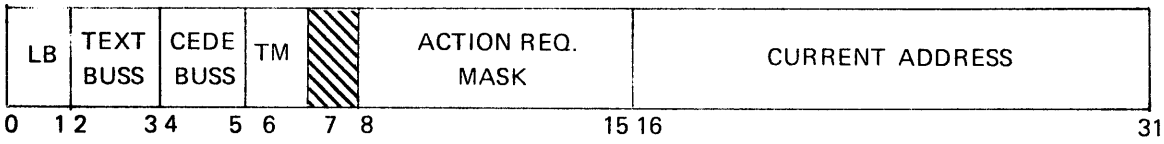


Figure 1-5. MINIFLOW Status Word Format

1.4.6 MINIFLOW Status Word

The two elements labeled in Figure 1-4 as "MINIFLOW Status Register" and "Current Address Register" are treated as a single 32-bit register by the BLOT (Block Transfer) Ministep. The composite register holds the MINIFLOW Status Word (MSW) in the format shown in Figure 1-5. The MINIFLOW Status Word facilitates initializing the processor from a cold start and when a target language changeover is executed.

The Language Board Select (LB) bits determine which Language Board set is active when executing target instructions. These two bits enable one Control and Operating Engine Language Board pair (out of a possible four). Two groups, of two bits each, separately select the External Buss address for I/O (TEXT BUSS) and Memory (CEDE

BUSS) communications. When the MSW Test Mode (TM) bit is on, the IC-9000 is in an abnormal state. This mode allows special operations to be performed to exercise and test automatic check and error Action Request logic. The true state (1) of the Test Mode bit enables:

- Writing into Control Memory with bad parity (BLOT/WBP Ministep; paragraph 2.3.6).
- Loading Operating Engine registers with bad parity (GENT Ministep; paragraph 2.2.5).
- Forcing check errors ("Check Test" State flip-flop; paragraph 1.4.8).

Action Request Mask (ACTION REQ. MASK) bits inhibit some Action Requests (AR's) or groups of AR's so that they cannot force a transfer of control (refer to paragraph 1.4.9).

A convenient way to initialize the MINIFLOW when executing a multiple block transfer (BLOT Ministep) from an external source, is to load the MSW as the final step. After loading the MSW, the Current Address Register holds the start-up address for the new MINIFLOW.

1.4.7 Control Engine Language Boards

Four Control Engine (CE) Language Boards are paired with corresponding Operating Engine (CE) Language Boards. Language Board sets are selected by two bits in the Miniflow Status Word. The principle function of CE Language Boards to to extract and translate data from fields in target language instruction words. The mechanism for performing this function includes the Operating Engine Primary and Secondary Target Instruction Registers (refer to paragraph 1.3.7) which hold the data in original format; the CE Language Boards, which perform the decomposition and translation operations; and the Control Engine Pointer Pseudo-registers (refer to paragraph 1.4.5) and the CE Language Board State pseudo-flip-flops (refer to paragraph 1.4.8) which make the outputs available to the Control Engine. Appendix B contains a more detailed description of Language Board capabilities and interfaces.

CE Language Boards are passive and contain no internal storage elements. Extracted and translated data is selectively routed to CE Pointers 08-14, which can be tested, transferred or used as addresses or operands. If counting capability is required, the contents of Pointer Pseudo-registers must be moved to one of the countable Pointers. CE Language Boards develop the "Group Entry" address into the execution MINIFLOW from the target language instruction order code and place it in a Pointer Pseudo-register. The Group Entry address is called the "target entry branch table. Other fields which are normally held in Pointer Pseudo-registers include operand and index register addresses, character and word counts, shift amount data, literal operands (up to 8 bits), etc.

Target language instructions generally contain mode control data needed by the MINIFLOW programmer. To provide access to execution mode data, a group of State

pseudo-flip-flops (CLB00-11) are driven by the CE Language Boards. Control Ministeps can sample these outputs to control execution sequencing. The "Language Board Control" State flip-flops (LBC 00-15) are sensible by CE (and OE) Language Boards. They can be used to provide program control inputs to CE Language Boards as required.

Another function of the CE Language Boards is to examine the mantissa of floating point operands for leading zeros during execution of the SHIN/NORMALIZE Ministep. The output of this sampling process is the Shift Amount control to the Primary and Extension Shifters and a count (modulo the target exponent base) to tally the amount of normalization that takes place. An output is also generated when the process is complete (SHD--"Shift Done" pseudo-flip-flop).

1.4.8 State Flip-flops

The Control Engine has a group of 256 addressable storage elements, called State flip-flops, which perform mode control functions. In most cases State flip-flops are actual, binary storage elements. Some are general-purpose and are available for scratch-pad use. Others are dedicated and are controlled by internal and external events. The remainder are synthesized from outputs of various logical elements in the processor and are called State pseudo-flip-flops. Table 1-4 is a listing of IC-9000 State flip-flops. They are grouped by characteristics and the addressing structure of the MOVE Ministep which can access them. Appendix C describes the individual State flip-flops and their functions.

- General Indicator flip-flops are not dedicated and may be used as scratch-pad registers for housekeeping and status data, program monitoring and testing.
- Language Board (LB) Control (LBC00-15) flip-flops are available to use as mode control inputs to the OE and CE Language Boards. When they are not needed for this function, they may be used as general indicators.
- The External Write (EWR0-7) group can be mechanized to provide signals to external devices, separately from the External Busses. When not dedicated they can be used as general indicators.
- The Action Request (AR) flip-flops respond to conditions which are to the IC-9000 Processor what system interrupts are to a machine language processor. They allow the normal MINIFLOW processing sequence to be modified by various internal conditions and external signals. The complement of Action Requests and a summary of their characteristics are described in paragraph 1.4.9.
- Control flip-flops are indicators which monitor status and control several important internal processor functions.
- Target System Interrupt flip-flops comprise four groups of eight bits each. For each Target System Interrupt flip-flop, there is a corresponding Interrupt Mask flip-flop. Each Interrupt Mask inhibits the true state of its corresponding Target System Interrupt flip-flop from causing a forced transfer of control immediately prior to an attempt to execute a CEDE/WIN Ministep.

Table 1-4. State Flip-Flop Listing
(Group 0)

General Indicator F/F's	General Indicator F/F's	Action Request F/F's	Control F/F's
Gen Ind GI00	LB Control LBC00	Power On	Carry Out 1 COF1
Gen Ind GI01	LB Control LBC01	Power Off	Carry Out 2 COF2
Gen Ind GI02	LB Control LBC02	Odd CM Parity	Zero Flag 1 ZRF1
Gen Ind GI03	LB Control LBC03	Even CM Parity	Zero Flag 2 ZRF2
Gen Ind GI04	LB Control LBC04	Invalid CM Addr	Invalid Digit IDF
Gen Ind GI05	LB Control LBC05	Stack Full	Invalid Sign ISF
Gen Ind GI06	LB Control LBC06	Stack Underflow	Shift Out Sign SOS
Gen Ind GI07	LB Control LBC07	MINIFLOW Trace	Shift Out Flag SOF
R00	R04	R08	R12
Gen Ind GI10	LB Control LBC10	Add-Shift Error	Last Buss In LBI
Gen Ind GI11	LB Control LBC11	Ext-Shift Error	Shift Extnsn SHE
Gen Ind GI12	LB Control LBC12	A Buss Parity	Mask Bank Sel MBS
Gen Ind GI13	LB Control LBC13	B Buss Parity	Mem Buss Inhibit MBI
Gen Ind GI14	LB Control LBC14	Extension Parity	LB Indicators LI0
Gen Ind GI15	LB Control LBC15	Mask Parity	LB Indicators LI1
Gen Ind GI16	LB Control LBC16	Aux Reg Parity	LB Indicators LI2
Gen Ind GI17	LB Control LBC17	Ext Buss Parity	LB Indicators LI3
R01	R05	R09	R13
Gen Ind GI20	Gen Ind GI40	Bad Addr Buss 0	TSI Inhibit SII
Gen Ind GI21	Gen Ind GI41	Bad Addr Buss 1	TSI Lockout SIL
Gen Ind GI22	Gen Ind GI42	Bad Addr Buss 2	Mem Cmnd 0 MC0
Gen Ind GI23	Gen Ind GI43	Bad Addr Buss 3	Mem Cmnd 1 MC1
Gen Ind GI24	Gen Ind GI44	Limit Violation	Clock Control CKC
Gen Ind GI25	Gen Ind GI45	MM Addr Compare	Run F/F RUN
Gen Ind GI26	Gen Ind GI46	Ext Call Buss 0	Check Test CKT
Gen Ind GI27	Gen Ind GI47	Ext Call Buss 1	Initiate Trace ITR
R02	R06	R10	R14
Gen Ind GI30	Ext Write EWR0	Ext Call Buss 2	AR Lockout 1 AL1
Gen Ind GI31	Ext Write EWR1	Ext Call Buss 3	AR Lockout 2 AL2
Gen Ind GI32	Ext Write EWR2	Ext Interrupt 1	AR Lockout 3 AL3
Gen Ind GI33	Ext Write EWR3	Ext Interrupt 2	AR Lockout 4 AL4
Gen Ind GI34	Ext Write EWR4	Ext Interrupt 3	AR Lockout 5 AL5
Gen Ind GI35	Ext Write EWR5	Ext Interrupt 4	
Gen Ind GI36	Ext Write EWR6	Ext Interrupt 5	Retry Cntr 0RTC0
Gen Ind GI37	Ext Write EWR7	Ext Interrupt 6	Retry Cntr 1 RTC1
R03	R07	R11	R15

Table 1-4. State Flip-Flop Listing (continued)
(Group 1)

Target System Interrupts	Target System Interrupt Masks	State Pseudo-F/F's	State Pseudo-F/F's
Interrupt SI00	Intrpt Mask, IM00	Carry Out COP	Sense SW 0 SSW0
Interrupt SI01	Intrpt Mask, IM01	Zero Sense ZSP	Sense SW 1 SSW1
Interrupt SI02	Intrpt Mask, IM02	Invalid Digit IDP	Sense SW 2 SSW2
Interrupt SI03	Intrpt Mask, IM03	Invalid Sign ISP	Sense SW 3 SSW3
Interrupt SI04	Intrpt Mask, IM04	Thru Zero THZ	Sense SW 4 SSW4
Interrupt SI05	Intrpt Mask, IM05	Wait AR Pending WAR	Sense SW 5 SSW5
Interrupt SI06	Intrpt Mask, IM06	Normalize Done NMD	RW MM Test RWM
Interrupt SI07	Intrpt Mask, IM07		~~Repeat Mode RP~~M
R00	R04	R08	R12
Interrupt SI10	Intrpt Mask, IM10	Buss Busy 0 BB0	Error Stop ERS
Interrupt SI11	Intrpt Mask, IM11	Buss Busy 1 BB1	Intrpt Pending NPT
Interrupt SI12	Intrpt Mask, IM12	Buss Busy 2 BB2	~~RW CM Test RWC~~
Interrupt SI13	Intrpt Mask, IM13	Buss Busy 3 BB3	Shift Done SHD
Interrupt SI14	Intrpt Mask, IM14	Buss Active 0 BA0	One Sense OSI00
Interrupt SI15	Intrpt Mask, IM15	Buss Active 1 BA1	One Sense OSI01
Interrupt SI16	Intrpt Mask, IM16	Buss Active 2 BA2	One Sense OSI02
Interrupt SI17	Intrpt Mask, IM17	Buss Active 3 BA3	One Sense OSI03
R01	R05	R09	R13
Interrupt SI20	Intrpt Mask, IM20	OE LB F/F OLB00	Zero Sense ZSI00
Interrupt SI21	Intrpt Mask, IM21	OE LB F/F OLB01	Zero Sense ZSI01
Interrupt SI22	Intrpt Mask, IM22	OE LB F/F OLB02	Zero Sense ZSI02
Interrupt SI23	Intrpt Mask, IM23	OE LB F/F OLB03	Zero Sense ZSI03
Interrupt SI24	Intrpt Mask, IM24	CE LB F/F CLB00	Zero Sense ZSI04
Interrupt SI25	Intrpt Mask, IM25	CE LB F/F CLB01	Zero Sense ZSI05
Interrupt SI26	Intrpt Mask, IM26	CI LB F/F CLB02	Zero Sense ZSI06
Interrupt SI27	Intrpt Mask, IM27	CE LB F/F CLB03	Zero Sense ZSI07
R02	R06	R10	R14
Interrupt SI30	Intrpt Mask, IM30	CE LB F/F CLB04	Zero Sense ZSI08
Interrupt SI31	Intrpt Mask, IM31	CE LB F/F CLB05	Zero Sense ZSI09
Interrupt SI32	Intrpt Mask, IM32	CE LB F/F CLB06	Zero Sense ZSI10
Interrupt SI33	Intrpt Mask, IM33	CE LB F/F CLB07	Zero Sense ZSI11
Interrupt SI34	Intrpt Mask, IM34	CE LB F/F CLB08	Zero Sense ZSI12
Interrupt SI35	Intrpt Mask, IM35	CE LB F/F CLB09	Zero Sense ZSI13
Interrupt SI36	Intrpt Mask, IM36	CE LB F/F CLB10	Zero Sense ZSI14
Interrupt SI37	Intrpt Mask, IM37	CE LB F/F CLB11	Zero Sense ZSI15
R03	R07	R11	R15

. The last group of 64 State flip-flops are not hardware but pseudo-flip-flops. These can be tested for either the true or complement condition but cannot be set and reset directly. A branch test on the first group of pseudo-flip-flops (Group 1, R08) causes a hiccup (one clock delay) if the test fails.

1.4.9 Action Request Servicing

There are 32 Action Request conditions in the IC-9000. The presence of an unmasked Target System Interrupt during a CEDE/WIN is also treated as an Action Request. Referring to Table 1-5, the first two Action Requests (AR's) provide a MINIFLOW entry for orderly start-up or shut-down when power is switched. The next five AR's (Priority 3-7) are set when high-priority errors or status conditions are detected in the Control Engine. Neglecting the "Trace" flip-flop, the next eight AR's function similarly for the Operating Engine. The four action requests labeled "Bad Addr" (Address) are set when an access is made to Main Memory and the bank addressed was not active in the system. This AR is also enabled if an I/O device is not present when addressed. The "Limit Violation" Action Request (Priority 21) indicates that a Main Memory address was out of the range previously defined as a boundary by optional memory protection (such as base or limit registers) which can be provided on OE Language Boards. The "External Call" Action Requests (Priority 23-26) enables units communicating with the External Busses to request servicing by the processor. The "External Interrupt" AR's provide a means for generating an interrupt from sources which are not necessarily tied to the External Busses.

The order in which the Action Requests arelisted in the table represents their priority; this is approximately the order in which servicing proceeds if more than one interrupt occurs. The first 16 AR's in the table are of such priority that a transfer out of sequence is initiated upon the occurrence of the Action Request. The "External Buss Parity" AR (Priority 16) is caused by external conditions but detected during internal transfers. The last 16 Action Requests in the table are initiated directly by conditions external to the processor. Breakout for this group is postponed until the first CEDE or TEXT Ministep with a "Wait" mode is executed. Breakouts for all Action Requests force a transfer to a dedicated location in Control Memory, called the AR Entry location. Because of timing constraints, a one-clock-time delay (hiccup) occurs at all breakouts. Entry location assignments are given in Table 1-5. The AR Entry locations, with one exception, are placed in the first few locations of Read-Write Control Memory, two locations per entry. This provides complete flexibility in handling the request, since a BEAD (Branch/Extended Address) will normally be programmed as one of the two Ministeps to transfer into a "soft" AR service routine. The contents of the AR Entry locations must be initialized when Control Memory is loaded. One exception is the "Power On" Action Request, whose entrance location is in Read-Only Control Memory. A non-volatile MINIFLOW routine is used to bring the processor up from a cold start.

When an Action Request causes a breakout from a sequence, a return address is set into the Subroutine Return stack. For most detected error conditions, the current address is used, so that a retry may be attempted. In a few cases, the continuation address is saved. Except for "Adder-Shifter" and "Extension Shifter Error", a return

Table 1-5. IC-9000 Processor Action Request Characteristics

Name	Priority	Delay	Special Action	AR Entry	Return	Mask	Lockout
Power On	1	Immed.	None	65280	No	No	PWR-ON sig
Power Off	2		None	0	Current		ARL1*
Parity, Odd CM	3		Inhibit Clock, Retry once	2			ARL2
Parity, Even CM	4		Inhibit Clock, Retry once	4			
Invalid CM ADDR	5		None	6			
Stack Underflow	6		None	8			
Stack Full	7		None	10			
MINIFLOW Trace	8		Reset Trace	12	Continue		ARL3
Adder-Shifter Error	9		None	14		ARM1	
Extsns-Shifter Error	10		None	16			
A Buss Parity	11		Inhibit Clock	18	Current		
B Buss Parity	12		Inhibit Clock	20			
Extension Parity	13		Inhibit Clock	22			
Mask Parity	14		Inhibit Clock	24			
Aux Reg Parity	15		Inhibit Clock	26			
Extrnl Buss Parity	16		Inhibit Clock	28			
Bad Addr Buss 0	17	Wait	None	30		No	ARL4
Bad Addr Buss 1	18	Mode		32			
Bad Addr Buss 2	19			34			
Bad Addr Buss 3	20			36			
Limit Violation	21			38		ARM2	
MM Addr Compare	22			40		ARM3	
Ext Call Buss 0	23			42		ARM4	ARL5
Ext Call Buss 1	24			44		ARM5	
Ext Call Buss 2	25			46		ARM6	
Ext Call Buss 3	26			48		ARM7	
Ext Interrupt 1	27			50		ARM8	
Ext Interrupt 2	28			52			
Ext Interrupt 3	29			54			
Ext Interrupt 4	30			56			
Ext Interrupt 5	31			58			
Ext Interrupt 6	32			60			

*ARL1 is set by both Power On and Power Off AR's

to the normal sequence is still possible after the MINIFLOW service routine has been executed and the condition cleared.

Some of the Action Requests are inactivated by Mask bits in the MINIFLOW Status Word (described in paragraph 1.4.6). Two of the Mask bits, Action Request Mask (ARM) 1 and 8, inhibit two Action Request groups, others allow individual requests to be masked-off as shown in Table 1-5. Requests masked-off are ignored and do not remain pending. Maskable Action Requests are active when the corresponding Mask bit is one (1), and inhibited for zero (0).

The Action Request system is mechanized with a multi-level priority structure which allows some AR's to interrupt certain others. A set of "Action Request Lockout" State Flip-flops (ARL 1 - 5) provide priority control. When a breakout occurs, the AR Lockout of that Action Request group is set. This inhibits other AR's in that group and lower priority groups. If an AR service routine is underway, and another Action Request occurs in a higher priority group, a breakout to the higher level is executed. This permits the execution of an interrupt while an interrupt is being serviced. Locked-out AR's remain pending. Lower priority AR's break out as soon as all Lockout flip-flops of higher priority are reset. If a Control Memory parity error is detected when the "Control Memory Error" AR Entry Ministep is accessed (two CM parity errors in a row), the IC-9000 stops.

1.4.10 Control Engine Data Transfers

In Figure 1-4, the "Move Select" functional block represents the control and gating elements used to move information between the various portions of the Control Engine; obtain data from the Operating Engine Exchange Buss interface; and send data back to the Operating Engine over the Control Engine Data Buss. Transfers of data are, in general, accomplished by the Control MOVE Ministep. Some subsidiary functions are performed by Move Select logic during the BRAD Ministep, to execute the "Copy A into B" test mode for "Branch" type Control Ministeps other than BRAD and BEAD and to perform the "Conditional Copy" operation during MAST. The MOVE Ministep executes 16-bit and 8-bit transfers.

State flip-flops are addressed as a group of 8-bit or 16-bit registers by the MOVE Ministep as well as one mode where a single State flip-flop can be used as the FROM address. The contents of any of the source addresses can be read out, but some destination addresses are not mechanized so that data may be read into them. This includes the State pseudo-flip-flops and Pointer Pseudo-registers 08 - 15. The Current Address Register contents cannot be altered by the MOVE Ministep. Selection logic can pick up one of four 8-bit bytes from the Operating Engine Exchange Buss if the MOVE Ministep specifies an 8-bit transfer. If the transfer is 16 bits, the most significant two bytes or the least significant two bytes will be picked up as a unit and transferred to the destination register. Control Engine Register addresses and a description of the MOVE Ministep can be found in paragraph 2.3.8.

In addition to the Control MOVE Ministep, the Block Transfer (BLOT) Ministep transfers data blocks of words to and from various groups of IC-9000 registers (in conjunction with the GENT Ministep) or between the External Buss interfaces and register groups (paired with a CEDE or TEXT Ministep). These functions are described in paragraph 2.3.6.

SECTION 2
MINISTEP FORMATS AND CONVENTIONS

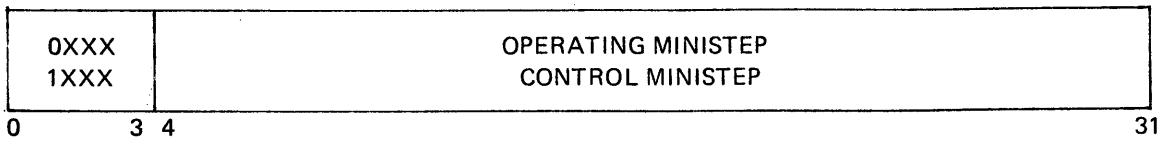


Figure 2-1. Basic Ministep Formats

2.1 GENERAL

IC-9000 Ministeps are 32 bits long and are divided into Operating and Control varieties. One exception occurs where the 32-bit word immediately following a current Operating Ministep is processed as data. This occurs when the B SELECT field in an Operating Ministep specifies a Long Immediate Data word. A 36-bit operand is synthesized, using the LONG IMMEDIATE EXTEND field in the Operating Ministep to supply the four high-order bits and the 32-bit successor word for the low-order part of the operand.

Bits are numbered 0 through 31, left to right. The leftmost bit in a word, field or sub-field is the most significant bit (MSB) for addressing, decoding, use as an operand, etc. The least significant bit (LSB) is to the right. Ministep names are generally acronyms which are four letters long and form an English word, except for the MOVE Ministep, which transfers (moves) data between Control Engine registers. The names of Ministeps and fields are capitalized. Where the same bit is used to perform multiple functions in different modes of execution, the alternatives are shown in the drawing as subdivided fields.

2.2 OPERATING INSTRUCTIONS

Operating Ministep formats are shown in Figure 2-2. Abbreviations and symbols follow a common notation where possible. Notation conventions for describing Operating Ministeps are shown below. Less commonly used designators are defined at their first appearance.

"A" is Operand A. "B" is Operand B. "$\overline{A}$" is the 1's complement of A. "COF1" is the "Initial Carry-Out" State flip-flop, which functions as a conditional carry-in for some arithmetic operations. "·" denotes the logical product (AND) operation. "U" denotes the logical union (OR) operation. "E" denotes the Exclusive OR function. "←" (Left Arrow) denotes that the value of the right-hand term replaces the contents of the elements to the left.

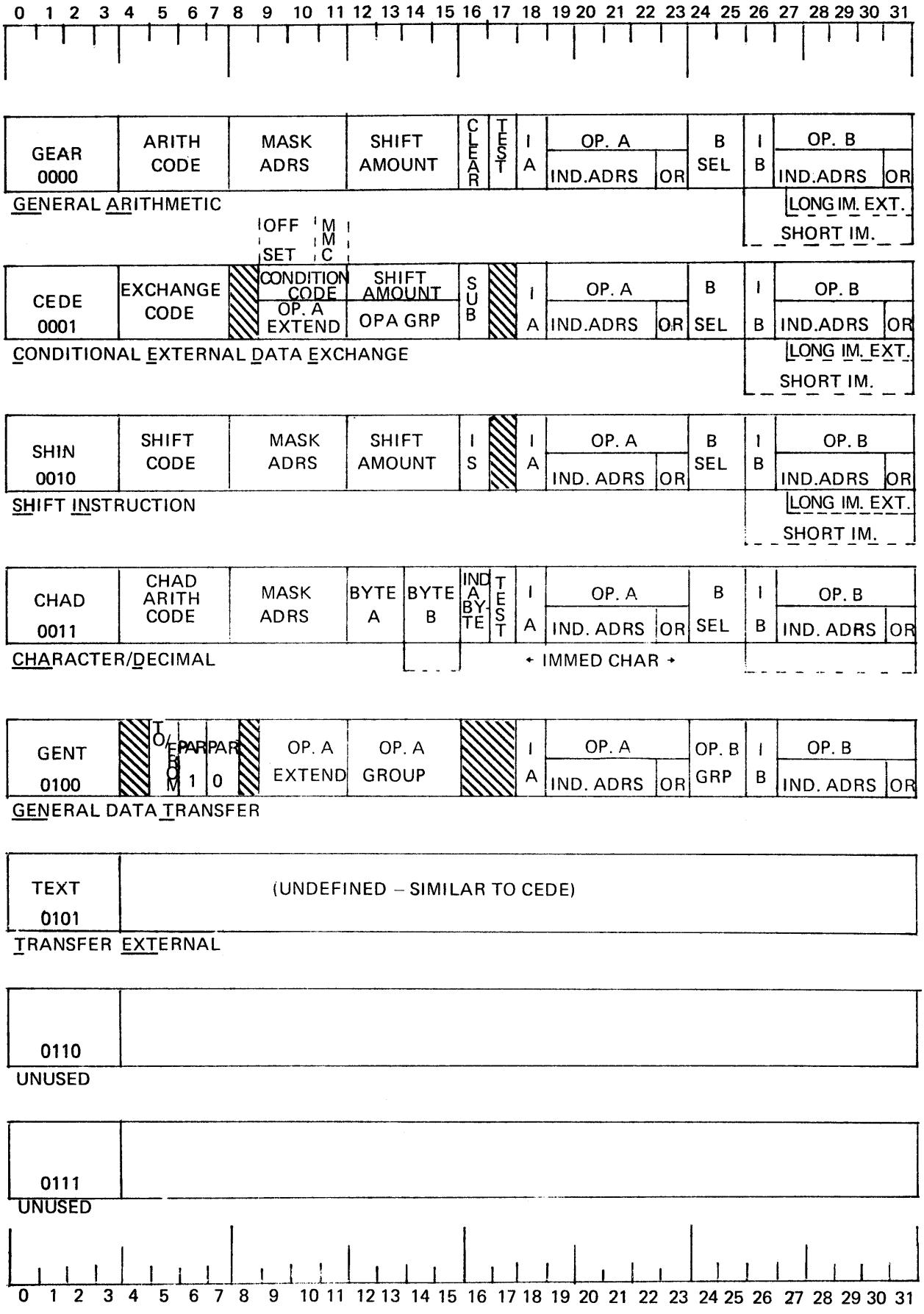


Figure 2-2. Operating Ministeps

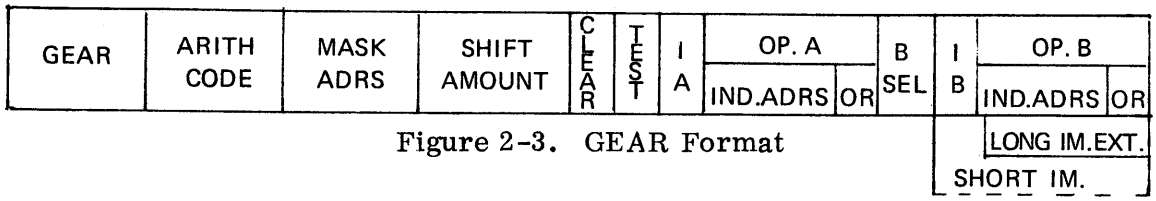


Figure 2-3. GEAR Format

2.2.1 GEAR - General Arithmetic

GEAR selects two operands and a Mask, routes them to the Primary Adder and specifies a shift of the result through the Primary Shifter. The OPERAND A and OPERAND B addressing structure of GEAR is repeated with some variations in the other Operating Ministeps.

ARITHMETIC CODE (ARITH CODE), Bits 4-7: The Arithmetic Code field commands one of 16 binary arithmetic and logical operations on OPERAND A and B inputs. Table 2-1 lists the Primary Adder operations and coding.

MASK ADDRESS (MASK ADRS), Bits 8-11: Specifies one of 16 Mask Registers. Mask Bank selection is accomplished by the MBS State flip-flop. Masking is always enabled for GEAR.

- Zero is forced in masked-out bits.
- One (1) bits in mask words are masked-in bits in the result.
- Carry generation is suppressed in masked-out bits.
- Carries propagate over masked-out bits.

SHIFT AMOUNT, Bits 12-15: Specifies a single-length shift on the result word. Coding is given in paragraph 2.2 (SHIN Ministep). Programmable shift amounts are left and right 0, 1, 2, 4, 6, 8, 12, and 16 bits. Bits shifted out of the result are lost, except that during left shifts, the last bit shifted out of the result word goes to the "Shift Extension" State flip-flop (SHE). Bits shifted in at the left end of the register are zeros or ones, as controlled by the "Shift Out Sign" State flip-flop (SOS). Bits shifted in at the right end of the word are zeros.

CLEAR, Bit 16: This bit is the mode specification for the masking operation. When CLEAR is true (1), masked-out bits (as translated by the SHIFT AMOUNT) are cleared to zero in the corresponding bits of the result (OPERAND A) register. When the CLEAR is zero, masked-out bits are undisturbed and zeros are shifted into unmasked fields from masked-out bit positions. Masked-in bits are copied into the result register in both cases for a SHIFT AMOUNT of zero. When CLEAR is false and shifts other than zero are specified, bits shifted into masked-out bit positions are lost. CLEAR true is a "No Op" when the TEST bit is true.

TEST, Bit 17: Inhibits transfer of the result word into the OPERAND A register when on (1). Permits testing status outputs of the result word without altering OPERAND A register contents.

INDIRECT A; B (IA; IB), Bits 18: 26: Indirect address control for OPERAND A and B selections. When the B SELECT coding specifies the General Registers, both IA and IB perform idential functions. IB is inactive for other B SELECT modes. When IA and IB are true (1), the respective indirect address (IND ADRS) field specifies one of the 16 Pointer Registers in the Control Engine. The contents of the five LSB's in the specified Pointer Register address General Registers as operands indirectly.

OPERAND A; B (OP A; OP B) Bits 19-23; 27-31: These address fields select one of 32 General Registers as OPERAND A and OPERAND B inputs. The OPERAND A mode is active when the IA bit is off (0). The OP B mode is active when the B SELECT mode specifies the General Registers for OP B and the IB bit is off.

INDIRECT ADDRESS A; B (IND ADRS), Bits 19-22; 27-30: Selects one of 16 CE Pointer Registers as an INDIRECT ADDRESS Register. When active, the least significant five bits of the specified Pointer address one of the General Registers. The A IND ADRS specifies a Pointer when the IA bit is one (1). The B IND ADRS field specifies a Pointer when B SEL specifies the General Registers (B SEL=0) and the IB bit is one (1); or when the contents of a Pointer are used as an 8-bit operand (B SEL=1).

OR (A; B), Bits 23-31: Modifies the indirect address in the Pointer Register by logically OR'ing into the least significant bit (LSB) position. If the LSB in the Pointer Register is a one (1), the OR bit specification does not change the operand address. If LSB is zero (0), a one (1) OR bit can alternately select one member of an even/odd register pair (normally used for double-register-length arithmetic).

B SELECT (B SEL), Bits 22 and 25: Selects one of four sources of Operand B inputs. The coding of the B SELECT Field is:

B SELECT Code	OPERAND B Input
0	General Registers
1	Pointer Registers
2	Short Immediate Data
3	Long Immediate Data

For General Register inputs (B SEL=0), the OP B specification is identical to that of the OP A field, both direct and indirect. B SEL=1 specifies the Pointer Registers, as addressed in the IND ADRS field, as an 8-bit auxiliary OP B input. This input is copied into the eight, LSB positions of the OP B input. OP B is zero in all other positions. B SEL=2 inputs bits 25-31 as a Short Immediate (literal) operand into the six LSB positions of OP B. Upper bits in OP B are zero. B SEL=3 specifies the Long Immediate Operand. The Long Immediate Operand is formed from the successor word from the Control Memory plus the 4-bit LONG IMMEDIATE EXTENSION (LONG IM EXT) to form up a complete, 36-bit, OP B input word. The 32-bit word from Control Memory goes into the 32 LSB positions of the OP B word and bits 28-31 in the OP B field are copied into the four MSB positions.

Table 2-1. Arithmetic Codes and Functions

ARITHMETIC CODE	PRIMARY ADDER OPERATION	
09	$A \leftarrow A+B$	
14	$A \leftarrow A+\overline{B}+1$	(A-B; 2's complement subtract)
10	$A \leftarrow \overline{A}+B+1$	(B-A)
11	$A \leftarrow A+B+COF_1$	(Conditional carry-in)
12	$A \leftarrow A+\overline{B}+COF_1$	
13	$A \leftarrow \overline{A}+B+COF_1$	
02	$A \leftarrow B$	(Clear and Add)
07	$A \leftarrow \overline{B}$	(1's complement)
05	$A \leftarrow A \cdot B$	(Logical AND)
01	$A \leftarrow A \cdot \overline{B}$	
03	$A \leftarrow \overline{A} \cdot B$	
06	$A \leftarrow AUB$	(Logical OR)
00	$A \leftarrow AU\overline{B}$	
04	$A \leftarrow \overline{A}UB$	
15	$A \leftarrow AEB$	(Exclusive OR)
08	$A \leftarrow AE\overline{B}$	(Compare)

State Flip-Flops Affected. Several State flip-flops and pseudo-flip-flops may be active during GEAR. They are:

- Carry Out Pseudo (COP)
- Carry-out Flip-Flop 1 (COF1)
- Carry-out Flip-Flop 2 (COF2)
- Zero Sense Pseudo (ZSP)
- Zero Register Flip-Flop 1 (SRF1)
- Zero Register Flip-Flop 2 (ZRF2)

(For SHIFT AMOUNTs other than zero)

- Shift Out Sign (SOS)
- Shift Out Flag (SOF)
- Shift Extension (SHE)

Their logic functions are shown in Table 2-2.

Table 2-2. State Flip-Flop Functions (GEAR)

FUNCTION	CONDITIONS	MECHANIZATION
Carry-out	ARITH CODES 09, 14, 10, 11, 12, 13	COF2 ·COF1· COP
Register Zero	All ARITH CODES Except Initial Carry-in (COF1)	ZRF2 ·ZRF1· ZSP
Register Zero With Carry-in	ARITH CODES 11, 12, 13	ZRF2 ·ZRF1· ZRF1 · ZSP
Sign Control	Right Shift ≠ 0	Bits shifted in copy SOS
Significance Check	Left Shift ≠ 0	Bits shifted out are compared to SOS. Comparison false sets SOF.
Register Extension	Left Shift ≠ 0	Last bit shifted out of register goes to SHE.

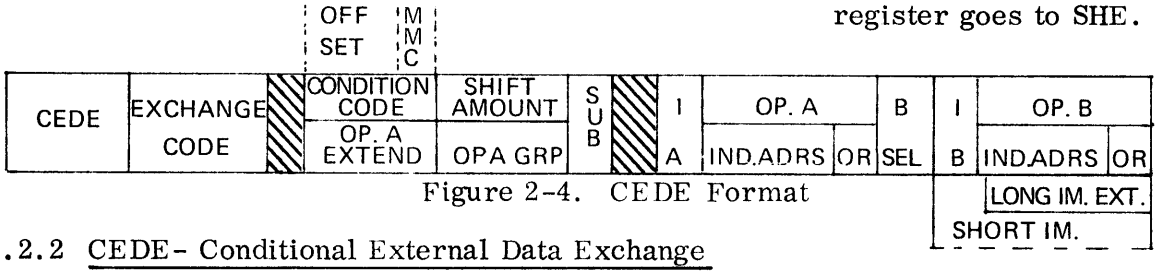


Figure 2-4. CEDE Format

2.2.2 CEDE- Conditional External Data Exchange

CEDE transfers data addresses and control codes between Main Memory and the Operating Engine External Registers. The active External Buss is specified in the MINIFLOW Status Word. CEDE types are mechanized to perform the sequences which fetch and translate target instructions, fetch and store operands and load MINIFLOW and processor initialization data. Corresponding timing and acknowledge inputs from Main Memory aid in sequence control. Some CEDE's place the IC-9000 Processor in the "Wait" mode. If the corresponding acknowledge signal has not been received from Main Memory, the Wait mode is entered unless an Action Request (AR) or, for the CEDE/WIN, a Target System interrupt is pending. Receipt of the correct acknowledge signal initiates execution of the CEDE Ministep. The Wait mode is not entered if the acknowledge signal is present prior to the start of CEDE execution and no AR's (or Target System Interrupts with CEDE/WIN) require servicing.

EXCHANGE CODE, Bits 4-7: Specifies one of the CEDE types for Execution. CEDE codes and functions are described in Table 2-3. A more detailed description of CEDE codes and functions can be found in Appendix D. CEDE Ministeps can be classed into three types:

- Memory Command Word Generation
- Data Transmission
- Combinatorial (Complex)

CONDITION CODE, Bits 9-11: The CEDE condition code field is usable by the MINI-FLOW programmer to provide specific inputs and control functions to Memory devices on the External Buss. These bits have no meaning to the IC-9000 Processor; however, they do have assigned functions for the SC-700 Memory unit.

OFFSET, Bits 9-10: Specifies the relative address of the data buss, referenced to the External Command Word buss, at the SC-700 Memory. If the offset field is zero (0) data to and from Memory is passed over the same buss as the incoming address. For OFF-SET's other than zero, the amount is added (modulo 4) to the number of the buss which accepted the CEDE Command Word to memory. Data words are then transferred over the OFFSET buss.

MAIN MEMORY CONTROL (MMC), Bit 11: Specifies that the address field in the Command Word is a control address and not an SC-700 Memory address. This bit is used to access registers of special-purpose devices which can be installed in the Memory cabinet, but which require separate addresses.

SHIFT AMOUNT, Bits 12-15: Coding of this field is specified in paragraph 2.2.3 (SHIN Ministep). The SHIFT AMOUNT field is active only for SHI (Shift In) and SHO (Shift Out). These two CEDE Ministeps may be used to assemble byte input data into register word format for SHI and take bytes from register words for transmission when SHO is executed.

OPERAND A (Composite): Includes OP A EXTEND, OP A GROUP, IA and OP A fields as specified below. This addressing mode is active for WOP (Wait for Operand) and WAS (Wait for Acknowledge and Store).

OPERAND A EXTEND (OP A EXTEND), Bits 9-11: An extension of the OP A address field. Affords an 8-bit, OP A span of 256 directly addressable register locations. Used with the OP A GRP to access up to 1024 Auxiliary Register or OE Language Board locations.

OPERAND A GROUP (OP A GRP), Bits 12-15: Divides Operating Engine Registers (and the Control Engine interface) into related groups of registers for addressing purposes. The coding of the OP A GRP field is shown below:

OPERAND A GROUP CODES

OP A GROUP	REGISTER ASSIGNMENT
0000	General Registers
0001	Unassigned
0010	Miscellaneous
0011	Data Mask Registers
0100	Aux. Bank 0
0101	Aux. Bank 1
0110	Aux. Bank 2
0111	Aux. Bank 3
10XX	Control Engine
1100	OE Language Board Bank 0
1101	OE Language Board Bank 1
1110	OE Language Board Bank 2
1111	OE Language Board Bank 3

The location assignment in the Miscellaneous Group (GROUP code 0010) is:

MISCELLANEOUS GROUP REGISTERS

LOCATION	NAME
00	Data Entry Switches
01	Main Memory Addr. Switches
02	Processor Addr. Switches
03	Unused
04	Primary Instruction Reg.
05	Secondary Instruction Reg.
06	Unused
07	Unused
08	External Register In 0
09	External Register In 1
10	External Register In 2
11	External Register In 3
12	External Register Out 0
13	External Register Out 1
14	External Register Out 2
15	External Register Out 3

SUBTRACT (SUB), Bit 16: Specifies performance of a two's complement subtraction. This bit is only active for CEDE varieties which combine two operands in the Primary Adder. When the bit is a one (1), two's complement subtraction is executed by the Primary Adder when the EXCHANGE CODE enables the operation. When the SUB bit is zero, an addition of the two operands is executed in the Primary Adder.

OPERAND A; B (Composite): Bits 18-31: Coding format is identical to GEAR Ministep (refer to paragraph 2.2.1) except both operands are not specified for some CEDE varieties.

Notes on Table Symbols and Callouts:

"EO" is an External Output Register.
"EI" is an External Input Register.
"ADR" is the SC-700 Memory Address field, bits 10-31 in the EO.
"CM BRANCH ADR" denotes a Control Memory MINIFLOW "initialization" entry (even location only).
"SE" is the Shift Extension Register.
"±" denotes an add or subtract operation, specified by the SUB field.
"OAD" is an operand address field extracted from an incoming word.
"LB" denotes the Operating Engine Language Boards.
"IR" denotes the OE Target Instruction Registers.

Table 2-3. CEDE Exchange Codes

CODE	MNEMONIC	FUNCTION	DESCRIPTION
00	FIN	$EO \leftarrow (A \pm B)_{ADR}$	Fetch Instruction
01	WIN*	CM BRANCH ADR $\leftarrow EI_{LB}$ A, $EO \leftarrow (EI_{OAD} \pm B)$** IR $\leftarrow$ EI	Wait for Instruction; Execute MINIFLOW entry; Generate Address and Save; Fetch Operand; Load Target Instruction in Register.
02	FOP	A, $EO_{ADR} \leftarrow (A \pm B)$	Fetch Operand
12	WOP	A $\leftarrow$ EI	Wait for Operand and Load
08	WOF	A $\leftarrow$ EI $EO_{ADR} \leftarrow B$	Wait for Operand and Load; Fetch Operand
15	WON	A $\leftarrow$ EI $EO \leftarrow B_{ADR}$	Wait for Operand and Load; Fetch Instruction
07	WIF	A, $EO_{ADR} \leftarrow (EI_{OAD} \pm B)$**	Wait for Indirect Word; Fetch Operand
03	SOP	$EO_{ADR} \leftarrow (A \pm B)$	Store Operand (Write Request)
09	WAS	EI, EO $\leftarrow$ A	Wait for Addr. Acknowledge and Store Data
10	UNUSED		

*Inhibited if Action Request or Target System Interrupt is pending.
**Inhibited if operand fetch unnecessary.

Table 2-3. CEDE Exchange Codes (continued)

CODE	MNEMONIC	FUNCTION	DESCRIPTION
11	SHO	EO ← A_{ODD} (0-17) A_{EVEN} (18-35); or A_{EVEN} (0-17) A_{ODD} (18-35); Left and Right Shift, respectively A_{ODD} ← A_{ODD} (shifted)	Shift on Output
13	WSI	A, SE ← EI (shifted)	Wait for Operand, Shift on Input
14	ROW	EO_{ADR} ← EO_{ADR} (Generate Command Strobe)	Retry Output Word
04	GAD	A ← (EO_{ADR} ± B)	Generate Address and Save
05	RMW	EO_{ADR} ← (A ± B)	Read/Modify/Write Cycle Request
06	WSS	Write Inhibit Output	Wait for Command Acknowledge and Suppress Store

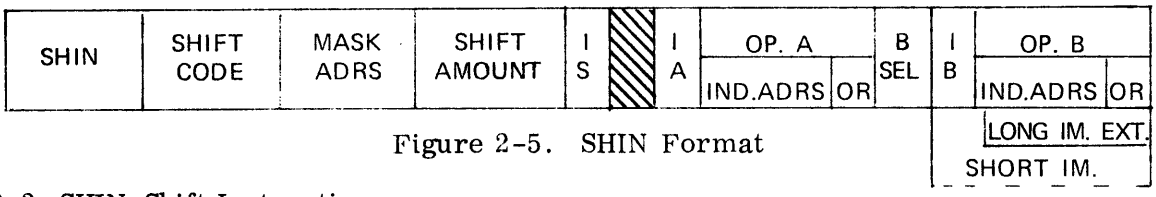


Figure 2-5. SHIN Format

2.2.3 SHIN-Shift Instruction

The SHIN Ministep controls single and double register shifts in several modes and provides the basic functions for performing Multiply, Divide and Normalize.

SHIFT CODE, Bits 4-7: Specifies one of 11 shift modes. A complete list of the SHIN codes and functions is given in Table 2-4. During double-length register pair shifts, the OP A field will normally address the even register and the odd address is implied. If the odd address is specified for a register pair shift, the OP A input and the OP B input are both the odd register, effectively operating as a one-register circular shift. When a SHIN Ministep which calls for a double-length shift is executed, the Extension Shifter functions with the Odd General Registers. Single-length shifts may use any General Register as an operand. Some SHIN Ministeps execute a double-length Shift with the Shift Extension Register. Both even or odd General Registers can be specified as OP A when the Shift Extension Register takes part in the Shift. OP B is used for addressing only with SHIN/MULTIPLY and DIVIDE, where it holds the Multiplicand and Divisor, respectively.

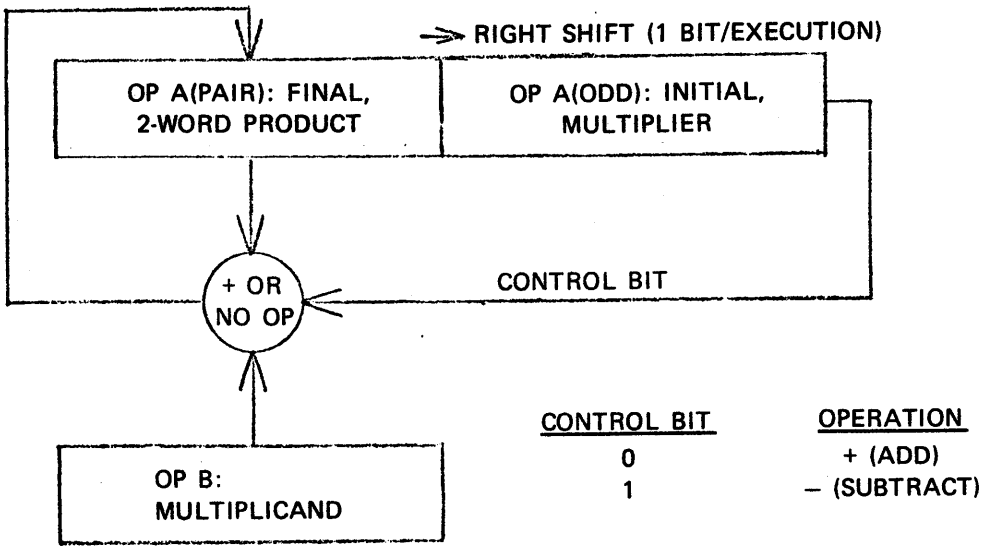


Figure 2-6. SHIN/MULTIPLY Flow Diagram

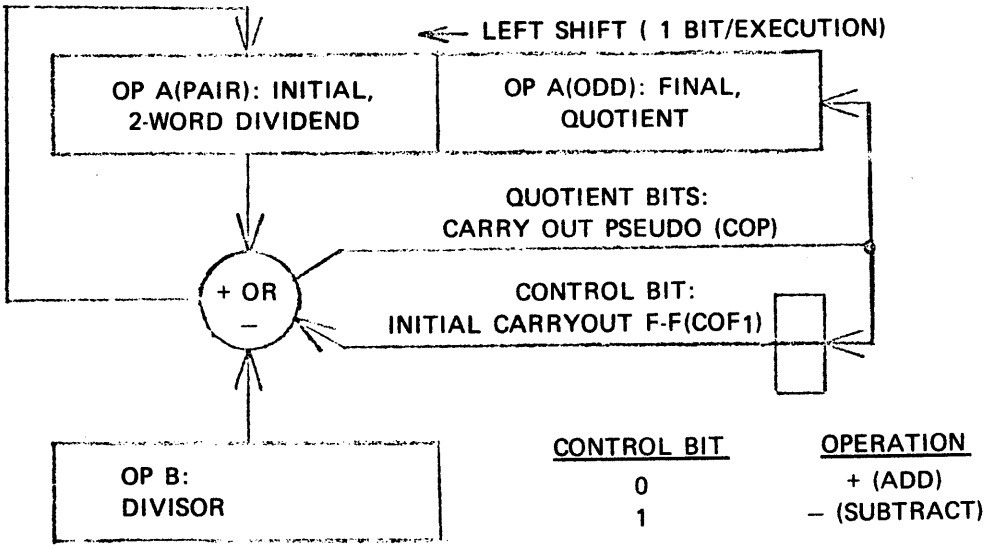


Figure 2-7. SHIN/DIVIDE Flow Diagram

During NORMALIZE and DIVIDE, when the two-register pair specified by OPERAND A is used in a target language format which does not require a 72-bit word length, data in the even register (upper half) must be right justified and the data in the odd register (lower half) must be left justified. For multiply, the multiplier in the odd register is left justified. Multiplicands and Divisors in the OPERAND B register (or auxiliary input locations) must be right justified, as they are processed relative to the even register of the OPERAND A pair. A flow diagram of the MULTIPLY and DIVIDE operation is shown in Figure 2-6 and Figure 2-7. NORMALIZE shifts are sequenced by the CE Language Boards and are executed in a manner similar to indirectly controlled shifts. A tally of the shifted amount is generated in the Shift Control Pointer Register (P07) by outputs from the CE Language Board.

MASK ADDRESS (MASK ADRS), Bits 8-11: Specifies one of the 16 addressable Mask Registers as described in paragraph 2.2.1 (GEAR Ministep). Masking affects only the input to the Primary Shifter and never the implied register of an even/odd register pair. Masked-out bits in the OP A register remain unchanged, since there is no CLEAR option on the SHIN Ministep.

SHIFT AMOUNT, Bits 12-15: Specifies the number of bits to be shifted. A zero (0) in bit 12 specifies a right shift. A one (1) in this bit specifies a left shift. Coding of the rest of the field (bits 13-15) is treated as a 3-bit control number. Coding specifies the following shifts:

Shift Amount Code	0	1	2	3	4	5	6	7
Shift Span	0	1	2	4	6	8	12	16

INDIRECT SHIFT (IS), Bit 16: Controls the source of shift control inputs to the Shifters. If the IS bit is one (1), the five LSB positions in CE Pointer Register 07 specify the amount of the shift, except where one or more of the three high-order bits is a one (1). In this event, a 16-bit shift is executed.

A shift amount of 16 bits or less, and which is a power of two (1, 2, 4, 8, 16), can be executed in one clock cycle. Indirect Shift execution out of the Shift Control Register, for the five LSB's, is related directly to the highest-order bit that is on. Indirect Shifts are executed in the following order of precedence:

Bit Pattern (Bits 11-15)	Shift Amount
1XXXX	16
01XXX	8
001XX	4
0001X	2
00001	1

At the time of execution, the amount of the shift is subtracted from the contents of the Shift Control Register. By pairing a Branch Ministep to test the "Shift Done" State pseudo-flip-flop, indirectly specified shifts up to the counting capability of the Pointer Register can be iteratively executed. BRAD (Branch and Decrement) Control Ministeps should not be paired with indirect shifts for simultaneous execution.

OPERAND A; B (Composite), Bits 18-31: Coding format is identical to paragraph 2.2.1 (GEAR Ministep) except both operands are not needed for some SHIN varieties.

State Flip-Flops Affected. The Shift Out Sign (SOS), Shift Out Flag (SOF) and Shift Extension (SHE) State flip-flops are active for all except double-length circular shifts. Logic mechanization is the same as for GEAR (refer to Table 2-3).

Table 2-4. Shift Codes and Functions

SHIFT CODE	OPERATION	SHIFT TYPE
0	A(EVEN) → A(ODD	CONN. LONG SHIFT
1	A(ODD)→ A(EVEN)	CONN. LONG SHIFT
2	A	SINGLE SHIFT
3	A(EVEN)/A(ODD)	DISCONN. DUAL SHIFT
4	A(EVEN) → A(ODD) → A(EVEN)	LONG CIRCLE SHIFT
5	A→ SE	CONN. LONG SHIFT
6	SE → A	CONN. LONG SHIFT
7	A→ SE→ A	LONG CIRCLE SHIFT
8	NORMALIZE: A(ODD) → A(EVEN)	CONN. LONG LEFT SHIFT
9	SHIFT MULTIPLY; A(EVEN → A(ODD) OP B holds Multiplicand	CONN. LONG RIGHT SHIFT
10	SHIFT DIVIDE: A(ODD→ A(EVEN) OP B holds Divisor	CONN. LONG LEFT SHIFT
11-15	UNASSIGNED	

" → " (Right Arrow) indicates that bits shifted out of the register on the left of the arrow are shifted into the register on the right, independently of shift direction. "/" (slash) indicates that no transfer of bits takes place. "A(EVEN)" and "A(ODD)" are a register pair, with A(ODD) implied when the OPERAND A address is even. "SE" denotes the Shift Extension Register (G31).

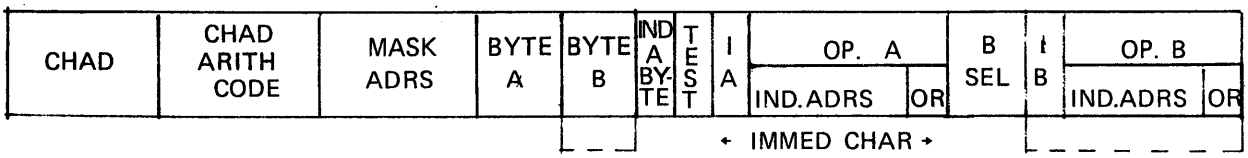


Figure 2-8. CHAD Format

2.2.4 CHAD - Character/Decimal

CHAD operates on the 32 low-order bits of the Operand A and Operand B inputs (see Figure 2-8). These 32 bits are grouped into four 8-bit bytes (bits 4-11, 12-19, 20-27, and 28-35). OP A and OP B selection incorporates direct and indirect addressing to the byte level. For decimal operations, the CHAD Ministep treats the 8-bit byte as two BCD digits. Invalid signs and digits are detected and indicated by State flip-flops and pseudo-flip-flops (IDF, IDP, ISF, ISP). Carry correction is automatic.

CHARACTER ARITHMETIC CODE (CHAR ARITH CODE), Bits 4-7: Selects decimal arithmetic, binary arithmetic and logical operations of 8-bit bytes. Table 2-5 lists the codes and functions of the Byte/Decimal Adder. Decimal arithmetic operations treat operands as two, 4-bit, binary-coded decimal (BCD) characters. Logical and binary arithmetic operations are executed on 8-bit bytes. The OP A Byte normally holds the result of the operation (TEST bit off).

MASK ADDRESS (MASK ADRS), Bits 8-11: Specifies one of 16 Data Mask Registers for the Mask word. For the CHAD Ministep, only the low-order eight bits of the Mask Word are active. Masking operations and functions are identical to GEAR.

BYTE A: B: Bits 12 and 13; Bits 14 and 15: Specifies byte positions within the OP A and OP B inputs, respectively.

INDIRECT A BYTE (IND A BYTE), Bit 16: Enables indirect specification of the Byte A address within the OP A word. When on (1), the two low-order bits of CE Pointer Register 02 provide the indirect Byte A Address. Pointer Registers 00 and 02 are decremented by one count each execution of CHAD when byte A is indirectly addressed. Branch Ministeps can test Pointer "Zero Sense" and "One Sense" pseudo-flip-flops and control the sequencing of operations to process character strings. For such operations, Pointer 02 is normally used to maintain the OP A byte position in the register word and Pointer 00 is used for the OP A character count in a variable-length string.

TEST, Bit 17: Identical to GEAR function. When on (1), no registers are changed by executing the Ministep but adder outputs can be tested by Control Ministeps.

OPERAND A (Composite), Bits 18-23: Same format as GEAR.

B SELECT (B SEL), Bits 24 and 25: Selects one of four sources of Operand B inputs. Coding of the B SELECT Field is:

B Select Code	B Operand Input
00	General Registers
01	Pointer Registers
02	Immediate Character
03	Indirect B Byte

For B SEL 00 and B SEL 01, the OP B inputs are the same as for the GEAR Ministe. B SEL 02 specifies the 8-bit IMMEDIATE CHARACTER (IMMED CHAR) as the OP B input. The IMMED CHAR input is a composite of bits 14, 15, and 26-31 of the Ministep in that order. B SEL 03 enables the two LSB's of Pointer Register 03 to address the byte position within the OP B input word. Pointer Registers 01 and 03 will be decremented each CHAD iteration when B SEL 03 is specified. Pointer 01 is normally used to hold the OP B byte count for processing a character string.

State Flip-Flops Affected

Several State flip-flops and pseudo-flip-flops are active during CHAD. They are:

- Carry Out Pseudo (COP)
- Carry-out Flip-Flop 1 (COF1)
- Carry-out Flip-Flop 2 (COF2)
- Zero Sense Pseudo (ZSP)
- Zero Register Flip-Flop 1 (ZRF1)
- Zero Register Flip-Flop 2 (ZRF2)
- Invalid Digit Pseudo (IDP)
- Invalid Digit Flip-Flop (IDF)
- Invalid Sign Pseudo (ISP)
- Invalid Sign Flip-Flop (ISP)

Their functions are shown in Table 2-6.

Table 2-5. Byte/Decimal OP Codes

OP CODE		OPERATION
	DECIMAL	
00	A ← A+B	(Add)
04	A ← A+$\hat{B}$+1	(Sub)
01	A ← A+B+COF1	(Add, Conditional Carry)
05	A ← A+$\hat{B}$+COF1	(Subtract, Conditional Carry)
	BINARY	
08	A ← A+B	(Add)
12	A ← A+$\overline{B}$+1	(Sub, 2's complement)
10	A ← $\overline{A}$+$\overline{B}$+1	(Sub, 2's complement)
09	A ← A+B+COF1	(Add, Conditional Carry)
13	A ← A+$\overline{B}$+COF1	(Subtract, Conditional Carry)
11	A ← $\overline{A}$+B+COF1	(Subtract, Conditional Carry)
	LOGICAL	
06	A ← A·B	(AND)
14	A ← AUB	(OR)
15	A ← AEB	(Exclusive OR)
07	A ← B	(Clear and Add)
02	Unused	
03	Unused	

"$\hat{B}$" represents the 9's complement of B.

Table 2-6. State Flip-Flop Functions (CHAD)

FUNCTION	CONDITIONS	MECHANIZATION
Carry-out	CHAR ARITH Codes 00, 04, 01, 05, 08, 12, 10, 09, 13, 11	COF2← COF1← COP
Register Zero	ALL CHAR ARITH Codes Except with Carry-in (COF1)	ZRF2← ZRF1 ← ZSP
Register Zero with Carry-in	CHAR ARITH Codes 01, 05, 09, 13, 11	ZRF2← ZRF1 ← ZRF1· ZSP
Invalid Digit	All Decimal Codes 00, 04, 01, 05	IDF← IDP If any of the four BCD operand digits is greater than 9, IDP is true.
Invalid Sign	All Decimal Codes 00, 04, 01, 05	ISF ← ISP If either of the LSD's of the two operands is less than 9, ISP is true.

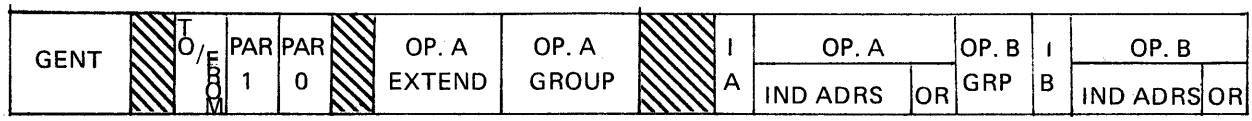


Figure 2-9. GENT Format

2.2.5 GENT-General Data Transfer

GENT performs direct transfers of the contents of Operating Engine Registers (see Figure 2-9).

TO/FROM, Bit 5: Designates whether the extended OPERAND A field is a data source or a destination. When TO/FROM is one (1), OPERAND A is moved to the OPERAND B location. When TO/FROM is zero (0), the transfer reverses direction. Data addressed by the OP A GRP and OP A fields transfers to the B GRP and OP B address.

PARITY ONE (PAR 1), Bit 6: In conjunction with PARITY ZERO bit, controls parity logic when the MINIFLOW Status Word TEST MODE (MSWTM) bit is one. Used to perform tests on parity error detection logic. Coding of these two bits is shown below.

PARITY ZERO (PAR 0), Bit 7: Same function as PAR 1 bit, except zero parity is controlled.

PARITY ONE	PARITY ZERO	PARITY BITS STATE*
0	0	Normal Parity
0	1	0
1	0	1
1	1	1

*MSW TEST bit is on (1).

OPERAND A (Composite): Includes OP A EXTEND, OP A GROUP, IA and OP A fields as specified below.

OPERAND A EXTEND (OP A EXTEND), Bits 9-11: An extension of the OP A address field. Affords an 8-bit, OP A span of 256, directly addressable register locations. Used with the OP A GRP to access up to 1024 Auxiliary Register and OE Language Board locations.

OPERAND A GROUP (OP A GROUP), Bits 12-15: Divides Operating Engine Registers (and the Control Engine interface) into related groups of registers for addressing functions. The coding of the OP A GRP field is shown below:

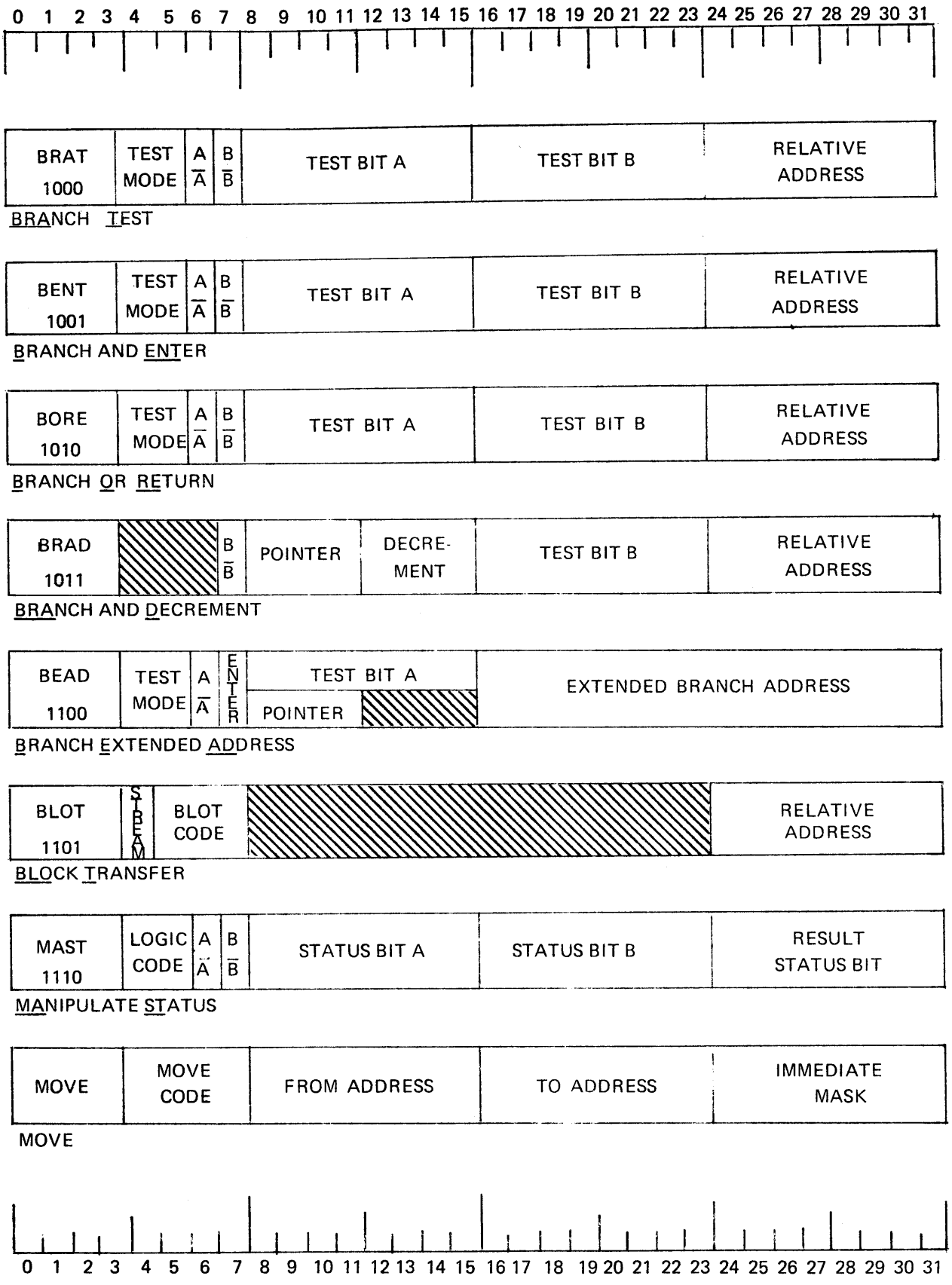


Figure 2-10. Control Ministeps

OPERAND A GROUP CODES

OP A GROUP	REGISTER ASSIGNMENT
0000	General Registers
0001	Unassigned
0010	Miscellaneous
0011	Unassigned
0100	Aux. Bank 0
0101	Aux. Bank 1
0110	Aux. Bank 2
0111	Aux. Bank 3
10XX	Control Engine
1100	OE Language Board Bank 0
1101	OE Language Board Bank 1
1110	OE Language Board Bank 2
1111	OE Language Board Bank 3

IA; OP A (Composite), Bits 18-23: Same as GEAR.

OPERAND B (Composite): Includes OB B GRP, IB and OB B fields as specified below:

OPERAND B GROUP (OP B GRP), Bits 24 and 25: Specifies register groups for OP B addressing. Several register-to-register transfers are not possible in the IC-9000, such as transfers from one Auxiliary Register to another Auxiliary Register address in the same clock cycle. Consequently, there are fewer OP B GRP assignments than OP A Group Assignments. They are:

OPERAND B GROUP CODES

OP B GROUP	REGISTER ASSIGNMENT
00	General Registers
01	Mask Registers
10	Miscellaneous
11	Control Engine

Register addresses in the Miscellaneous (both OP A and OP B) Group are shown in Table 2-7.

Table 2-7. Miscellaneous Group Registers

LOCATION	NAME
00	Data Entry Switches
01	Main Memory Addr Switches
02	Processor Addr. Switches
03	Unused
04	Primary Instruction Reg.
05	Secondary Instruction Reg.
06	Unused
07	Unused
08	External Register In 0
09	External Register In 1
10	External Register In 2
11	External Register In 3
12	External Register Out 0
13	External Register Out 1
14	External Register Out 2
15	External Register Out 3

2.3 CONTROL MINISTEPS

Control Ministep formats and field definitions are shown in Figure 2-10. In the tables in this section, A and B designators refer to TEST BIT A and TEST BIT B, respectively (instead of OPERAND A and OPERAND B). TEST BIT fields address the 256 State flip-flops and pseudo-flip-flops of the IC-9000 (refer to paragraph 1.4.1).

Individual State flip-flop addresses in all Control Ministeps, have the following format (see Figure 2-11):

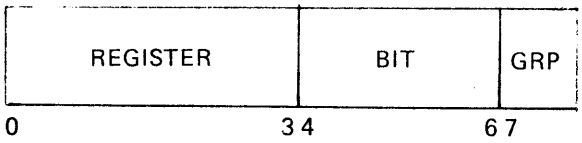


Figure 2-11. Control Ministep Individual Address Format

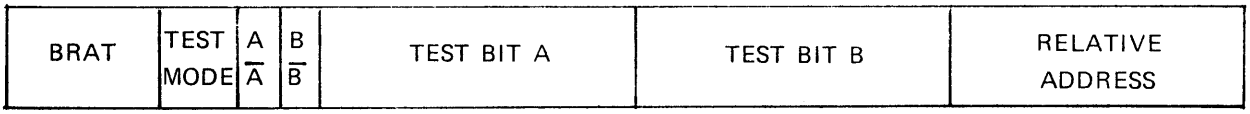


Figure 2-12 BRAT Format

2.3.1 BRAT-Branch Test

BRAT is the least complex Branch Ministep (see Figure 2-12). If the logic state specified by the TEST MODE, A/$\overline{A}$ and B/$\overline{B}$ fields is true, a branch address is generated by adding the continuation address and the contents of the RELATIVE ADDRESS field of the Ministep.

TEST MODE, Bits 4 and 5: Specifies the type of operation or logical combination of the TEST BIT A and B fields which are being sampled. When the TEST MODE condition is true (1) the branch is taken. "Move A to B" takes place whether branching occurs or not. The coding of the TEST MODE field is:

TEST MODE CODE	TEST CONDITION
0	TEST A and Move A to B (B ← A)
1	TEST AUB (OR)
2	TEST A·B (AND)
3	TEST AEB (Exclusive OR)

A/$\overline{A}$; B/$\overline{B}$, Bit 6; Bit 7: Specifies the logical states of TEST BIT's A and B, for Testing. A/$\overline{A}$, when one (1) specifies that the one side of the flip-flop is tested for the logical one (1) or set, condition. When A/$\overline{A}$ is zero (0), the test is for the zero (complement) output of the flip-flop true. The B/$\overline{B}$ field is coded in the same way. It performs the same operation for TEST BIT B, except for TEST MODE 0 (Move A to B), when it specifies whether a copy or complement move is to be taken.

TEST BIT A; B, Bits 8-15; Bits 16-23: Independently specify addresses of two of the 256 State flip-flops which are to be tested. As the two specifications are independent; the address of the same flip-flop can be coded in both fields.

RELATIVE ADDRESS, Bits 24-31: Specifies the amount by which the continuation address is to be altered. RELATIVE ADDRESS is added to the continuation address instead of the current address, in order that a zero increment has the correct position relative to all other possible branch points. The span of the relative branch address field is Continuation Address +127, -128.

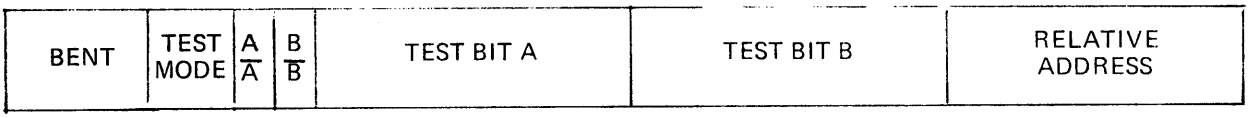


Figure 2-13. BENT Format

2.3.2 BENT-Branch and Enter

BENT is very similar to BRAT. The significant difference is that when the test is satisfied and the branch taken, a subroutine entry is executed. The continuation address (the two Ministeps that would have been accessed if the branch had not taken) is loaded into the Subroutine Return Stack. Paragraph 1.4.4 contains a description of the operation of the Subroutine Return Stack. Refer to Figure 2-13.

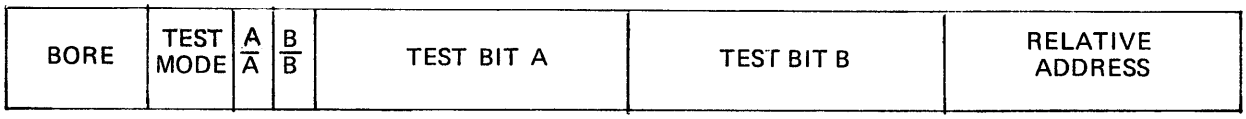


Figure 2-14. BORE Format

2.3.3 BORE-Branch or Return

BORE complements the BENT Ministep. All other specifications are the same as BRAT. If the test is satisfied, the branch is taken. If the test fails, a subroutine return is executed by extracting a return address from the active Subroutine Return Register. The Subroutine Pointer Register is decremented by one and MINIFLOW execution proceeds from the return address point. Refer to Figure 2-14.

Figure 2-15. BRAD Format

2.3.4 BRAD-Branch and Decrement

BRAD is a specialized branch Ministep which is generally used for loop and count control. It operates in only one mode, and samples TEST BIT B for the true or complement state. BRAD should not be executed in conjunction with a SHIN Ministep when the INDIRECT SHIFT (IS) bit is on or NORMALIZE code is specified. Refer to Figure 2-15.

POINTER ADDRESS (POINTER), BITS 8-11: Addresses the CE Pointer Register which is to be modified. One of the Counting Pointers (P00-07) must be addressed for BRAD to change the Pointer contents.

DECREMENT, Bits 12-15: Specifies the amount by which the Pointer Register is to be modified when BRAD is executed. Coding of the DECREMENT field uses the MSB as a sign, which gives a counting span of plus seven (+7) to minus eight (-8) in a single execution. If TEST BIT B is the "Zero Sense" flip-flop of the register specified by the POINTER field, BRAD can function as a loop or count control. However, using an incremented valus of other than one (plus or minus), it is possible to pass through zero and end up with a non-zero residue in the Pointer. To test this situation, the "Through Zero" State pseudo-flip-flop is provided. Testing must take place during the same clock cycle that the modification occurs. The Through Zero output is developed so late in the execution cycle that a one-clock-cycle delay (hiccup) occurs when the branch is not taken.

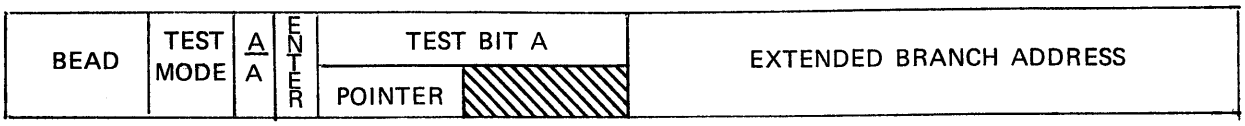


Figure 2-16. BEAD Format

2.3.5 BEAD-Branch-Extended Address

BEAD has four modes of operation. One of its major functions is to provide a capability for specifying absolute branch addresses to Control Memory.

TEST MODE, Bits 4 and 5: Specifies test and addressing variations. TEST MODE functions are:

TEST MODE CODE	TEST FUNCTIONS
0	Conditional Absolute Branch
1	Absolute Branch plus Pointer
2	Continuation plus Pointer
3	Unconditional Absolute Branch

The four variations are:

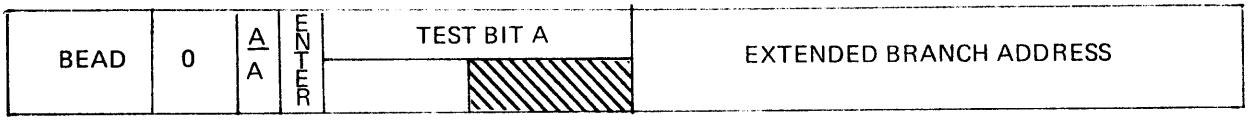


Figure 2-17. BEAD Format, Conditional Absolute Branch

2.3.5.1 Conditional Absolute Branch. If the test condition specified by the A/$\overline{A}$ field (bit 6) is true for the State flip-flop specified in TEST BIT A, the branch address in the Control Memory is specified absolutely in the 16-bit EXTENDED BRANCH ADDRESS field.

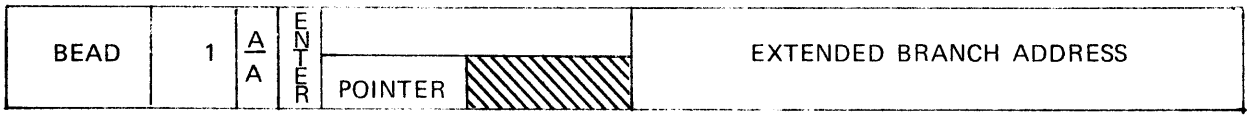


Figure 2-18. BEAD Format, Absolute Branch Plus Pointer

2.3.5.2 Absolute Branch Plus Pointer. When this BEAD is executed, the branch address is generated by adding the contents of the POINTER register specified in bits 8-11 to the contents of the EXTENDED BRANCH ADDRESS field as an 8-bit positive number.

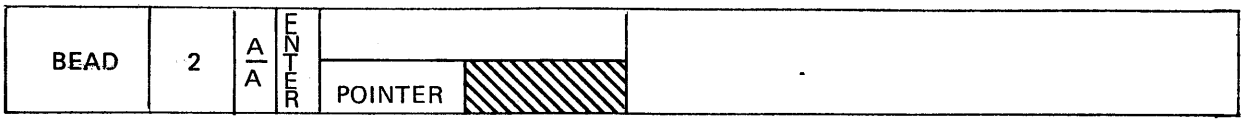


Figure 2-19. BEAD Format, Continuation Plus Pointer

2.3.5.3 Continuation Plus Pointer. In this mode of the BEAD Ministep, the absolute branch capability is not employed. The effect is similar to TEST MODE 1 in the preceding paragraph, except that the branch is taken by adding the contents of the specified POINTER Register to the MINIFLOW continuation address. The contents of the POINTER are treated as an 8-bit positive number.

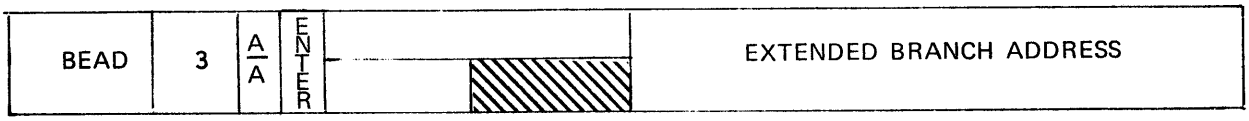


Figure 2-20. BEAD Format Unconditional Absolute Branch

2.3.5.4 Unconditional Absolute Branch. In this mode, BEAD takes the branch to the location specified in the EXTENDED BRANCH ADDRESS field unconditionally and unmodified.

The field specifications of the remaining BEAD bits are:

A/$\overline{A}$, Bit 6: This bit is active with TEST MODE 0 and specifies the test condition for TEST BIT A as in the BRAT Ministep.

ENTER, Bit 7: Executes a subroutine entry when a branch is taken. The continuation address goes to the Subroutine Return Stack. ENTER is active for all TEST MODE A types. When true, this bit causes BEAD to function similarly to the Branch and Enter (BENT) Ministep.

TEST BIT A, Bits 8-15: Active during TEST MODE 0. Addresses one of the State flip-flops for testing, as in BRAT.

POINTER ADDRESS (POINTER), Bits 8-11: Active during TEST MODE's 1 and 2. Specifies the Pointer Register whose contents are to be used to develop the branch address.

EXTENDED BRANCH ADDRESS, Bits 16-31: Holds a 16-bit, absolute branch address to Control Memory. Active during TEST MODE's 0, 1, and 3.

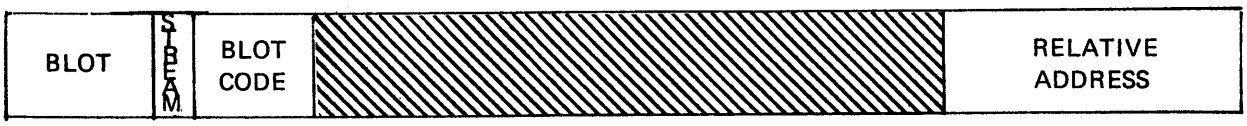


Figure 2-21. BLOT Format

2.3.6 BLOT-Block Transfer

BLOT is used to transfer multiple-word blocks of data within the processor, from an external source or to an external element. BLOT can also load in multiple (chained) blocks from external sources for processor start-up and initialization, Control Memory overlay, etc. BLOT is always teamed with CEDE, GENT or TEXT Ministeps, depending upon whether the move is to or from an external device or is an internal transfer. When transferring data to and from the Control Memory and the Subroutine Return Stack, the BLOT code controls the transfer directly, in conjunction with an Operating Ministep (CEDE, GENT or TEXT). For Operating Engine transfers and when loading multiple blocks, BLOT functions as a sequence control for the operation. Pointer Registers 00-03 are mechanized to function as counters and address generators during BLOT execution. Refer to Figure 2-21.

STREAM CONTROL (STREAM), Bit 4: The STREAM bit is effective when a CEDE/WOP (Wait for Operand) is executed with a BLOT Ministep. The function of the STREAM bit is to allow loading of multiple operand words from an input device which is itself designed for block transfers. (The block-transfer mode of read or write operation requires only a start address and an explicit or implicit block length specification.) STREAM resets the "Buss Busy" State flip-flop after the execution of the CEDE/WOP;BLOT Ministep pair so that another data word can be read in without sending a Command Word out.

BLOT CODE, Bits 5-7: Specifies the type of block transfer operation. Mnemonics for block transfers and their functions are:

CODE	MNEMONIC	FUNCTION
0	RCM	Read one block from CM; send to OE
1	WCM	Write one block into CM from OE with good parity
2	RSB	Read one block from Subroutine Stack; send to OE
3	WSB	Write one block into Subroutine Stack; from OE
4	MOE	Move one block in OE
5	WBP	Write one block into CM from OE with bad parity
6	LMB	Load Multiple Blocks

RELATIVE ADDRESS, Bits 24-31: Specifies the increment for branching until Pointer 01 (Word Counter) goes to one (1), unless Load Multiple Block (LMB) is active and the CHAIN bit in the Load Control Word is true.

2.3.6.1 Single Block Load Operations. Single block transfers use Pointer Registers 00-03 which in combination, are called the "Load Control Word". Pointer Register 00 is available for use as an indirect OE address pointer. 01 will always be used as a word counter and Pointer Registers 02 and 03 are used together as a 16-bit counter to designate a Control or Operating Engine address. When BLOT is executed, all three counters (P00, P01, P02/03) are decremented by one. BLOT execution starts at the high-order address and works down. Pointer 01 is tested for "One Sense" true and exits when the one (1) count is present. Pointers must be initialized for all single block transfers. When Pointer 01 (the word counter) has a count other than one, the contents of the RELATIVE ADDRESS field is used as the branch address. The count of one in Pointer 01 causes the continuation Ministep pair to be accessed. In order to utilize the full count span of Pointer 01 (256 words), the word count must start at zero (0), to adjust for the breakout at the count of one.

Single block transfers use a GENT;BLOT pair to transfer data between the Control Engine and the Operating Engine or between Operating Engine Register Groups. The GENT Ministep can specify both source and destination register addresses directly, but indirect addressing is generally used, with P00 and P02/03 available for use as independent Pointers. Single block load operations move data into the processor from an external source, such as Main Memory or an I/O device. Execution of this function uses a CEDE or TEXT paired with the BLOT Ministep. Where the registers are in the Operating Engine, the CEDE and TEXT Ministep OP A GRP and OP A EXTEND fields specify the data destination. When the destination registers are in the Control Memory, the appropriate BLOT code must be used to perform the transfer. Single block store operations are used to read a block of words out to an external device. If the Control Memory is the source of data, the CEDE/WAS is paired with the BLOT/RCM to perform the operation.

When the combined Pointer 02/03 is used to indirectly address registers in the Operating engine and the Subroutine Stack in the Control Engine, care must be taken to prevent rollover of the address count within a register group. This is caused by attempting to load or read a register with a higher address than is contained in the group (for instance, addressing a General Register with an address greater than 31), or counting one of the Pointers used for addressing through zero and going around to count 255.

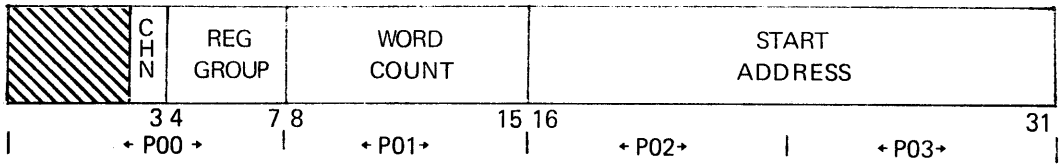


Figure 2-22. Load Control Word Format (LMB)

2.3.6.2 Multiple Block Load Operations. The Load Multiple Blocks (LMB) mode of BLOT utilizes the Load Control Word (LCW), shown in Figure 2-22, as a header for blocks which are read in from an external source. BLOT/LMB, generally in conjunction, with a CEDE/WOP is used for initializing the processor from a cold start or

for reinitializing the processor for a language changeover. The operation is basically a scatter/load technique. A sequence of self-defining data blocks can be loaded into all addressable register groups in the IC-9000 Operating Engine, Control Memory and the Subroutine Return Stack. The first word in any chained block must be the Load Control Word (LCW), which defines the number of words in the block, the register group they are to be loaded into and the starting address within the group. To load the LCW, the count in Pointer 00 must be initialized to zero. The LCW also includes a CHAIN (CHN) bit, which indicates whether or not there is another block following the current one. A count of one (1) in Pointer 02 during LMB addresses the MINIFLOW Status Word. If the CHAIN bit is true, only the 16-bit MINIFLOW Status Register is loaded and P01 is counted to zero (0) for the next LCW. The branch is taken. When the CHAIN bit is false, the Current Address Register is loaded, as well as the MINIFLOW Status Register, when P01 is one. The next Ministep pair is addressed by the Current Address Register. During LMB, the REG(ISTER) GROUP coding controls the transfer destination in conjunction with the count in P02 and P03. P00 is not decremented during LMB. Register Group assignment is as follows:

CODE	GROUP
0000	General Registers
0001	Mask Registers
0010	Miscellaneous Registers (Operating Engine)
0011	Unassigned
0100	Auxiliary Register Bank 0
0101	Auxiliary Register Bank 1
0110	Auxiliary Register Bank 2
0111	Auxiliary Register Bank 3
1000	Control Memory
1001	Subroutine Stack
1010	MINIFLOW Status Word
1100	OE Language Board Bank 0
1101	OE Language Board Bank 1
1110	OE Language Board Bank 2
1111	OE Language Board Bank 3

Figure 2-23. MAST Format

2.3.7 MAST-Manipulate Status

MAST sets or resets one State flip-flop based on a logical combination of the set or reset condition of two State flip-flops. The manipulation to be performed is specified by the LOGIC CODE field. Pseudo-flip-flops can be sensed but cannot be directly modified. Refer to Figure 2-23.

LOGIC CODE, Bits 4 and 5: Specifies the type of logical combination that is to be imposed on the RESULT STATUS BIT. The LOGIC CODE functions are shown below:

LOGIC CODE	LOGIC MODE
0	If B test = 1 then RESULT ← A (Conditional Move)
1	RESULT ← AUB (OR)
2	RESULT ← A·B (AND)
3	RESULT ← AEB (Exclusive OR)

LOGIC CODE 0 provides a conditional move of the logic state of STATUS Bit A or its complement ($A/\overline{A}$ field) to the RESULT STATUS BIT if the STATUS BIT B test (as specified by the $B/\overline{B}$ bit) is true.

The other three codes provide the logical operations of OR, AND, and Exclusive OR.

$A/\overline{A}$; $B/\overline{B}$; Bit 6; Bit 7: These fields control sampling of the set or reset condition of STATUS BIT A and B, respectively. In conjunction with the LOGIC CODE field, these bits provide a capability to specify logical combinations of the two test bits, to derive set and reset inputs to the RESULT BIT State flip-flop.

STATUS BIT A; B, Bits 8-15; Bits 16-23: Specifies the inputs to the combining logic which drives the RESULT BIT. It is possible to code TEST BIT A and B for the same State flip-flop. Addressing format is the same as the TEST BIT A and B fields of the BRAT Ministep.

RESULT STATUS BIT, Bits 24-31: Addresses the driven State flip-flop. The function specified in the LOGIC CODE field and in the $A/\overline{A}$ and $B/\overline{B}$ fields is used to reset or set the RESULT STATUS BIT, depending on the inputs and the logical mechanization of the RESULT flip-flop. Addressing format is the same as for the two STATUS BIT fields.

MOVE	MOVE CODE	FROM ADDRESS	TO ADDRESS	IMMEDIATE MASK

Figure 2-24. MOVE Format

2.3.8 MOVE-Control Engine MOVE

MOVE provides a means for moving data between registers in the Control Engine. MOVE can address all Control Engine Registers as if they are 8 bits wide. When a 16-bit transfer is specified, two, 8-bit bytes are transferred in either normal or reversed address sequence. Register addressing format, except for the FROM address of the MOVE/MOM (Move One to Many) is shown below. FROM in the MOVE/MOM is the same format for addressing individual State flip-flops shown in paragraph 2.3. Refer to Figure 2-24.

MOVE CODE, Bits 4-7: Specifies one of six different modes of operation. MOVE CODE functions are:

MOVE CODE	OPERATION	MNEMONIC	MASKABLE	DATA LENGTH
000	Move Short Immediate	MSI	YES	8 bits
001	Move One to Many	MOM	YES	8 bits
010	Move and Replace	MAR	YES	8 bits
011	Move and Complement	MAC	YES	8 bits
100	Move and Clear	MCL	YES	8 bits
101	Move Double Byte	MDB	NO	16 bits

For MSI, the FROM field is treated as an 8-bit data word and moved to the specified TO address. MOM uses FROM as a State flip-flop address (address format is the same as for the BRAT Ministep). The state of the designated flip-flop is moved to all masked-in bits of the TO register. For MAR, FROM specifies an 8-bit register whose contents replace corresponding masked-in bits in the TO location. MAC is similar to MAR, but bits which are moved are complemented. Masked-out bits in the destination register are not changed. MCL differs from other MOVE's, as masked-out bits are cleared to zero, otherwise it is identical to MAR. MDB allows two 8-bit bytes to be transferred. This MOVE is not maskable and is done on even/odd register pairs only. The two bytes will end up in normal or reversed sequence due to addressing mode. Register address in the TO and FROM Address fields are decoded as follows for 16-bit MOVES:

FROM ADDRESS	TO ADDRESS	SEQUENCE
Even	Even	Normal
Even	Odd	Reversed
Odd	Even	Reversed
Odd	Odd	Normal

FROM; TO ADDRESS, Bits 8-15; Bits 16-23: Specifies the source and destination registers in the Control Engine for MOVE. Register addresses in the Control Engine are divided into six groups, each containing up to sixteen 8-bit registers. Register address assignments are shown in Table 2-8. Addressing format is shown in Figure 2-25.

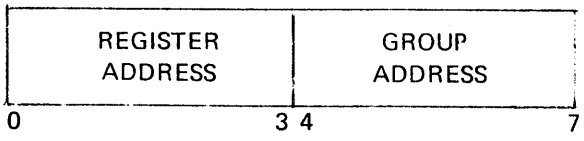


Figure 2-25 Addressing Format

The Subroutine Return Registers, although 16 bits wide, can also be addressed by 8-bit transfers. Access to the least significant eight bits of a Return address is obtained by specifying the corresponding odd register in the appropriate group (Groups 4 and 5) during an 8-bit MOVE. Byte transposition between upper and lower halves of the register can be accomplished by coding TO and FROM fields according to the rules given previously.

IMMEDIATE MASK, Bits 24-31: Specifies which destination register bits will be affected by the MOVE. Masked-in bits allow transfer without alteration. For the MOVE/MCL (Move and Clear), masked-out bits are cleared to zero in the TO location. For other MOVE CODE options, masked-out bit positions remain in their original state. The Double-byte MOVE, MDB is not maskable.

Table 2-8. Control Engine Register Address Assignments

GROUP 0 - STATE FLIP-FLOPS

GENERAL INDICATOR F/F's			
R00	R01	R02	R03
LANGUAGE BOARD CONTROL F/F's		GEN. IND. F/F's	EXT. WRITE F/F's
R04	R05	R06	R07
ACTION REQUEST F/F's			
R08	R09	R10	R11
OE CONTROL F/F's		CE CONTROL F/F's	
R12	R13	R14	R15

GROUP 1 - STATE FLIP-FLOPS

TARGET SYSTEM INTERRUPT F/F's			
R00	R01	R02	R03
TARGET SYSTEM INTERRUPT MASK F/F's			
R04	R05	R06	R07
OE PSEUDO F/F's		OE & CE PSEUDO F/F's	
R08	R09	R10	R11
MANUAL & POINTER ONE SENSE		POINTER ZERO SENSE	
R12	R13	R14	R15

Table 2-8. Control Engine Register Address Assignments (continued)

GROUP 2 - POINTER REGISTERS

F/F POINTERS			
R00	R01	R02	R03
		SUBRTN CNTR	SHIFT CNTR
R04	R05	R06	R07
LB PSEUDO-POINTERS			
R08	R09	R10	R11
R12	R13	R14	R15

GROUP 3 - MISCELLANEOUS

MINIFLOW STATUS		CURRENT ADDRESS REG	
R00	R01	R02	R03
CE DATA BUSS OUT			
R04	R05	R06	R07
OE EXCHANGE BUSS IN			
R08	R09	R10	R11

GROUP 4 - SUBROUTINE STACK REGISTERS

R00 (SUBRTN REG 00)	R02 (SUBRTN REG 01)
R04 (SUBRTN REG 02)	R06 (SUBRTN REG 03)
R08 (SUBRTN REG 04)	R10 (SUBRTN REG 05)
R12 (SUBRTN REG 06	R14 (SUBRTN REG 07)

GROUP 5 - SUBROUTINE STACK REGISTERS

R00 (SUBRTN REG 08	R02 (SUBRTN REG 09)
R04 (SUBRTN REG 10)	R06 (SUBRTN REG 11)
R08 (SUBRTN REG 12)	R10 (SUBRTN REG 13)
R12 (SUBRTN REG 14)	R14 (SUBRTN REG 15)

NOTE

TEXT FORMAT AND CODING

The TEXT Ministep is not implemented for the initial version of the IC-9000 Processor, as system applications currently scheduled do not require direct communication to external devices other than SC-700 Main Memory Modules and a special purpose Data Exchange Unit (DEU) in the Memory Cabinet. When TEXT is implemented, retrofit to installed processors will be scheduled on a non-interference basis.

The TEXT Ministep format is similar to CEDE, with TRANSFER CODES tailored for communication with a broad range of devices and I/O interfaces.

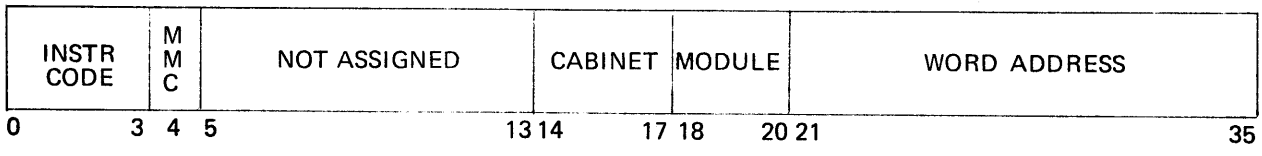
APPENDIX A SC-700 MEMORY UNIT

The SC-700 Memory Cabinet holds up to eight, 32K-word Memory Modules (K=1024). Each of the four processor busses can communicate with up to sixteen cabinets of eight Memory Modules. The nominal cycle time of the SC-700 unit is 700 nanoseconds. Read access time, after the receipt of an address and the initiation of a Read/Restore cycle, is a nominal 300 nanoseconds. Actual read access will not be this fast due to delays in the transmission of information to the processor External Buss Register. Assuming only one SC-700 Memory Cabinet, typical access times are on the order of 350 nanoseconds. There will be some minor variations due to differing transmission path lengths in the Memory Cabinet to the various Memory Modules. When more than one Memory Cabinet is used on a buss, the signals are "chained" through each cabinet by line receivers and drivers in the interface area. Additional circuit and transmission delays are incurred for each successive employment of this chaining technique. Two-way interlace of multiple Memory Modules is also available to improve effective access time in memory-limited applications.

The SC-700 Memory has four External Busses for communication with the outside world, and was designed to complement the IC-9000 Processor architecture. Any of the four IC-9000 External Busses may be interfaced to any of the four SC-700 Memory busses. It is even possible to connect more than one buss from a processor into a Memory Cabinet to different, or even the same Memory Unit. The SC-700 is designed to accept addresses from one buss, and, directed by "Buss Offset" coding in the External Command (address) word, read data from or transfer data to a "foreign" buss. The format of the command word is shown in the diagram below.

The SC-700 has two I/C interface mechanization modes. The Direct-Coupled interface (DCIO) is used where compatible signal levels, short cable runs and a benign electromagnetic environment allow. The Alternating Current interface (ACIO) uses transformer coupling and a high frequency carrier (85 mhz) to provide isolation and a capability to operate in high ambient noise environments.

Several card positions are reserved in the Memory Cabinet for implementation of functions which are conveniently handled on the Memory Buss, but which are separated from the Memory functions. These facilities allow multiple use of the Memory Buss as an aid to system implementation. The address of a control element is distinguished from a Memory location by the presence of the MMC signal (bit 4 of the External Command word).



EXTERNAL COMMAND WORD FORMAT

The SC-700 INSTRUCTION (INSTR) CODE field, bits 00 and 01 are decoded as follows:

CODE	OPERATION
00	Unused
01	Write into Location
10	Read from Location
11	Read/Modify/Write to Location

After reading out data for a Read/Modify/Write cycle, the SC-700 stays on the buss until the write cycle is initiated by receipt of a data word. No other commands can be accepted until the sequence is completed.

The other two bits of the INSTR CODE field (bits 02, 03) contain the "Relative Offset" between the buss which received the External Command word and the buss over which data is to be passed. The amount of the offset is added to the command buss number, modulo four, to obtain the offset buss address.

Bit four, the MAIN MEMORY CONTROL (MMC) bit, when true, indicates that the word address field in the External Command word is not directed to one of the SC-700 Memory Units within the addressed cabinet. The cabinet has spare module slots for incorporating added facilities (such as protect key check logic or an operator's console interface) which are required to communicate over the same External Buss used for access to Main Memory.

Bits 5 - 13 are not assigned but are potentially available to perform subsidiary control functions (such as holding a memory protect key) where needed.

Bits 14 - 17 select one of up to sixteen Main Memory CABINETS which can be chained on the buss.

Bits 18 - 20 select one of up to eight SC-700 Memory MODULES which can be accommodated in each cabinet.

Bits 21 - 35 define the WORD ADDRESS of one of 32K locations in the Memory Module (or special purpose control device in the Memory Cabinet).

In addition to the data bits of the External Command Word, there are several other lines that provide subsidiary logic and timing signals. The logic functions that are provided are:

- Parity - Four parity bits are transferred each time a data word is passed over the buss. Parity is generated at the data source and is maintained in the Memory. Parity checking is accomplished at the receiver. Each parity bit applies to a 9-bit byte.

- Presence Acknowledge - This signal is generated by the addressed Memory Module. Lack of this signal notifies the originator of the External Command word that the addressed Memory Module is unavailable or inactive.

- Store Suppress - This signal is originated by the requestor of a Store or Read/Modify/Write cycle. It causes the Memory to restore the original data in the word and not continue in the storage mode. The usual cause for suppressing a store is the detection of a Memory Protect Address Violation at the IC-9000.