IBM Field Engineering

# Theory of Operation

# 1130 Computing System

SY26-5978-4

## PREFACE

The IBM Field Engineering Theory of Operation 1130 Computing System contains information on the 1131 Central Processing Unit and console and the adapter circuits for the input/output units: console printer, console keyboard, and single disk storage. Descriptions of the adapter circuits for other features are found in the IBM Field Engineering Theory of Operation 1130 Computing System-Features (Y26-3670). Descriptions of the adapter circuits for features that require an IBM 1133 Multiplexer Control are found in the IBM Field Engineering Theory-Maintenance 1133 Multiplexer Control (Y26-4014).

Information regarding the Input/Output units may be found in the manual of instruction for each unit as listed in the IBM FE Bibliography -- 1130 Computing System (Y26-1130).

System maintenance diagram pages with AA prefixes are referred to in this manual and should be available to the reader. These diagrams are a part of the logics included in each machine shipment. The reader should also be familiar with solid logic technology (SLT) packaging and documentation.

The users of this manual are cautioned that specifications are subject to change at any time and without prior notice by IBM. Wiring diagrams (logics) at the engineering change level of that specific machine are included in each machine shipment.

Fifth Edition (May 1968)

This manual, Form Y26-5978-4 is a major revision of Form Y26-5978-3 and includes all previous supplements.

Significant changes or additions to the specifications contained in this publication are continually being made. When using this publication in connection with the operation of IBM equipment, check the latest FE Publications Systems Sequence Listing for revisions or contact the local IBM branch office.

The illustrations in this manual have a code number in the lower corner. This is a publishing control number and is not related to the subject matter.

Copies of this and other IBM publications can be obtained through IBM Branch Offices.

A form is provided at the back of this publication for your comments.

This manual was prepared by the IBM Systems Development Division, Product Publications, Department 455, San Jose, California 95114.

LIST OF ABBREVIATIONS ..... v

CHAPTER 1. INTRODUCTION ..... 1-1
DESCRIPTION ..... 1-1
OPERATION ..... 1-4
Data Flow and CPU Registers ..... 1-4
CPU Cycles ..... 1-4
Interrupt Principle ..... 1-6
Cycle Steal Principle ..... 1-9
Program Execution Sequence ..... 1-9
MACHINE LANGUAGE ..... 1-10
Core Storage Word ..... 1-10
Instruction Format ..... 1-11
Data Format ..... 1-12
INSTRUCTION SET AND OPERATIONS ..... 1-13
Load and Store ..... 1-13
Arithmetic ..... 1-13
Shift Left ..... 1-15
Shift Right ..... 1-16
Branch or Skip ..... 1-16
Wait ..... 1-19
Input/Output ..... 1-19

CHAPTER 2. FUNCTIONAL UNITS ..... 2-1
MULTI-INPUT FLIP-FLOP (FF) ..... 2-1
T-CLOCK AND TIMING ..... 2-1
CYCLE TIMER ..... 2-3
I1 FF ..... 2-3
I2 FF ..... 2-3
IX FF ..... 2-3
IA FF ..... 2-4
E FF ..... 2-4
E1 FF ..... 2-4
E2 Time ..... 2-4
E3 FF ..... 2-4
Cycle Control Counter ..... 2-4
REGISTERS ..... 2-5
Storage Buffer Register (B) ..... 2-5
Operation, Format, Tag, and Modifier Registers ..... 2-5
Arithmetic Factor Register (D) ..... 2-5
Accumulator (A) ..... 2-6
Temporary Accumulator Register (U) ..... 2-6
Accumulator Extension Register (Q) ..... 2-6
Instruction Address Register (I) ..... 2-7
Storage Address Register (M) ..... 2-7
CORE STORAGE ..... 2-7
Magnetic Core Theory ..... 2-7
Addressing ..... 2-10
Core Storage Arrays ..... 2-12
Drive Current Generation -- 3.6-Microsecond Storage ..... 2-20
Drive Current Generation -- 2.2-Microsecond Storage ..... 2-21
Inhibit/Sense -- 3.6-Microsecond Storage ..... 2-22
Inhibit/Sense -- 2.2-Microsecond Storage ..... 2-22
Timing -- 3.6-Microsecond Storage ..... 2-23
3.6-Microsecond Storage Clock ..... 2-23
Time Delay Circuit ..... 2-23
Timing -- 2.2-Microsecond Storage ..... 2-23
ADD-SUBTRACT CIRCUITS ..... 2-27
Add Circuits ..... 2-27
Subtract Circuits ..... 2-29
CYCLE STEAL CIRCUITS ..... 2-30
X-Clock ..... 2-30
Cycle Steal Controls ..... 2-30
INTERRUPT CIRCUITS ..... 2-30
Interrupt Forced 'BSI' ..... 2-31
Interrupt Priority ..... 2-32
CONSOLE KEYBOARD ..... 2-37
Keyboard Mechanics ..... 2-37
Bail and Latch Contacts ..... 2-38
Keyboard Restore ..... 2-39
Keyboard Interlocks ..... 2-40
Keyboard Electrical Functions ..... 2-40
CONSOLE PRINTER ..... 2-40

CHAPTER 3. PRINCIPLES OF OPERATION ..... 3-1
Instruction Cycles ..... 3-1
INSTRUCTION CYCLE 1 (I1) ..... 3-2
Setting Control Registers ..... 3-2
Format = 0, Tag = 00 ..... 3-2
Format = 0, Tag ≠ 00 ..... 3-3
Format = 1 ..... 3-4
INSTRUCTION CYCLE 2 (I2) ..... 3-4
INDEXING CYCLE (IX) ..... 3-4
INDIRECT ADDRESSING CYCLE (IA) ..... 3-5
Execution Cycles ..... 3-6
E-CYCLE FLIP-FLOPS ..... 3-6
Load Accumulator E1 Cycle ..... 3-6
Load and Store Instruction Group ..... 3-8
LOAD ACCUMULATOR (LD) ..... 3-8
LOAD DOUBLE (LDD) ..... 3-9
STORE ACCUMULATOR (STO) ..... 3-10
STORE DOUBLE (STD) ..... 3-10
LOAD INDEX (LDX) ..... 3-11
Tag = 00 ..... 3-11
Tag ≠ 00 ..... 3-12
STORE INDEX (STX) ..... 3-12
Tag = 00 ..... 3-12
Tag ≠ 00 ..... 3-12
LOAD STATUS (LDS) ..... 3-12
STORE STATUS (STS) ..... 3-13
Arithmetic Instruction Group ..... 3-14
ADD (A) ..... 3-14
ADD DOUBLE (AD) ..... 3-15
SUBTRACT (S) ..... 3-16
SUBTRACT DOUBLE (SD) ..... 3-16
MULTIPLY (M) ..... 3-16
1131 Multiply Algorithm ..... 3-17
Multiply Cycles ..... 3-18
DIVIDE (D) ..... 3-20
E1 Overflow Checks ..... 3-20
E2 Overflow Checks ..... 3-20

End Operation Overflow Check . . . . . 3-22
AND (AND) . . . . . 3-22
OR (OR) . . . . . 3-22
EXCLUSIVE OR (EOR) . . . . . 3-23
Shift Left Instruction Group . . . . . 3-24
SHIFT LEFT ACCUMULATOR (SLA) . . . . . 3-24
SHIFT LEFT ACCUMULATOR AND EXTENSION (SLT) . . . . . 3-25
SHIFT LEFT AND COUNT ACCUMULATOR (SLCA) . . . . . 3-25
SHIFT LEFT AND COUNT ACCUMULATOR AND EXTENSION (SLC) . . . . . 3-26
Shift Right Instruction Group . . . . . 3-28
SHIFT RIGHT ACCUMULATOR (SRA) . . . . . 3-28
SHIFT RIGHT ACCUMULATOR AND EXTENSION (SRT) . . . . . 3-29
ROTATE RIGHT (RTE) . . . . . 3-29
Branch or Skip Instruction Group . . . . . 3-31
BRANCH AND STORE IAR (BSI) . . . . . 3-31
Operation When F = 0 . . . . . 3-31
Operation When F = 1 . . . . . 3-32
BRANCH OR SKIP ON CONDITION (BSC AND BOSC) . . . . . 3-32
Skip Condition FF On . . . . . 3-33
Skip Condition FF Off . . . . . 3-33
Branch Out or Skip on Condition (BOSC) . . . . . 3-34
MODIFY INDEX AND SKIP (MDX) . . . . . 3-34
Short Format MDX, Tag = 00 . . . . . 3-34
Short Format MDX, Tag ≠ 00 . . . . . 3-35
Long Format MDX, Tag = 00 . . . . . 3-35
Long Format MDX, Tag ≠ 00 . . . . . 3-37
WAIT . . . . . 3-37
Input/Output Instruction . . . . . 3-39
EXECUTE I/O (XIO) . . . . . 3-39
XIO Sense DSW or XIO Sense ILSW . . . . . 3-40
XIO Control . . . . . 3-40
XIO Initiate Read or XIO Initiate Write . . . . . 3-41
XIO Read or XIO Write . . . . . 3-41
Console, Keyboard, and Printer Operations . . . . . 3-42
CONSOLE BIT SWITCHES . . . . . 3-42
XIO Read, Area 7 . . . . . 3-42
XIO Sense DSW, Area 7 . . . . . 3-42
CONSOLE KEYBOARD . . . . . 3-42
Sense Interrupt . . . . . 3-43
Sense Device . . . . . 3-45
Control . . . . . 3-45
Read . . . . . 3-45
CONSOLE PRINTER . . . . . 3-46
Sense Interrupt . . . . . 3-47
Sense Device . . . . . 3-47
Write . . . . . 3-47

CHAPTER 4. FEATURES . . . . . 4-1
Single Disk Storage . . . . . 4-1
DESCRIPTION . . . . . 4-1
Disk Storage Unit . . . . . 4-1
Disk Cartridge Assembly . . . . . 4-1
Access Mechanism . . . . . 4-1
Disk Organization and Capacity . . . . . 4-2
Disk Storage Timing . . . . . 4-3
Disk Storage Data Checking . . . . . 4-3
ADAPTER FUNCTIONAL UNITS . . . . . 4-4
Word Count Register . . . . . 4-4
File Data Register . . . . . 4-4
File Core Address Register . . . . . 4-5
File Check Counter . . . . . 4-5
Bit Counter . . . . . 4-6
Sector Register . . . . . 4-6
Sector Counter . . . . . 4-6
ADAPTER OPERATIONS . . . . . 4-7
Sense Interrupt . . . . . 4-7
Sense Device . . . . . 4-8
Control . . . . . 4-8
Initiate Write . . . . . 4-9
Initiate Read . . . . . 4-11

CHAPTER 5. POWER SUPPLIES AND CONTROL . . . . . 5-1
PRIMARY POWER INPUT . . . . . 5-1
POWER SUPPLY OUTPUTS . . . . . 5-1
POWER SEQUENCING . . . . . 5-1
MPS . . . . . 5-2
Mid-Pac . . . . . 5-4
POWER ON RESET . . . . . 5-4

CHAPTER 6. CONSOLE AND MAINTENANCE FEATURES . . . . . 6-1
Section 1. Console . . . . . 6-1
CONSOLE KEYBOARD . . . . . 6-1
Function Keys . . . . . 6-1
Indicator and Switch Panels . . . . . 6-2
CONSOLE PRINTER . . . . . 6-3
CONSOLE DISPLAY PANEL . . . . . 6-3
Indicators . . . . . 6-3
Switches . . . . . 6-5
Section 2. Maintenance Features . . . . . 6-7
CE SWITCHES . . . . . 6-7

APPENDIX A. MACHINE CHARACTERISTICS . . . . A-1

APPENDIX B. SPECIAL CIRCUITS . . . . . B-1

INDEX . . . . . X-1

| | | |
|---|---|---|
| ACC | - | accumulator |
| CB | - | circuit breaker |
| CCC | - | cycle control counter |
| CPU | - | central processing unit |
| CS | - | cycle steal |
| DSW | - | device status word |
| EA | - | effective address |
| E-cycle | - | execute cycle |
| F | - | format |
| FF | - | flip-flop |
| FL | - | flip-latch |
| Hz | - | hertz (cycles per second) |
| IA | - | indirect addressing |
| IAR | - | instruction address register |
| I-cycle | - | instruction cycle |
| ILSW | - | interrupt level status word |
| I/O | - | input/output |
| IOCC | - | input/output control command |
| IX | - | indexing |
| ms | - | milliseconds |
| ns | - | nanoseconds |
| Op | - | operation |
| PT | - | paper tape |
| SAC | - | storage access channel |
| SAR | - | storage address register |
| SCA | - | synchronous communications adapter |
| SP | - | sample pulse |
| SPD | - | sample pulse driver |
| XIO | - | 'execute I/O' |
| XR | - | index register |

## DESCRIPTION

- The IBM 1130 Computing System comprises a central processing unit (CPU) and some configuration of input/output (I/O) devices.
- The IBM 1131 Central Processing Unit is the controlling unit of the system.
- Core storage in the CPU may contain 4,096, 8,192, 16,384, or 32,768 addressable locations.
- Each addressable location has 18 cores for storing a 16-bit binary word and two parity bits.
- I/O devices may be contained within the CPU, connect directly to the CPU, or connect to the CPU via the storage access channel (SAC) and the IBM 1133 Channel Multiplexer.
- All operations are initiated or controlled by programs stored in core storage.
- The CPU console has operator controls, an indicator panel, a keyboard, and an output printer.

The IBM 1130 Computing System is a solid-state electronic computing system composed of a central processing unit (CPU) and a variety of optional input/output (I/O) devices and units. The 1130 system, with its wide range of configurations, is used in many different engineering computation and data processing applications.

The IBM 1131 Central Processing Unit (CPU) contains the control logic circuits, the console panel, the console keyboard, the console printer, and core storage. Core storages are available which operate in 3.6- or 2.2-microsecond cycles. All core storages with more than 8,192 (8k) positions require an extension (commonly called a blister) on the left end of the 1131 unit to house the additional core storage. The blister is also required for any core storage (minimum size is 8k) which operates at 2.2 microseconds. Core storage is available with 4,096 (3.6-microsecond cycle only), 8,192, 16,384, or 32,768 positions.

Each addressable position of core storage contains 18 cores and can store a 16-bit data or instruction word and two parity bits. Only information in binary form can be processed, and all encoding and decoding for I/O operations must be performed by stored programs.

The CPU also contains the adapter circuits for machine features, and, in some cases, the complete feature circuit (Figure 1-1). All circuits and mechanisms for single disk storage drive 0 are in the CPU when this feature is installed. Also the circuits for the synchronous communications adapter (SCA) and the storage access channel (SAC) are within the CPU when these features are installed. External I/O units for which the adapter circuits are in the CPU include:

IBM 1055 Paper Tape Punch.
IBM 1132 Printer.
IBM 1134 Paper Tape Reader.
IBM 1231 Optical Mark Page Reader.
IBM 1442 Model 5 Card Punch.
IBM 1442 Model 6 or 7 Card Reader Punch.
IBM 1627 Plotter.
IBM 2501 Card Reader.

A system can include either a 1231 or a 2501, but not both.

In addition to the I/O devices listed, SAC allows connection of one of the following devices to the system:

1. IBM 1133 Multiplex Control Enclosure.
2. IBM 2250 Display Unit, Model 4.
3. Non-IBM devices.

If SAC is used to connect an 1133, the 1133 must include storage access channel II (SAC II) if the 2250 model 4 or any non-IBM device is also to be included in the system. If the 2250 model 4 uses SAC II, no other device can be connected to SAC II.

Besides the circuits for SAC II, if specified, the 1133 contains the adapter circuits for the IBM 2310B Disk Storage and IBM 1403 Printer when these features are specified. One 1403, model 6 or 7, can be connected to the system via the 1133. One to four disk storage drives can be connected via the 1133 in addition to the single disk drive, which is basic to CPU models 2 and 3.

The model variations of the 1131 CPU are shown in Figure 1-2. The basic and optional features available for each model are shown in Figure 1-3. The 208V feature is basic to models 2C, 2D, and all

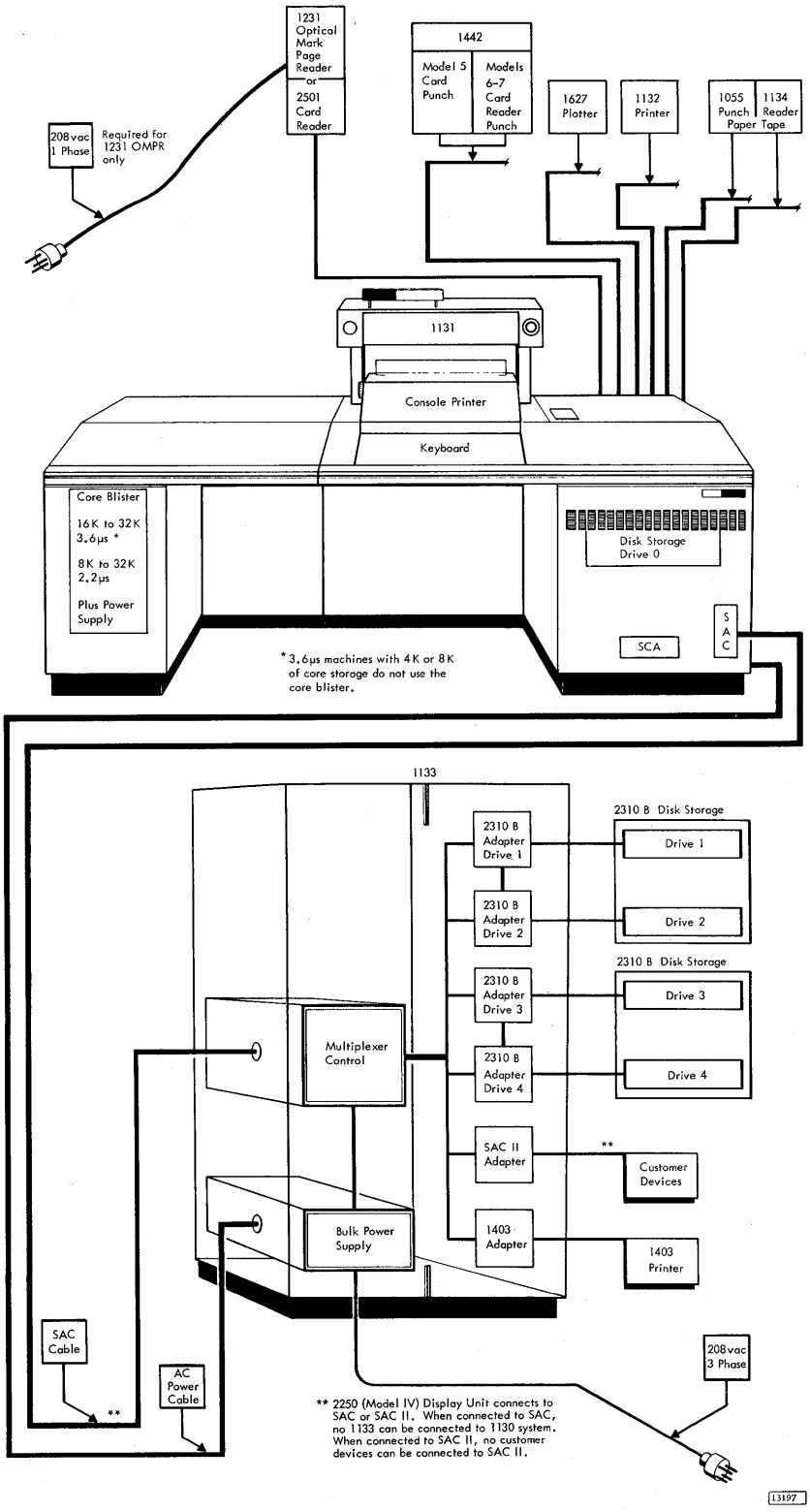



Figure 1-1. IBM 1130 System Configuration

| Model | Speed | Core Storage | Single Disk Storage | Input Power (U.S.A. Only) |
|---|---|---|---|---|
| 1A | 3.6 µs | 4K | No | 115 vac |
| 1B | 3.6 µs | 8K | No | 115 vac |
| 2A | 3.6 µs | 4K | Yes | 115 vac |
| 2B | 3.6 µs | 8K | Yes | 115 vac |
| 2C | 3.6 µs | 16K | Yes | 208/230 vac |
| 2D | 3.6 µs | 32K | Yes | 208/230 vac |
| 3B | 2.2 µs | 8K | Yes | 208/230 vac |
| 3C | 2.2 µs | 16K | Yes | 208/230 vac |
| 3D | 2.2 µs | 32K | Yes | 208/230 vac |

16212

Figure 1-2. Model Variations of 1131

model 3's but is optional on models 1A, 1B, 2A, and 2B. However, Models 2A and 2B must have the 208V feature to allow connection of a 2501 card reader, a 1231 page reader, or an 1133 Multiplexer Control.

The console on the 1131 CPU contains control keys or switches, an indicator panel, a console keyboard and a console printer. The control keys and switches are used by the operator to communicate with and control operation of the system. The indicator panel provides a visual indication of the contents of various registers and the status of control circuitry within the computer. The console bit switches are used for manual entry of data and instructions into core storage. The console printer prints serially at the rate of 15.5 characters per second.

The console bit switches, used in conjunction with other console keys, are the only means of entering information without a program in the CPU.

| Basic Features | 1131 Model 1* | 1131 Model 2* | 1131 Model 3 |
|---|---|---|---|
| Console Keyboard | A and B | A, B, C, and D | B, C, and D |
| Console Printer | A and B | A, B, C, and D | B, C, and D |
| 3.6 µs Core Storage | A and B | A, B, C, and D | |
| 115 vac Power | A and B | A and B | |
| Single Disk Storage | | A, B, C, and D | B, C, and D |
| 2.2 µs Core Storage | | | B, C, and D |
| 208 vac Power (60 Hz) | | C and D | B, C, and D |
| **Optional Features** | | | |
| 208 vac Power (60 Hz) | A and B | A and B | |
| 1055 Paper Tape Punch | A and B | A, B, C, and D | B, C, and D |
| 1132 Printer | A and B | A, B, C, and D | B, C, and D |
| 1134 Paper Tape Reader | A and B | A, B, C, and D | B, C, and D |
| 1442 Models 6 and 7 Card Read Punch | A and B | A, B, C, and D | B, C, and D |
| 1627 Plotter | A and B | A, B, C, and D | B, C, and D |
| Synchronous Communications Adapter | A and B | A, B, C, and D | B, C, and D |
| Storage Access Channel I (SAC) | A and B | A, B, C, and D | B, C, and D |
| 1133 Multiplexer Control | | A and B‡, C and D † | B, C, and D† |
| 2310 B Disk Storage | | A, B, C, and D** | B, C, and D** |
| 1403 Printer | | A, B, C, and D** | B, C, and D** |
| Storage Access Channel II (SAC II) ¤ | | A, B, C, and D** | B, C, and D** |
| 2501 Card Reader *** | | A and B‡, C and D | B, C, and D |
| 1442 Model 5 Card Punch | A and B | A and B‡, C and D | B, C, and D |
| 1231 Optical Mark Page Reader *** | | A and B‡, C and D | B, C, and D |

* Model 1 can be field changed to Model 2 or 3.
Model 2 can be field changed to Model 3.
(Only systems with midpack power supplies can be upgraded, in the field, to a Model 3D.)

† Storage Access Channel 1 required on all models.

‡ 208 vac or 230 vac 60 Hz power feature required in U.S.A. on Models A and B.

** 1133 Multiplexer Control required on all models.

¤ Required to connect any device or devices previously connected to SAC when an 1133 Multiplexer Control is connected to SAC.

*** A system can not have both a 2501 and a 1231

16213 A

Figure 1-3. Basic and Optional Features for the 1130 System

However, a brief program can be entered from a card reader or paper tape reader by using the console program load key. This brief program must be written to cause a more expanded loader program to be entered in a "bootstrap" manner. The loader program, in turn, loads object programs for execution.

The foregoing description of the IBM 1130 Computing System shows that only when a program is stored in the CPU and the CPU is executing that program can the system perform useful functions.

## OPERATION

### Data Flow and CPU Registers

- A group of registers and their control circuits provide the data handling required in the execution of programs.
- Data may be changed during transfer from one register to another.

Execution of a program consists to a great extent of transferring data between the registers of the CPU. During the transfers, the data may remain unchanged or may be modified by the circuits controlling the registers. The data flow and uses of the CPU registers are briefly described in the following paragraphs.

Figure 1-4 shows data flow in the 1130 system with emphasis on the data flow within the CPU. I/O devices are lumped and shown attached to their respective input and/or output buses. More comprehensive unit data flow diagrams for I/O units appear in the manuals for those units.

The usage of many of the data paths is immediately apparent although a few do not become apparent until a more detailed description of machine operations is given. Words read from core storage are set into the storage buffer register (B-register) except for the parity bits which enter parity checking circuits. The B-register also determines what is written into core storage, either directly or through the parity generator circuits. Data from an input device is set into the B-register for storage, and data from storage is placed on the output bus via the B-register. An instruction, read from storage and set into the B-register, can set the various parts of the operation register. The storage address register (M-register) selects the core storage location to be read and/or written.

The instruction address register (I-register) provides an address to the M-register when an instruction is to be read from storage. By the end of the execution of any instruction, the I-register has been adjusted to the address of the next instruction to be executed. The address in the I-register can also be set into the accumulator (A-register) to be used in computing the address of data.

The arithmetic factor register (D-register) can receive a word from core storage, via the B-register, for use in arithmetic operations. When an instruction is given to set a word from storage into the accumulator, the word is transferred through both the B- and D-registers.

A word in the accumulator which must be saved while the accumulator is used for another purpose can be transferred to the temporary accumulator (U-register) and later returned to the accumulator.

The accumulator extension (Q-register) can be used to double the capacity of the accumulator. There are two data paths between the A- and Q-registers. One allows bit-by-bit shifting through the two registers, and the other allows exchanging of the complete words in the A- and Q-registers. The word in the accumulator can be transferred to the B-register for entry into core storage. If the value in the Q-register is to be stored, it must first be transferred to the accumulator.

The carry and overflow indicators can be set when the D- and A-registers are involved in an arithmetic operation. The result in the accumulator can be affected by the setting of the carry indicator.

Data paths not described here are described with the appropriate operations.

### CPU Cycles

- Each instruction requires one or more cycles for execution.
- An electronic machine clock (T-clock) provides timings within cycles.
- During every clock cycle, the addressed core storage location is read out and written back. The word written back may be different from that read out.
- An I1 cycle is required for every instruction. An I2 cycle is required for some operations.
- Instruction cycles obtain instruction words from core storage.
- Instruction words determine the operation to be performed and either the effective address or the additional cycles required to calculate the effective address.

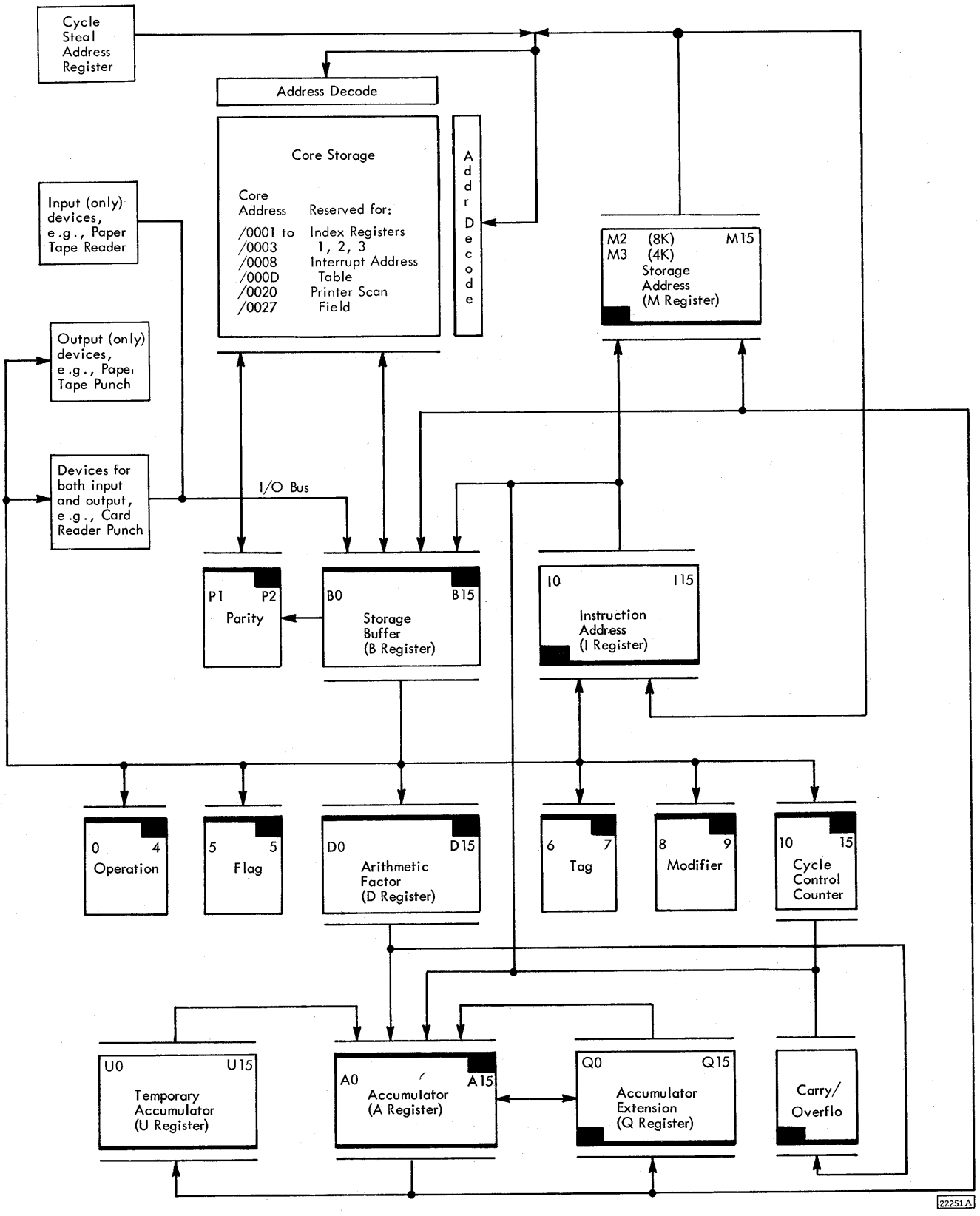



Figure 1-4. 1130 System Data Flow

- An indexing cycle (IX cycle) may be required to calculate the effective address.

- An indirect addressing cycle (IA cycle) may be required to obtain the direct address of data.

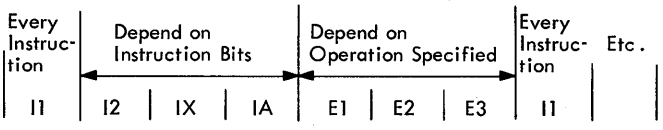
- The operation to be performed determines which, if any, of the execute cycles (E1, E2, and E3) are required by an instruction.

The instructions of a program are recognized and executed during a succession of machine cycles (Figure 1-5). Timings within each cycle are provided by an electronic machine clock, called the T-clock. During the early portion of a T-clock cycle, the addressed core storage location is read and the data set into the B-register. From there the word can be set into other registers. Later in the cycle, the word in the B-register is written into the same core storage location. Between read and write times, the word in the B-register can be changed.

Instruction Cycles

In execution of a program, the first cycle taken is an instruction cycle (I1 cycle), during which the instruction is read from core storage, set into registers for decoding, and written back into core storage, unchanged. An I1 cycle occurs for each instruction in the program. A second instruction cycle (I2 cycle) occurs when the instruction occupies two core storage locations. (A two-word instruction is called "long format".)

When execution of an instruction requires data from core storage, the address of that data is obtained during the I2 cycle or is computed during the instruction cycle or cycles.




I1 Occurs for every instruction.
I2 Depends on instruction format bit (bit 5 = 1).
IX Occurs when index register is used to compute effective address (bit 6 or 7 = 1).
IA Occurs when computed address is an indirect address (bits 5 and 8 = 1).
E1, E2, E3 Occur as required for operation specified.

13198

Figure 1-5. Cycle Sequence

Indexing

The address of data to be processed (called the effective address or EA) may not be fully computed by the end of the I1 or I2 cycle. Addition of the value stored in an index register (XR) to the EA thus far computed may be required. The index registers are actually the three core storage locations /0001, /0002, and /0003. When addition of an index register is required for the computation of an effective address, an indexing (IX) cycle follows I-cycles. Bits in the instruction itself indicate when indexing is required and the index register to be used.

Indirect Addressing

The address computed during I-cycles and the indexing cycle, if one occurs, can be an indirect address. In other words, the computed address can be the core storage location at which the address of the data is stored. An indirect addressing (IA) cycle is required to read the word at the indirect address. That word is the direct address of the data to be processed in execution of the instruction. A bit in the instruction word (first word) of a long-format instruction indicates when indirect addressing is required.

Execution Cycles

After all cycles required to establish the operation code and the effective address (direct) of the data have been completed, execution of the instruction starts. The execution cycles for any instruction depend on the operation to be performed in executing the instruction. There are three types of execution cycles, designated E1, E2, and E3. A few instructions require no E-cycles, some require only an E1 cycle, and some require an E1 cycle and one or more E2 cycles. The E3 cycle is required only when the instruction calls for transfer of data to or from an I/O device. When execution of any instruction is complete, the CPU takes another I1 cycle to obtain the next instruction from core storage. Remember that the next instruction may not be in the next position of storage following the one just executed. Execution of the program continues in this manner until stopped by an instruction to wait or by manual intervention.

Interrupt Principle

- The interrupt principle allows more than one I/O device to be operated at one time.

- Interrupts are requested to allow transfers of data or device status.
- Interrupts eliminate the loss of processing time which would otherwise occur while waiting for data transfers.
- The CPU program must contain subroutines to service interrupts by providing data to an I/O device or accepting data or status information from an I/O device.
- Interrupts cause automatic branches to the servicing subroutines.
- The device status word (DSW) identifies the cause of each interrupt.
- Every device is assigned to one or more interrupt levels.
- More than one device can be assigned to an interrupt level.
- The interrupt level status word (ILSW) identifies which device caused an interrupt when more than one device is assigned to an interrupt level.
- Interrupts are serviced according to their priority levels.
- High priority levels can interrupt while lower priority level interrupts are being serviced.

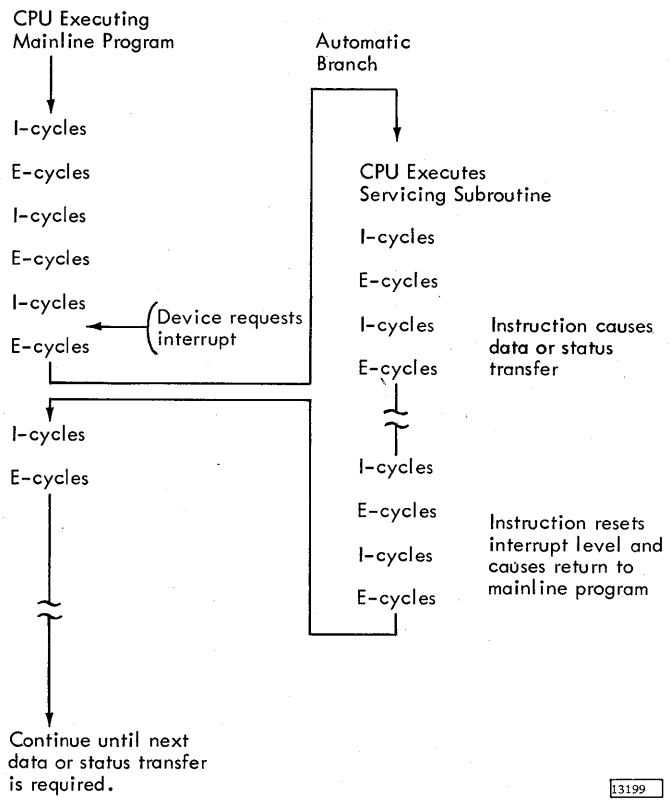



Figure 1-6. Interrupt Principle

## Interrupt Causes

An interrupt occurs when there is a need to transfer data to or from certain I/O devices or a need to transfer status information from a device to the CPU. The interrupt principle allows the CPU to continue processing the mainline program between these transfers (Figure 1-6). Little CPU processing time need be lost waiting for the slower I/O devices. A 1442 card reader operation shows the advantages of the interrupt method.

Assume that the CPU is processing the data from a card and will soon need the data from the following card. Any time after all data is received from the first card, the program can contain an instruction to start the mechanisms necessary to feed and read the next card. Only a few machine cycles are required to set up circuits which start the mechanical motion. The CPU is then free to continue processing the data from the first card.

When the following card has fed into the position where a column has been read, the card reader signals the CPU that a column of data is available for storage. That is, the card reader requests the CPU to interrupt its processing long enough to store the data now available to the CPU. The interrupt cannot occur until the CPU completes execution of its present instruction. At that time, the interrupt (a read response interrupt) causes an automatic branch to a location in storage where a program subroutine starts. The instructions of the subroutine provide for storing the available data. (They service the interrupt.) When the data has been stored, the CPU resumes processing with the instruction it would have processed next had it not been interrupted. As card motion continues, the following columns are read, and each column generates a read response interrupt request to the CPU. Each request must be serviced before the next column is read or data is lost.

After the last column of the card has been read and the data transferred, this "operation complete" status information must be transferred to the CPU.

The card reader sends an additional interrupt request to the CPU. The interrupt servicing subroutine transfers the status information to the CPU. Analysis by the CPU program indicates that this interrupt request is a result of operation complete rather than an indication that data is available.

Device Status Word

Recognition of the cause of an interrupt request is made possible by the fact that a device status word (DSW) is always available to the CPU. A position of the DSW for a device can be set to a 1 to indicate a particular status or condition within that device. In the 1442 card reader example, as each read response interrupt is requested, a 1 is set into position 0 of the DSW. On the other hand the operation complete status sets a 1 in position 4 of the DSW to indicate operation complete. Program analysis of the DSW reveals the status or condition which caused an interrupt.

Interrupt Levels and Priority

The foregoing examples of the interrupt principle show that interrupt requests can be generated by different conditions within a device. In a system with many devices attached and in use, several interrupt requests can occur nearly simultaneously. Therefore, it is necessary to establish a sequence for servicing the requests. Still using a card reader as an example, once a card starts through the reading station, interrupt requests occur at a fixed, quite rapid rate. The time for servicing the interrupt is limited; so this type of interrupt request is assigned a high priority, or interrupt level 0. The operation complete interrupt request does not require such immediate servicing and is assigned to the lower priority interrupt level 4. Different types of interrupt requests from the same device can be assigned to different interrupt levels. Also, several devices can be assigned to the same level.

Interrupt Level Status Word

When several I/O devices are assigned to an interrupt level and one of them activates an interrupt request, the CPU program must have a means of identifying which device caused the interrupt request. Each device which uses interrupt level 4, for instance, has a position assigned to it in the interrupt level status word (ILSW) for level 4. When a device activates an interrupt request, the device also sets a 1 in its assigned position of the ILSW. Analysis of the ILSW by the program provides the required identification of the device which caused the interrupt.

In review, an I/O device can request to interrupt the CPU program execution for any of a number of reasons. The CPU services the highest priority level interrupt request presently active by automatically branching to a program subroutine for that level. The subroutine program (Figure 1-7) analyzes the ILSW to determine which device caused the interrupt and which portion of the subroutine to execute. The condition within the device which caused the interrupt request is determined by analyzing the DSW. This analysis determines the exact small portion of the interrupt subroutine which must be executed to service that interrupt request. If no other interrupt requests are active when subroutine execution is completed, control returns to the mainline program at the point at which it was interrupted. If other requests are active, they are serviced in the order of their priority (levels 0, 1, 2, etc.) before the return to the mainline program.

If the CPU is processing a lower interrupt level subroutine, and a higher level request occurs, the CPU branches immediately to the higher level subroutine (Figure 1-8). When the higher level subroutine is completed, the program automatically returns to the point where the lower level subroutine

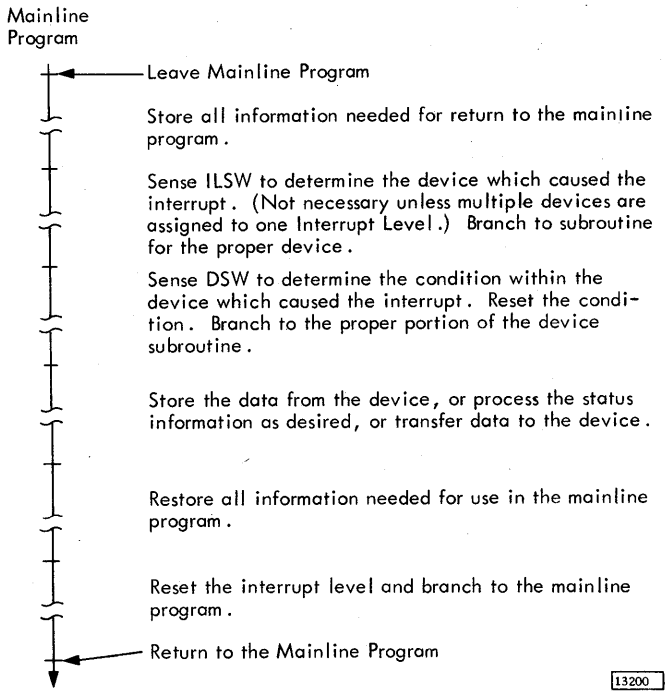



Figure 1-7. Essentials of Interrupt Servicing Subroutine

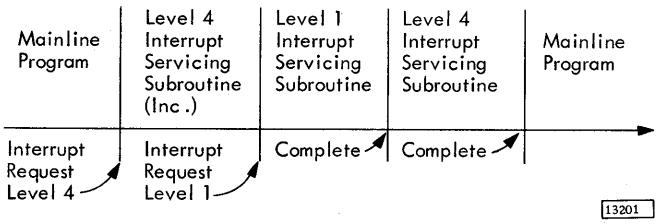



Figure 1-8. Interrupt Priorities

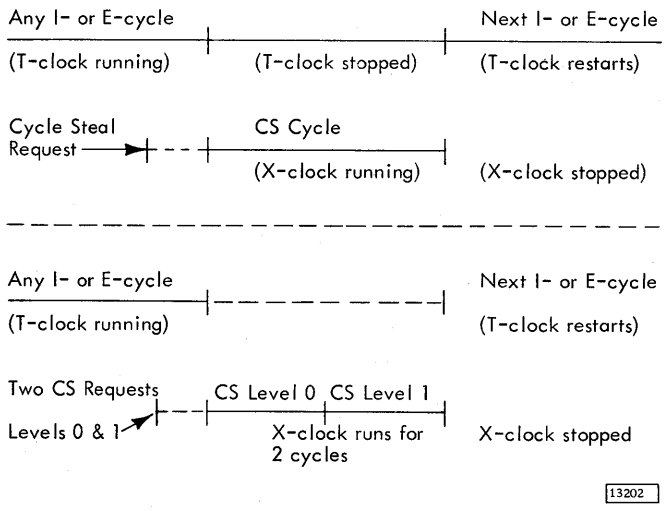



Figure 1-9. Cycle Stealing

was left. When the subroutine for the lowest active level is completed, control returns to the mainline program.

## Cycle Steal Principle

- I/O devices requiring higher data transfer rates than possible with the interrupt method use cycle steals.
- Data transfers occur without the need for program control.
- During cycle steal, basic machine T-clock stops and X-clock provides timings.
- Cycle steal can start at the end of any T-clock cycle.
- Cycle steal requests are assigned to priority levels.
- Cycle steal devices request interrupts only to transfer status information.

### Reason for Cycle Steal

Certain input devices may have successive words of data available for storage too rapidly to operate on the interrupt basis. This may also be true of an output device which needs data words from storage in rapid succession. Such a device can request a CPU storage cycle as soon as possible to store a word or to obtain a word from core storage. In other words, the device "steals a cycle" or cycle steals.

When a cycle steal occurs, the T-clock stops at the end of the machine cycle it is in when the request is made (Figure 1-9). A second clock (the X-clock) starts and provides timings for cycle steal. When the required cycle steal has been completed, the X-clock stops and the T-clock restarts, allowing the CPU to continue processing. In the cycle steal operation, the only register affected is the storage buffer register. Cycle steal may occur at the end of any T-clock cycle because the data in that register has already been used. For example, a cycle steal could occur between I1 and I2 of an instruction requiring two I-cycles. Note the difference from interrupts, which can occur only at the end of execution of an instruction.

Cycle stealing devices use interrupts only to transfer status information such as operation complete, busy, etc.

### Cycle Steal Priority

Among devices which use the cycle steal method, certain ones require data from storage, or make data available for storage, faster than other cycle steal devices. Therefore, for cycle stealing, a priority of cycle steal levels has been established similar to interrupt priority. Cycle steal (CS) level 0 has the highest priority, followed by CS levels 1, 2, and 3. When two or more devices request cycle steals at the same time, the machine clock remains stopped while successive X-clock cycles occur. When all requested cycle steals have been completed, the X-clock stops. The T-clock restarts, and the sequence of CPU I-cycles and E-cycles resumes.

## Program Execution Sequence

Usually, program execution consists of executing a series of instructions which are stored in progres-

sively higher addresses of core storage. The sequence can be altered, however, by interrupt-forced branches or by instructions of the program itself. Moreover, the sequence of CPU cycles required to execute a program can vary in many ways from the common I-cycle, E-cycle, I-cycle, E-cycle, etc. Figure 1-10 shows one of the infinite number of cycle sequences which could occur in execution of part of a program. The figure shows that a higher level interrupt can occur while a lower level interrupt is being serviced. Also, cycle steal can occur at the end of any type of cycle.

## MACHINE LANGUAGE

### Core Storage Word

- Each addressable core storage location stores 18 bits.
- Sixteen positions, 0 through 15, comprise a binary word for processing.
- Two positions maintain parity in core storage.

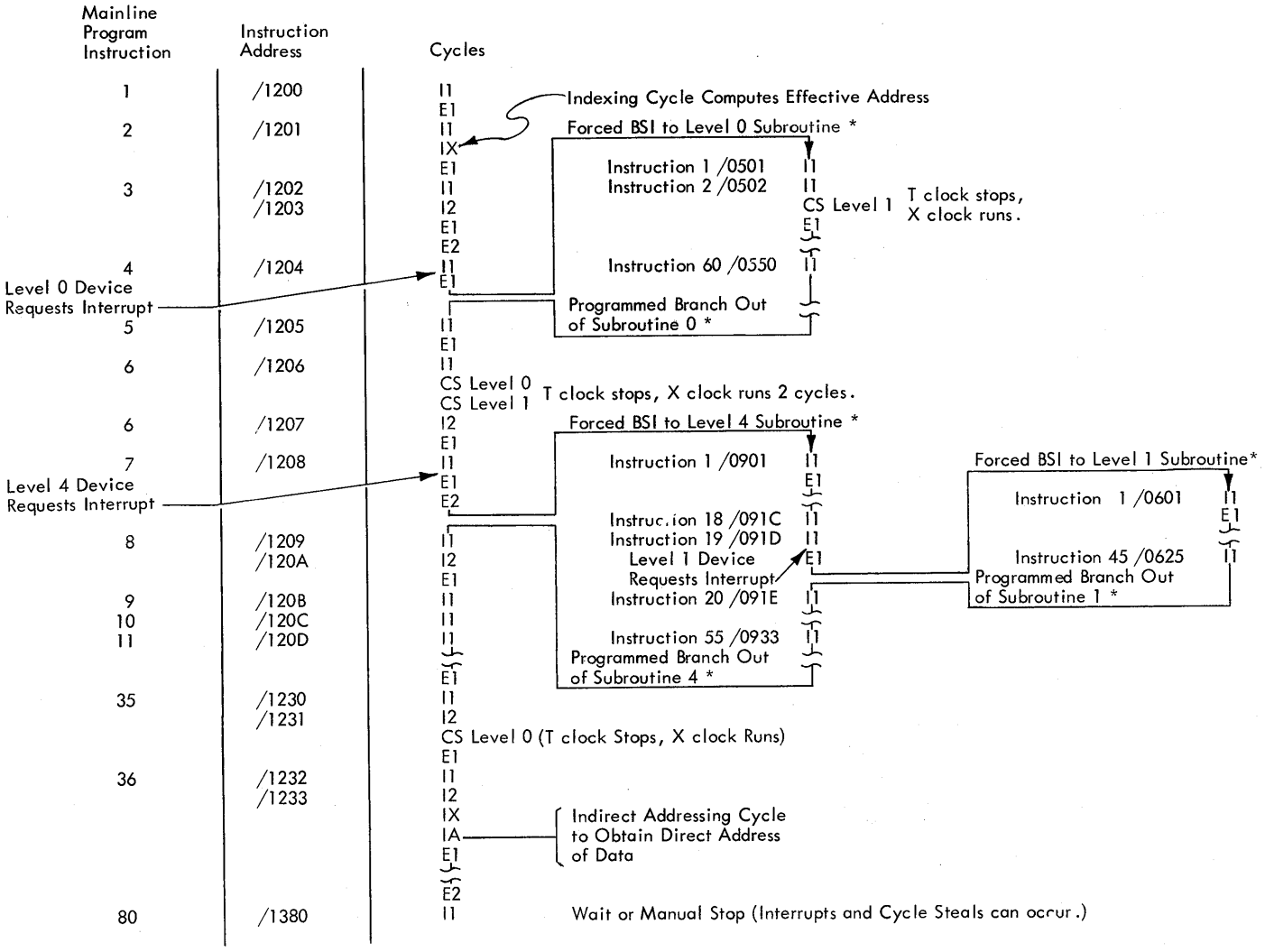



Figure 1-10. Typical Cycle Sequence

- Parity bit 1 maintains odd parity for positions 0 through 7, and parity bit 2 for positions 8 through 15.

Although each addressable core storage location stores 18 bits of information (either 0 or 1), only 16 are used in the basic word for processing. The other two bits are used for parity checking. Each half word within core storage must contain an odd number of 1's (Figure 1-11), or a parity error occurs. Parity bit 1 is set to a 1 when positions 0 through 7 contain an even number of 1's, and parity bit 2 is set to a 1 when positions 8 through 15 contain an even number of 1's.

The 16-bit word can contain coded information or numerical information in binary form. When the word contains numerical information, each binary 1 has a decimal value which is dependent on the position of the 1 in the word (Figure 1-12). The decimal value of the whole word is the sum of the decimal values of all 1's in the word. Thus, the decimal value of the word shown in Figure 1-11 is 787.

Types of coded information having no numerical significance include character codes for I/O operations and program instructions.

Instruction Format

- An instruction can contain one or two words.

- The positions of a one-word instruction are: 0-4, operation code; 5, format (always 0); 6-7, tag; and 8-15, displacement.

- The positions of the first word of a two-word instruction are: 0-4, operation code; 5, format (always 1); 6-7, tag; 8, indirect addressing; and 9-15, modifiers. The second word is an address.

An instruction can have either 16 or 32 positions, requiring either one or two addressable locations for storage. The format of the single-word instruction and the first word (lower address location) of the two-word instruction are similar (Figure 1-13).

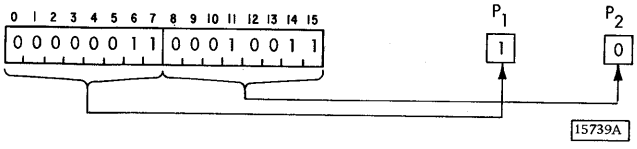



Figure 1-11. Bit Positions and Parity

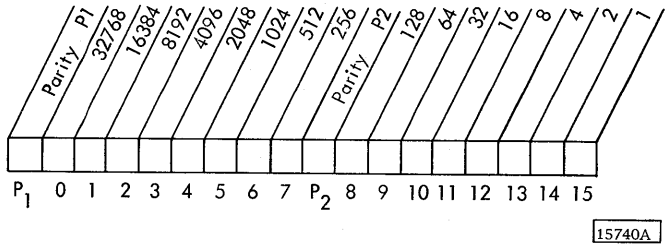



Figure 1-12. Decimal Values

Bit positions 0-4 contain the operation code (Figure 1-14), which specifies the operation to be performed when the instruction is executed. Bit position 5 specifies whether an instruction is one-word or two-word and is set to a 1 when the instruction is a two-word instruction. Bit positions 6 and 7 contain the tag which specifies the index register to use in indexed or indexing instructions. The coding of bit positions 6 and 7 are:

00 No index register.
01 Index register 1.
10 Index register 2.
11 Index register 3.

Bit positions 8 through 15 of a one-word instruction may contain a displacement for use in computing the effective address (EA) of data to be processed by the instruction. Any other use of these bit positions is described in "Chapter 3, Principles of Operation" with the operation which makes such special use of them.

Bit position 8 of the first word in a two-word instruction is set to a 1 when indirect addressing is required. Bit positions 9 through 15 may be set with modifiers which further define the operation to be performed. The second word of a two-word instruction contains an address to be used in the execution of the instruction. The address may be the indirect

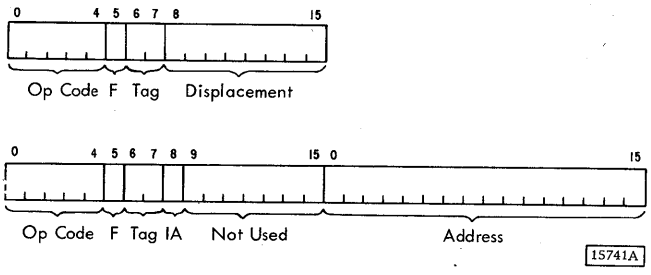



Figure 1-13. Instruction Format

| Instruction | Mnemonic | Binary OP Code |
|---|---|---|
| Load and Store | | |
| Load ACC | LD | 11000 |
| Load Double | LDD | 11001 |
| Store ACC | STO | 11010 |
| Store Double | STD | 11011 |
| Load Index | LDX | 01100 |
| Store Index | STX | 01101 |
| Load Status* | LDS | 00100 |
| Store Status | STS | 00101 |
| Arithmetic | | |
| Add | A | 10000 |
| Add Double | AD | 10001 |
| Subtract | S | 10010 |
| Subtract Double | SD | 10011 |
| Multiply | M | 10100 |
| Divide | D | 10101 |
| And | AND | 11100 |
| Or | OR | 11101 |
| Exclusive Or | EOR | 11110 |
| Shift Left* Modifier Bits 8 & 9: | | |
| Shift Left ACC 00 | SLA | 00010 |
| Shift Left ACC and EXT 10 | SLT | 00010 |
| Shift Left and Count ACC 01 | SLCA | 00010 |
| Shift Left and Count ACC and EXT 11 | SLC | 00010 |
| Shift Right* Modifier Bits 8 & 9: | | |
| Shift Right ACC 00 or 01 | SRA | 00011 |
| Shift Right ACC and EXT 10 | SRT | 00011 |
| Rotate Right 11 | RTE | 00011 |
| Branch or Skip | | |
| Branch and Store IAR | BSI | 01000 |
| Branch or Skip on Condition | BSC | 01001 |
| Modify Index and Skip | MDX | 01110 |
| Wait* | WAIT | 00110** |
| Input/Output | | |
| Execute I/O | XIO | 00001 |

* Valid in short format only
** All unassigned OP codes are defined as wait operations

13204

Figure 1-14. 1130 Instruction Set

address of the direct address to be used, when indirect addressing is called for.

Figure 1-14 shows that a few operation codes can be used in the one-word (short) instruction format only.

## Data Format

- A negative number is processed in the 2's complement form. There is always a 1 in the high-order position of the word (or first word of a double word).
- Two words can be processed as a double word by certain instructions.
- A double word can store a positive binary number with a decimal equivalent of 2,147,483,647, or a negative binary number which is 1 greater.
- The program usually sets up a data table for I/O operations.
- The format of a data table depends on the device using it.

A negative binary number, when stored in 1130 core storage, is indicated by a 1 present in the bit 0 position, and positions 1 through 15 contain the 2's complement of the number. The 2's complement of any number is its 1's complement plus 1. Thus, the binary number 1000 0000 0000 0000, with a decimal value of -32,768, is the greatest negative number which can be stored in a single word. The binary number 0111 1111 1111 1111, with a decimal value of 32,767, is the greatest positive number which can be stored in a single word.

In order to increase the size of the number which can be processed, the 1131 instruction set has instructions which allow combining of two core storage words. The first word of the resulting double word must be stored in an even address location of storage. Bit 0 position of the first word of a double word contains the sign, and the binary number has 31 significant positions. The range of decimal values which can be stored in a double word is from +2,147,483,647 to -2,147,483,648.

Data for use by an output device must be coded and stored by the program, usually in a data table. A data table (Figure 1-15) usually consists of a number of consecutive core storage locations, with the lowest address location storing the first data to be transferred to or from the device. The format of the data table and its size are largely dependent on the type of the device. For some devices, the first word of the data table contains a word count, which is the number of data words in the data table. Other

| | | |
|---|---|---|
| /1500 | 1st Data Word | Word Count |
| /1501 | 2nd Data Word | 1st Data Word |
| /1502 | 3rd Data Word | 2nd Data Word |
| /1503 | 4th Data Word | 3rd Data Word |
| /154E | 79th Data Word | 78th Data Word |
| /154F | Last Data Word | 79th Data Word |
| /1550 | Not Used | Last Data Word |

13205

Figure 1-15. Two Forms of Data Tables - 80 Positions

devices do not require the word count word, because the number of words required is fixed.

Data received from an input device is also usually stored by the program in a data table, with the first word received placed in the lowest address location. For some input devices, the first word of the data table must contain a word count, showing how many data words are to be accepted. Each data word received is placed in the next higher address location.

## INSTRUCTION SET AND OPERATIONS

The operation codes of the 1130 instruction set (Figure 1-14) can be grouped into six major functions: load and store, arithmetic, shift left, shift right, branch or skip, and input/output. Note that certain instructions are valid in the short (one-word) format only. None of the short format instructions allow indirect addressing.

### Load and Store

#### Load

Load instructions take data from core storage and set it into other circuit components (registers or status indicators) without changing the data in the core storage location. 'Load accumulator' or 'load index' sets the accumulator or an index register, respectively, to the value that is in the addressed core storage location. 'Load status' sets the carry and overflow indicators according to positions 14 and 15 of the addressed core storage location. 'Load double' sets the accumulator and its extension to the value stored in the addressed double word of core storage. The accumulator extension contains the low-order positions. In no case is core storage changed by a load instruction.

#### Store

Store instructions set data into core storage. 'Store accumulator' or 'store index' replaces the value in the addressed core storage location with the value in the accumulator or selected index register. 'Store index' with no index register selected (tag=00) stores the value presently in the instruction address register (IAR).

'Store double' sets the values of the accumulator and its extension into a double word of core storage. The values in the accumulator, its extension, the index register, and the IAR are not changed by these instructions.

'Store status' sets a 1 in the addressed core storage word, position 14, if the carry indicator is on, and position 15 if the overflow indicator is on. Positions 0-7 of the addressed word are not changed, but positions 8-15 are reset to 0 before storing the status. The indicators are turned off by storing.

### Arithmetic

The CPU performs arithmetic operations by an add-to-accumulator method. Therefore, just before any arithmetic instruction, the program usually contains a 'load accumulator' or 'load double' instruction to set one factor into the accumulator or accumulator and its extension. At the end of an arithmetic operation, the result is in the accumulator.

#### Add or Subtract

An 'add' or 'subtract' instruction (Figure 1-16) obtains the second factor from the addressed core storage location and sets it into the arithmetic factor register (D-register). Addition to or subtraction from the accumulator proceeds, and the result of the operation stays in the accumulator. The second factor is unchanged in core storage but is destroyed in the D-register.

#### Add Double or Subtract Double

In performing 'add double' or 'subtract double' (Figure 1-17), a double word is obtained from core storage at the effective address (EA) and EA +1 locations.

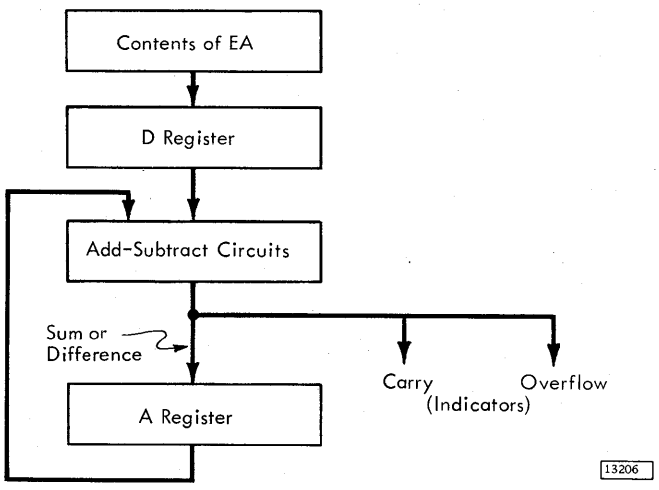



Figure 1-16. Add or Subtract Data Flow

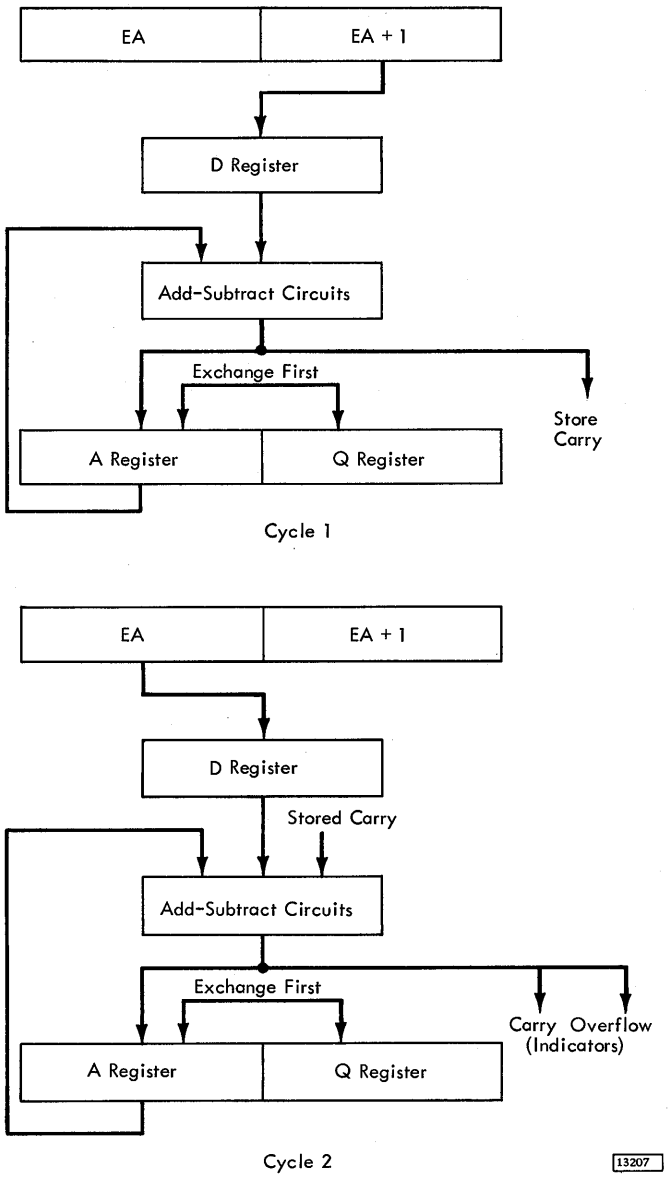



Figure 1-17. Add Double or Subtract Double Data Flow

One double-word factor must already be present in the accumulator (A-register) and its extension (Q-register), with the low-order position in the extension. Only one word from each factor can be added or subtracted during a single cycle, and the low-order words must be processed first.

Early in the first cycle, the values in the A- and Q-registers are exchanged. The word from EA +1 is set into the D-register, and the required add or subtract occurs. During the second cycle, the A- and Q-registers are again exchanged, and the word from EA is set into the D-register. The high order words of each factor are added or subtracted, and the complete double word result remains in the A- and Q-registers. Any carry required from the high order position of the Q-register to the low-order position of the A-register is sensed during addition or subtraction of the low-order words. The circuits then provide this carry when the high-order words are processed.

## Multiply

The multiply method used requires that the multiplier be set into the accumulator before the multiply instruction is given. Early in the multiply operation (Figure 1-18) the multiplier is transferred to the Q-register by exchanging the values in A- and Q-registers, and the A-register is then cleared. Then, as multiplication proceeds, the multiplicand is read from storage and added to or subtracted from the A-register, and the partial product is shifted right in the A-register. As shifting occurs, the multiplier shifts out of the Q-register, and the low-order positions of the product shift from the A-register into the high-order positions of the Q-register.

The combination of 1's or 0's present in Q-register positions 14 and 15 determine whether the multiplicand is added or subtracted or the partial product is merely shifted. When multiplication is complete, a double word product remains in the A- and Q-registers, with low-order positions in the Q-register.

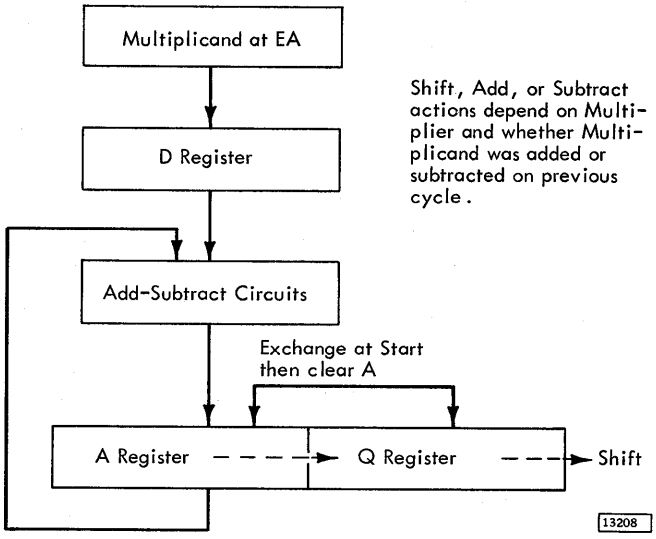



Figure 1-18. Multiply Data Flow

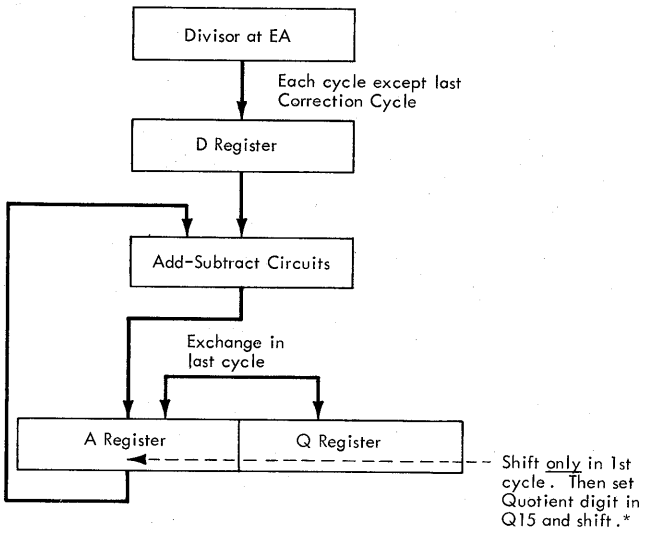
## Divide

Before the program gives a 'divide' instruction, the double-word dividend must have been placed in the A- and Q-registers. Division (Figure 1-19) requires 18 machine cycles, consisting of 16 reduction cycles, and a cycle each for correcting the quotient and remainder, if correction is necessary.

During each reduction cycle, the A- and Q-registers are automatically shifted left one position. Also, the divisor, which is read from core storage, is added to or subtracted from the A-register. Comparison of the signs of the divisor and the remainder in each cycle determines whether to add or subtract. The same comparison determines which quotient digit, 0 or 1, is set into Q-register position 15.

After the 16 reduction cycles the quotient is in the Q-register and the remainder is in the A-register. During cycle 17, both quotient and remainder are tested for the need of correction, and any required remainder correction is made. During cycle 18, the contents of the A- and Q-registers are exchanged, and any required quotient correction is made.

At the end of division, the quotient (in the A-register) has a sign determined algebraically by the signs of divisor and dividend. The remainder (in the Q-register) carries the sign of the dividend.




* Quotient digit and whether to add or subtract determined by comparison of signs (divisor and remainder).

Figure 1-19. Divide Data Flow

## AND, OR, and Exclusive OR

The logical instructions are grouped with arithmetic instructions. They also require that one of the words to be processed already be present in the accumulator. Any of the logical instructions first reads the addressed core storage word and sets it into the D-register. Each position of the D-register is then compared with the corresponding position of the accumulator. The result, which is set into the accumulator, depends on which instruction is being executed.

AND: When the position in the accumulator and the corresponding position of the D-register both contain 1's, the 1 is left in the accumulator. Any other condition leaves a 0 in that accumulator position.

OR: When either the accumulator position or the corresponding D-register position contains a 1, a 1 is set into that accumulator position.

Exclusive OR: When either the accumulator position or the corresponding D-register position contains a 1, and the other does not, a 1 is set in the accumulator. Any other condition leaves a 0 in that accumulator position.

## Shift Left

The 'shift left' operation code performs four different shift left operations, with modifier bit positions 8 and 9 further defining the operation to be performed.

Modifier 00 causes all bits in the accumulator to be shifted left a specified number of positions. Modifier 10 causes a similar shift of all bits in both the accumulator and its extension. A 1 shifted out of the high-order position of the accumulator turns on the carry indicator, and a 0 turns it off. The number of positions shifted is determined by the displacement (tag = 00) or an index register (tag $\neq$ 00; the tag selects the index register).

Modifiers 01 and 11, with the tag = 00, cause basic shift left operations like modifiers 00 and 10. However, when the tag $\neq$ 00, shift left and count operations occur, with modifier 01 affecting the accumulator only, and modifier 11 affecting both the accumulator and its extension. At the end of a shift left and count operation, the selected index register contains the shift count that remains when shifting is stopped.

Shifting is stopped in a shift left and count operation when either of two conditions occurs:

1. The bits have been shifted the number of positions indicated in the selected index register

before any 1 has been shifted into the high-order position of the accumulator. In this case, the carry indicator is turned off.

2. A 1 is shifted into the high order position of the accumulator before the bits have been shifted the number of positions indicated in the selected index register. The number has been decreased by 1 for each position shifted, and the remainder is set back into the index register. The carry indicator is turned on.

Should both conditions described occur simultaneously, the carry indicator is turned off.

Shift Right

A 'shift right' instruction causes the bits in the accumulator or accumulator and its extension to be shifted right the number of positions indicated in the displacement (tag = 00), or the selected index register (tag ≠ 00; the tag selects the index register). With modifiers 00 or 01, the instruction shifts only the accumulator. Modifier 10 specifies 'shift right accumulator and extension,' and bits from the low order position 15 of the accumulator are shifted into high-order position 0 of the extension.

Modifier 11 causes the operation to become a 'rotate right,' affecting both accumulator and its extension. Instead of being lost as in other shift right operations, 1's which are shifted out of position 15 of the extension are set into position 0 of the accumulator. Bits continue to shift in a loop-around manner until they have shifted the number of positions indicated in the displacement or the selected index register.

Branch or Skip

When a branch occurs, the CPU obtains its next instruction from a core storage location other than the next one following the location of the 'branch' instruction. Certain instructions to branch are unconditional and cause a branch every time they are encountered in the program. Other branch instructions are conditional, and the branch occurs only when certain conditions within the CPU are met. Bit positions 10-15 of the instruction word are set to 1's to indicate which conditions are to be tested, as follows:

| Bit Position | Condition |
|---|---|
| 15 | Overflow indicator off |
| 14 | Carry indicator off |
| 13 | Accumulator contents even |
| 12 | Accumulator positive and not zero |
| 11 | Accumulator negative |
| 10 | Accumulator zero |

Branch and Store IAR

The 'branch and store IAR' (BSI) instruction may be either short or long format, and the operations differ.

Short Format (F = 0): This instruction first stores the contents of the instruction address register (which is the address of the next sequential instruction) at the EA generated for the instruction. Then EA + 1 is set into the IAR and an unconditional branch to EA + 1 occurs. For example, assume the BSI instruction is at address /0500 and the EA is /0600. The IAR contains /0501, which is stored at /0600 by the instruction, and an unconditional branch to /0601 occurs.

Long Format (F = 1): In long format, the BSI instruction calls for a conditional branch. If any of the conditions being tested is true, no branch occurs. If none of the tested conditions is true, the contents of IAR are stored at the EA, and a branch to EA + 1 occurs.

The branch forced by an interrupt is an unconditional long-format BSI using indirect addressing. The forced indirect address is an address in the interrupt address table, /0008 to /000D inclusive. These addresses are reserved for interrupt levels 0 through 5, respectively. For example (Figure 1-20), assume the address in location /0008, which is reserved for interrupt level 0, is /0900. Also assume the servicing subroutine for interrupt level 0 has its first instruction in /0901. An interrupt level 0 request occurs when the IAR contains 00451 (the address of the instruction that would have been executed next had the interrupt not occurred). The forced BSI indirect at this time sets the return address (/0451) into the EA, which is /0900. An unconditional branch occurs to /0901, the first instruction in the interrupt level 0 servicing subroutine.

Branch or Skip on Condition

The 'branch or skip on condition' (BSC) instruction may be either short or long format, and the resulting operations differ. Figure 1-21 shows examples of BSC operations.

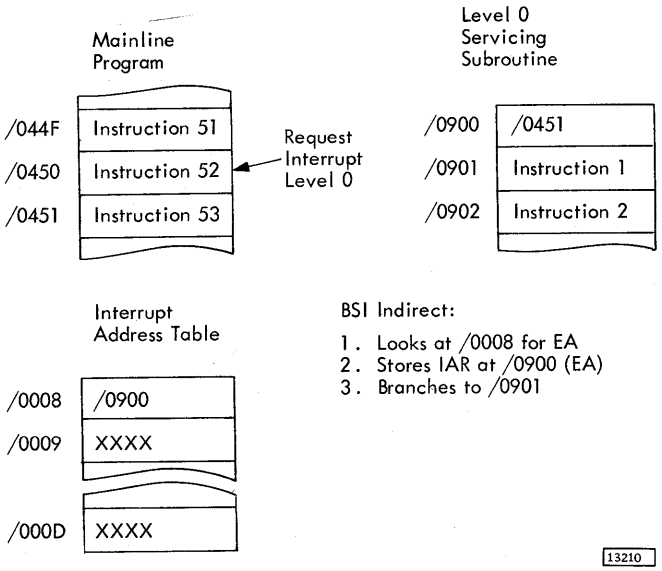



Figure 1-20. Interrupt Forced BSI

Short Format (F = 0): The short-format BSC instruction causes the program to use the instruction from the next higher core address or to skip one address and use the second higher address. The skip is conditional and occurs if any condition tested by the instruction is true. If no tested condition is true, no skip occurs. The instruction following a short-format BSC instruction must also be short format. Otherwise, any skip would be to the address word of a two-word instruction.

Long-Format (F = 1): The long-format BSC instruction causes a branch to the EA when none of the tested conditions is true. If the instruction does not specify any conditions to be tested, the program branches unconditionally to the EA.

Programming Note: A BSC with an indirect address is often used to cause a return from a subroutine which was entered by a 'branch and store IAR' instruction. Remember that the 'branch and store IAR' set a mainline program return address at its EA. The indirect address of the BSC instruction is

| Instruction Word | | | | | | | | Skip Conditions | | | | | | Result |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| F | | Displacement | | | | | | Accumulator Status | | | | Indicators | | |
| 5 | - | 10 | 11 | 12 | 13 | 14 | 15 | Zero | Minus | Plus | Even | Carry | O'flow | |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | | - | No | - | **Yes** | - | - | Skip |
| 0 | 1 | 0 | 1 | 0 | 0 | 1 | | No | - | No | - | - | On | No Skip |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | | - | No | - | - | **Off** | - | Skip |
| 0 | 1 | 0 | 0 | 1 | 0 | 1 | | No | - | - | **Yes** | - | **Off** | Skip |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | | **Yes** | No | No | No | On | On | Skip |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | | - | - | - | - | - | - | No Skip |
| 1 | 1 | 0 | 1 | 1 | 0 | 1 | | No | - | No | **Yes** | - | On | No Branch |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | | - | - | - | - | - | - | Branch |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | | No | **Yes** | No | **Yes** | On | On | No Branch |
| 1 | 1 | 1 | 0 | 1 | 0 | 1 | | No | No | - | No | - | **Off** | No Branch |
| 1 | 0 | 1 | 0 | 1 | 0 | 0 | | - | No | - | No | - | - | Branch |

☐ Indicates condition causing skip or preventing branch.

240408

Figure 1-21. BSC Examples

set the same as the EA of the BSI instruction. Then, if the BSC causes a branch, the mainline program is reentered at the return address. In the example of Figure 1-19, the indirect address of the BSC is set to /0900. Thus, if the branch occurs, control returns to the mainline program at /0451.

An additional function can be performed by setting bit 9 = 1 in the instruction word when the BSC is used to return from an interrupt servicing subroutine. The presence of a 1 in bit 9 position makes the instruction a 'branch out or skip on condition' (BOSC). Branching as a result of the BOSC shows servicing of that level of interrupt is complete, and the level is reset. Servicing of other interrupt requests on the same or lower priority levels can proceed. If none is active, control returns to the mainline program. BOSC is a conditional branch, and the interrupt level is reset only when the branch occurs. The BOSC instruction should not be used to return from subroutines other than interrupt servicing subroutines. Use a BSC with bit 9 = 0 for conditional branches.

Modify Index and Skip

The 'modify index and skip' instruction can be either short format or long format. Figure 1-22 shows the results of execution of the different formats of the instruction. The skip consists of skipping a single core storage location to obtain the next instruction. When there is a possibility of a skip, the instruction following the 'modify index' must be of the short format. Otherwise the skip is to a location containing an address rather than an instruction.

Short Format (F = 0): The short-format 'modify index' instruction causes the value in the displacement positions to add to the IAR (tag = 00) or an index register (tag ≠ 00; the tag determines which index register). The displacement may be negative, indicated by a 1 in bit position 8. Before the addition takes place, the circuits set the binary digit (0 or 1) in position 8 into positions 0 through 7, to create a 16-position displacement.

When the tag = 00 and the displacement is added to IAR, the instruction causes an unconditional branch to the new address in IAR. When the displacement is 0, the instruction acts as a 'no operation' code, and when the displacement is 1, a skip is effected. A displacement of -1 sets the IAR back to the address of the 'modify index' instruction, which is repeated over and over. Any other possible displacement value causes either a branch ahead or branch back.

| Format | Tag | IA | Add | Skip |
|---|---|---|---|---|
| 0 | 00 | - | Displ. to IAR | Unconditional |
| 0 | 01 | | Displ. to XR-1 | Conditional* |
| 0 | 10 | - | Displ. to XR-2 | |
| 0 | 11 | - | Displ. to XR-3 | |
| 1 | 00 | - | Displ. to Contents of core storage (specified by Address word). | |
| 1 | 01 | 0 | Address word to XR-1 | |
| 1 | 10 | 0 | Address word to XR-2 | |
| 1 | 11 | 0 | Address word to XR-3 | |
| 1 | 01 | 1 | Contents of core storage specified by Addr. word - C (Addr) to XR-1 | |
| 1 | 10 | 1 | C (Addr.) to XR-2 | |
| 1 | 11 | 1 | C (Addr.) to XR-3 | |

*Skip one core storage location following 'MDX' if modified factor reaches zero or changes sign while being modified.

22264-A

Figure 1-22. 'MDX' Functions

When the tag ≠ 00, the displacement is added to the value in the selected index register (XR) and the new value is set back into the index register. A skip occurs only when the value in the register reaches 0 or changes sign as it is modified.

Long Format (F = 1): The functions of this instruction depend on the tag and whether or not indirect addressing is called for.

When the tag = 00, the instruction causes the displacement to be added to the contents of the addressed core storage location. The instruction also sets the sum back into the same core storage location. In this case, position 8 of the instruction word is the sign of the displacement and does not call for indirect addressing. A skip occurs only when the value of the core storage word reaches 0 or changes sign during modification.

When the tag ≠ 00 and the instruction is long format, indirect addressing may or may not be used. In either case, a value is added to the value in a selected index register, and the sum is set back

into the register. A skip occurs only if the result is 0 or the sign changes during modification. Without indirect addressing, the value that is added to the index register is the address word itself. With indirect addressing, the value that is added is the content of the core storage location addressed by the instruction.

Wait

Although the operation code 00110 is shown for a 'wait' instruction, note that any code not assigned to another operation causes a 'wait.' Execution of the instruction stops the CPU T-clock, which can be restarted manually or by an interrupt forced branch. Manual restart causes resumption of program execution with the next sequential instruction. An interrupt causes a branch to the servicing subroutine for that level of interrupt. Cycle steals can occur without affecting the 'wait' condition because cycle steals use the X-clock instead of the T-clock.

Input/Output

- There is only one I/O instruction: 'execute I/O' (XIO).
- XIO instruction reads a two-word input/output control command (IOCC) from the effective address (EA) and EA + 1.
- The IOCC consists of an address word and a control word that contains the area, function, and modifier codes.
- I/O devices operate on a direct program control basis or on a cycle steal basis.
- Direct program control devices request interrupts for either data or status transfers. All operations are program controlled including data transfers.
- Cycle steal devices request interrupts only for transfer of status information. Data transfers are accomplished by cycle stealing, but all other operations are program controlled.

'Execute I/O' (XIO) is the one instruction for controlling all I/O operations. It may be either a one- or two-word instruction. The XIO instruction causes reading of a two-word input/output control command (IOCC) from the positions of core storage selected by the EA and EA + 1. The EA must be even. During execution of the instruction, the word at EA + 1, which is the control word of the IOCC, is read first (on the E1 cycle). The IOCC control word (Figure 1-23) contains area, function, and modifier codes, which are set into the U-register for decoding. The area code selects the I/O device which is to perform the operation specified by the function code. In some cases the modifier code further defines the operation. The function codes are:

| | |
|---|---|
| 001 | 'Write' |
| 010 | 'Read' |
| 011 | 'Sense interrupt' |
| 100 | 'Control' |
| 101 | 'Initiate write' |
| 110 | 'Initiate read' |
| 111 | 'Sense device' |

During the following E2 cycle, the address word at EA is read from storage except when the function is 'sense interrupt' or 'sense device.' When the function code is 'sense interrupt,' the E2 cycle sets the interrupt level status word (ILSW) in the accumulator. When the function code is 'sense device,' the E2 cycle sets the device status word (DSW) in the accumulator. For all other function codes, the E2 cycle reads the address word of the IOCC from EA.

The function code and the area code (device) determine the use which is made of the address word. Detailed description of the usage is included with the description of each device. In general, the address word either further defines the function or provides an address used in the transfer of data. Data transfer (Figure 1-24) is either on the direct program control basis or the cycle steal basis, depending on the device. Regardless of the method of data transfer used, an I/O operation is initiated by an XIO instruction. This instruction sets up conditions within the device such that the device can request either an interrupt or a cycle steal to transfer data.

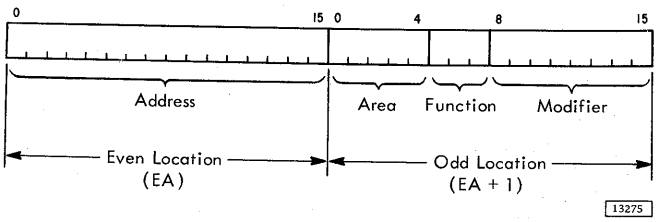



Figure 1-23. Input/Output Control Command (IOCC)

| Direct Program Control | |
|---|---|
| Instruction | Function |
| XIO CONTROL (in mainline program) | Activates circuits to start on I/O operation and allow the device to interrupt when data transfer is required. Interrupt causes forced branch to interrupt servicing subroutine. |
| XIO SENSE INTERRUPT (in subroutine) | Sets ILSW in accumulator. Program analyzes ILSW to determine which device on the active interrupt level caused the interrupt. |
| XIO SENSE DEVICE (in subroutine) | Sets DSW in accumulator. Program analyzes DSW to determine condition in device which caused interrupt (data transfer to or from device?). |
| XIO READ or XIO WRITE (in subroutine) | E3 cycle stores word in or obtains word from core storage location specified by the address word of the IOCC. Program updates address. |

| Cycle Steal | |
|---|---|
| Instruction | Function |
| XIO INITIATE READ or XIO INITIATE WRITE (in mainline program) | Activates circuits to start an I/O operation and allow the device to cycle steal when data transfer is required. Transfers the address word to the cycle steal address register.<br>CPU continues executing mainline program. Data is transferred one word per cycle steal until the required number of words have been transferred. Then the device requests an interrupt to transfer the status information. Machine circuits update the cycle steal address register. |

13211

Figure 1-24. Data Transfer Methods

### Direct Program Control

Devices which transfer data on the direct program control basis request an interrupt when data transfer is required. The interrupt servicing subroutine must contain an XIO instruction to cause the data transfer. A function code of 'read' in the IOCC causes transfer to the CPU, and a function code of 'write' causes transfer from the CPU. 'XIO read' and 'XIO write' are the only instructions which require an E3 cycle for execution.

### Cycle Steal

Devices which transfer data on a cycle steal basis request a cycle steal whenever the device has data for or requires data from the CPU. The only XIO instruction needed is the one which initiates the operation. The IOCC control word contains a function code of 'initiate read' or 'initiate write.'

## MULTI-INPUT FLIP-FLOP (FF)

This circuit is used in clock, counter, and register circuits and as a standard flip-flop. Much of the 1130 system logic design utilizes the characteristics of this flip-flop and its associated input circuits. A functional description of the flip-flop is in Appendix B.

## T-CLOCK AND TIMING

- The 1131 uses a 2.25-MHz (3.6-$\mu$s. cycle) or a 3.64-MHz (2.2-$\mu$s. cycle) free-running oscillator for the basic pulse generation.
- Figures 2-1 and 2-2 show the clock, clock advance circuits, and a timing chart of the clock output.

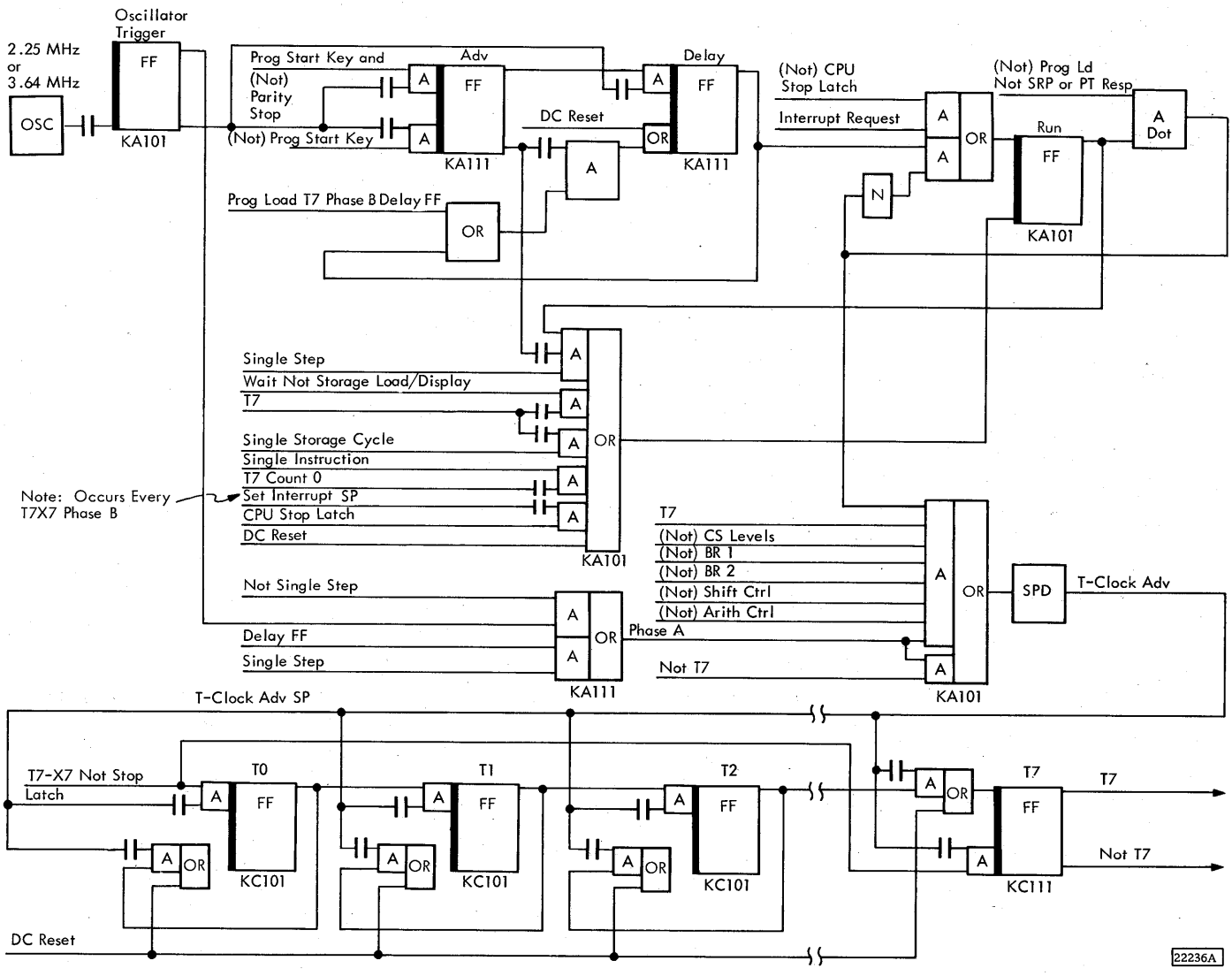



Figure 2-1. T-Clock Control

- 'Oscillator trigger' FF provides subclock-step pulses for clock advance generation and other system timing purposes. Flip-flop on is phase A, and off is phase B.
- 'Advance' FF is turned on by the first phase B after the start key is operated.
- 'Delay' FF is turned on by the 'advance' FF and the next phase B pulse.
- 'Run' FF is turned on by the 'delay' FF or by an interrupt request.
- Clock advance pulses occur at every phase A time after the 'run' FF is turned on.
- The clock ring consists of eight FF's, 'T0' through 'T7.'
- A basic machine cycle is one complete cycle of the clock ring.
- A cycle is extended for various operations by holding the 'T7' FF on.
- Reference maintenance diagram: AA211.

The clock is stopped with the 'T7' FF on and all other clock FF's off. 'DC reset' sets the clock to this position when power is turned on.

Starting the clock requires a 'T-clock advance SP' (sample pulse) to turn off 'T7' and turn on 'T0.' Figure 2-1 shows the conditions necessary to activate 'T-clock advance SP' when 'T7' is on. Once started, the clock is stepped to the next T7 by phase A pulses, if the single step switch is off. Each time 'T7' is on, all conditions for stepping from T7 to T0 must be active to keep the clock running.

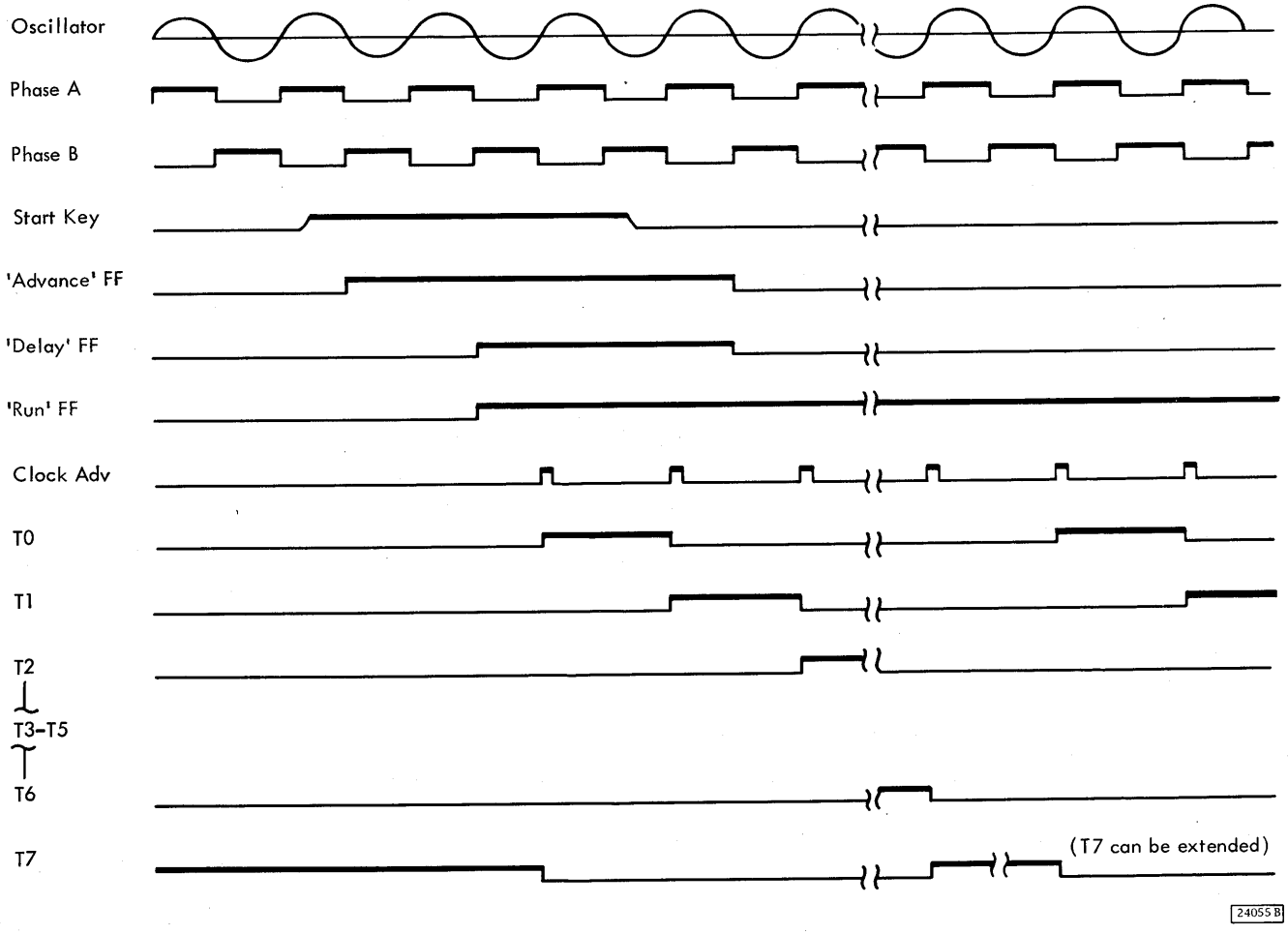



Figure 2-2. Clock Timing

When certain arithmetic, shift, and branch operations require more phase A and phase B pulses than one cycle provides, clock advance from T7 to T0 is prevented. The 'T7' FF stays on while the 'oscillator trigger' FF outputs provide timing pulses during the extended T7 time. When these operations end, the T-clock can step to T0 and continue running.

The clock can be stopped at T7 by turning off the 'run' FF in one of the following ways:

1. Pressing immediate stop key, thereby turning on CPU 'stop latch.'
2. Executing 'wait' instruction.
3. Operating in single storage cycle mode (every T7).
4. Reaching CCC count 0 condition when operating in single instruction mode.

The clock can also be advanced one step at a time (T7 to T0, T0 to T1, etc.) when the CPU is in single step mode. In this case, phase A pulses are not developed by the 'oscillator trigger' FF. Instead, each time the program start key is pressed, the 'delay' FF turning on activates 'phase A.' The clock advances one step each time the key is pressed.

## CYCLE TIMER

- Cycle timer is composed of seven flip-flops: I1, I2, IX, IA, E, E1, and E3 (Figures 2-3 and 2-4).
- Each flip-flop gates the circuits necessary for that type of cycle.
- E2 cycle is gated by 'E' FF and not 'E1', or 'E3.'
- Functions of the various cycles are described in the "Principles of Operation" chapter.
- Reference maintenance diagram: AA211.

The cycle timer selects the sequence of cycles for each instruction. The format of the instruction and the operation to be performed determine which cycle timer FF is turned on for any cycle.

I1 FF

- DC reset on by a power on reset or pressing the reset key with the CPU not running.
- Turned on after the last cycle of the previous instruction by the 'end op T0 SP' (pulse).
- Turned off at T0 if 'not end op T0 SP' line is activated.

I2 FF

- DC reset off.
- Turned on at T0 following I1 if the format bit (bit 5 of instruction read during I1) is a 1 and the instruction is not a shift instruction.
- Turned off at next T0.

IX FF

- DC reset off.

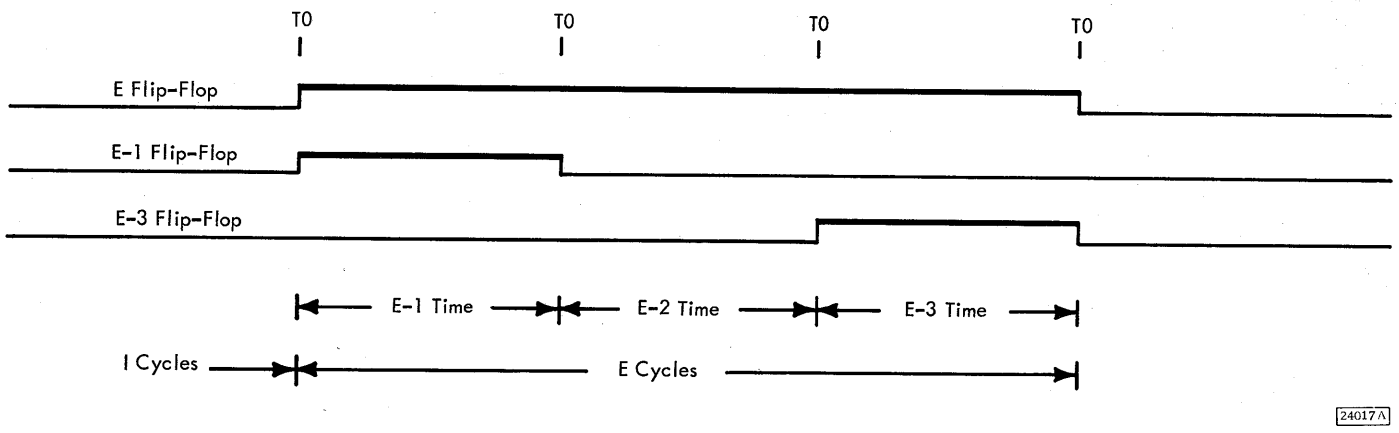



Figure 2-3. Development of E Cycle Times

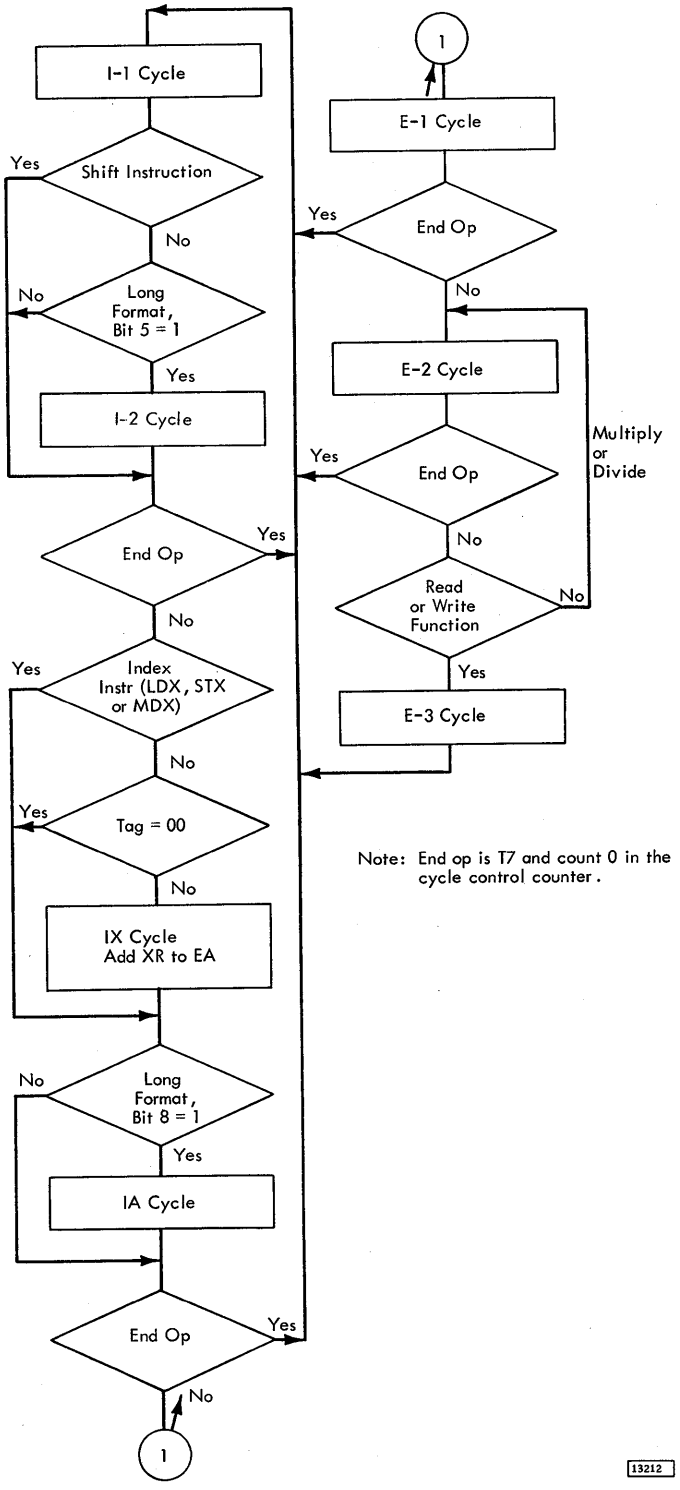



Figure 2-4. I and E Cycles

- Turned on at T0 following I1 or I2 if either tag bit (6 or 7) is a 1 and the instruction is not an index instruction ('load index,' 'store index' or 'modify index').
- Turned off at next T0.

IA FF

- DC reset off.
- Turned on at T0 of a two-word instruction (bit 5 = 1) following I2 or IX if the IA bit (bit 8) is a 1.
- Turned off at next T0.

E FF

- DC reset off.
- Turned on at T0 following any I-cycle if the conditions to end operation or to turn on the 'I2,' 'IX,' or 'IA' FF are not met.
- Remains on for E1, E2, and E3 cycles. Turned off by 'T0 end op' line.

E1 FF

- DC reset off.
- Turned on in the same manner as the 'E' flip-flop.
- Turned off at next T0.

E2 Time

- The 'E1' off and 'E3' off conditions are ANDed with the 'E' flip-flop on to provide E2 time.
- Turned off at next T0.

E3 FF

- Turned on at T0 following an E2 cycle, if the instruction is 'read' or 'write.'
- Turned off by 'T0 end op' line.

Cycle Control Counter

- Is a six-position binary counter.

- Can be set during I1 as required by the operation code.
- Can be set from the B-register for shift instructions.
- Can control either the number of cycles to be taken or the number of positions the accumulator is shifted.
- Can be decremented at T1 of each execute cycle when required by the operation code.
- Can be decremented by phase A pulses when shifting.
- Value in counter reaching 0 indicates the last cycle of an operation.

The cycle control counter (CCC) consists of six flip-flops: CCC-1, CCC-2, CCC-4, CCC-8, CCC-16, and CCC-32. It can be set to 1, 16, or 18 during I1 cycle, depending on the instruction, or it can be loaded from the B-register during a shift instruction. The 'CCC decrement sample,' which steps the counter value towards 0, may be either a phase A pulse when shifting, or a T1 pulse for other operations. Figure 2-5 shows the CCC counting E-cycles in a divide operation, being decremented each T1 time.

## REGISTERS

The registers of the CPU usually consist of a group of multi-input flip-flops which are used for temporary storage of bits. The cycle times at which any register is used and the function of the register depend on the instruction. The use of each register in each instruction is described in "Chapter 3, Principles of Operation."

### Storage Buffer Register (B)

- Storage buffer register has 16 positions.
- Words read out of core storage are set into the B-register.
- Odd parity is checked when a word is set in the B-register.
- Words to be written into core storage are first set into the B-register.
- Odd parity is generated for each half of the B-register when a word is written into core storage.
- Inputs to the B-register are from:

  Sense amplifiers (core storage).
  I/O input bus.
  I-register.
  Accumulator.

- Outputs from the B-register go to:

  Inhibit circuits for core storage.
  D-register.
  I/O output bus.
  I-register.
  Parity check and generator.
  Op register ($B_0$ - $B_4$).
  Format register ($B_5$).
  Tag register ($B_{6-7}$).
  Modifier register ($B_{8-9}$).
  Cycle control counter ($B_{10-15}$).

### Operation, Format, Tag, and Modifier Registers

- This group of registers comprises ten flip-flops. Operation register has five; format register has one; tag register has two; modifier register has two.
- Each flip-flop is turned on when the corresponding B-register 0-9 FF is on at T2 of I1 cycles.
- Operation register is set by B-register positions 0-4. Format register is set by B-register position 5. Tag register is set by B-register positions 6 and 7. Modifier register is set by B-register positions 8 and 9.
- These registers specify the operation to be performed in execution of the instruction.

### Arithmetic Factor Register (D)

- Arithmetic factor register has 16 positions.
- D-register is set from the B-register with one factor for arithmetic and logical operations.
- D-register and the accumulator have interconnections that implement arithmetic and logical operations.
- D-register output can be gated to the accumulator unchanged.

Accumulator (A)

- Accumulator has 16 positions.
- A-register and D-register are interconnected to perform arithmetic and logical operations.
- Result of any arithmetic or logical operation is set into the A-register.
- The contents can be shifted right or left.
- Inputs:
  A0-15
  - D-register.
  - I-register.
  - Q-register.
  - U-register.
  - A- or Q-register when shifting.

Temporary Accumulator Register (U)

- Temporary accumulator has 16 positions.
- Input is from the corresponding positions of the accumulator.
- U-register provides temporary storage for the contents of the accumulator.
- U-register is set with the control word of the IOCC in 'XIO' operation.
- Output controls adapter circuits for XIO operations.

Accumulator Extension Register (Q)

- Accumulator extension register has 16 positions.
- Q-register is used as an extension of the low-order end of the accumulator for double precision.

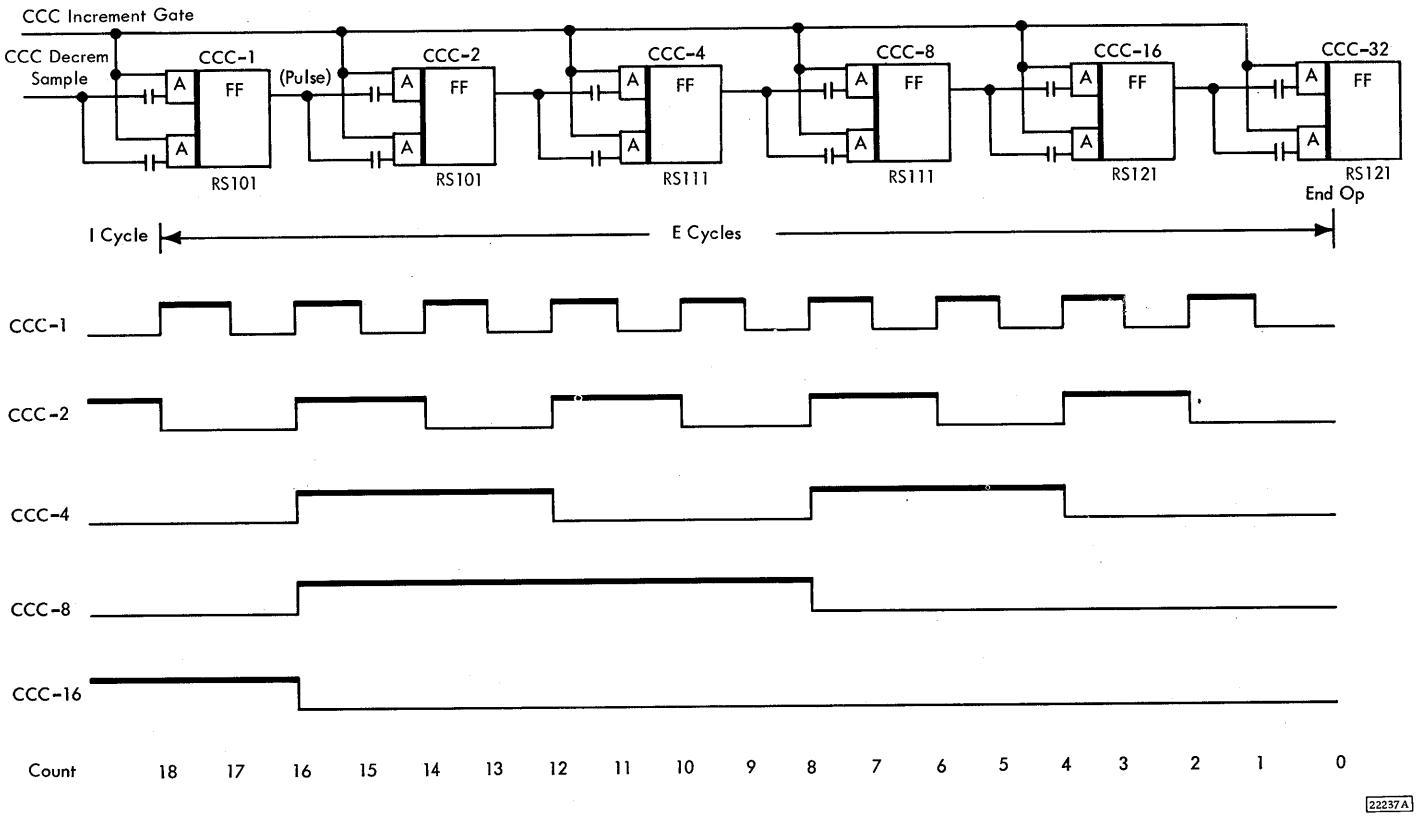



Figure 2-5. Cycle Control Counter Decrementing from 18

- Inputs and outputs are from and to the corresponding positions of the A-register to allow exchanging the contents.
- Q0 position has a shift right input from and a shift left output to A15 position.
- Contents can be shifted left or right one position at a time.
- Q15 has an input to set 1's in the divide quotient.
- Q14 and Q15 outputs are used to control multiply operations.

Instruction Address Register (I)

- Instruction address register is a 16-position register connected as a counter, and provides the address of the next instruction.
- Input is from B-register when the load IAR key is pressed.
- Input can be from M-register when incrementing circuit is not used.
- Contents can be increased by 1 each I-cycle.
- Output is set into the M-register at T0 time of I1 (and I2, if I2 is taken).
- Output can be set into the A-register for use in generating an effective address.

The 16 flip-flops of the I-register are connected as a binary counter (Figure 2-6). All the flip-flops are gated by the increment gate when incrementing is required. The timing of the 'start IAR increment' pulse varies with the operation being performed. This drive pulse changes the state of 'I15' FF. 'I15' turning off changes the state of 'I14' FF. This binary-counter method is used for all the flip-flops except 'I11,' 'I7,' and 'I3.' Figure 2-6 shows the method used to change the state of 'I11' FF. Note that only when 'I15' through 'I12' are all on is the AND satisfied. Then, when 'I15' turns off, the 'increment 11 sample' pulse occurs without waiting for 'I12,' speeding up the process. Similar circuits control 'I7' and 'I3.'

Storage Address Register (M)

- Storage address register has 16 positions.
- Input is from the I-register or the A-register.
- Output can be gated back into the I-register.
- With certain exceptions, M-register output addresses core storage.
- Positions of the M-register used to address core storage depend on the size of core storage.
- For index, cycle steal, or double-precision operations, other lines substitute for M-register output to address core storage.

The 16-position storage address register may be set either from the I-register or the accumulator. It must be set early in the core storage cycle if it is to be used to address core storage on that cycle. All 16 positions are not always necessary to address core storage. For example, only 'M4' through 'M15' FF's are used in addressing a 4k core storage.

For certain cycles of an index operation, an address of /0001, /0002, or /0003 is required. The M-register outputs are all blocked, and substitute lines are activated to simulate 'M14' and/or 'M15.'

In double-precision operations, 'M15' output is simulated to obtain the EA + 1 word from core storage.

For cycle steal cycles, the adapter circuits of the device which is cycle stealing provide lines to address core storage. These lines substitute for M-register outputs.

## CORE STORAGE

- The 4k or 8k core storage units are self-contained on a single SLT board.
- The 1130 models 1 and 2 have a core storage unit cycle time of 3.6 microseconds.
- The 1130 Model 3 core storage unit cycle time is 2.2 microseconds.

Magnetic Core Theory

A magnetic core is a small doughnut-shaped ring that is uniformly constructed of ferrite particles bonded together by a ceramic material. The ferrite particles have good magnetic properties and the core has a high retentivity of the magnetic flux lines after the magnetizing force is removed. It is this property of retentivity that makes a magnetic core useful as a storage device.

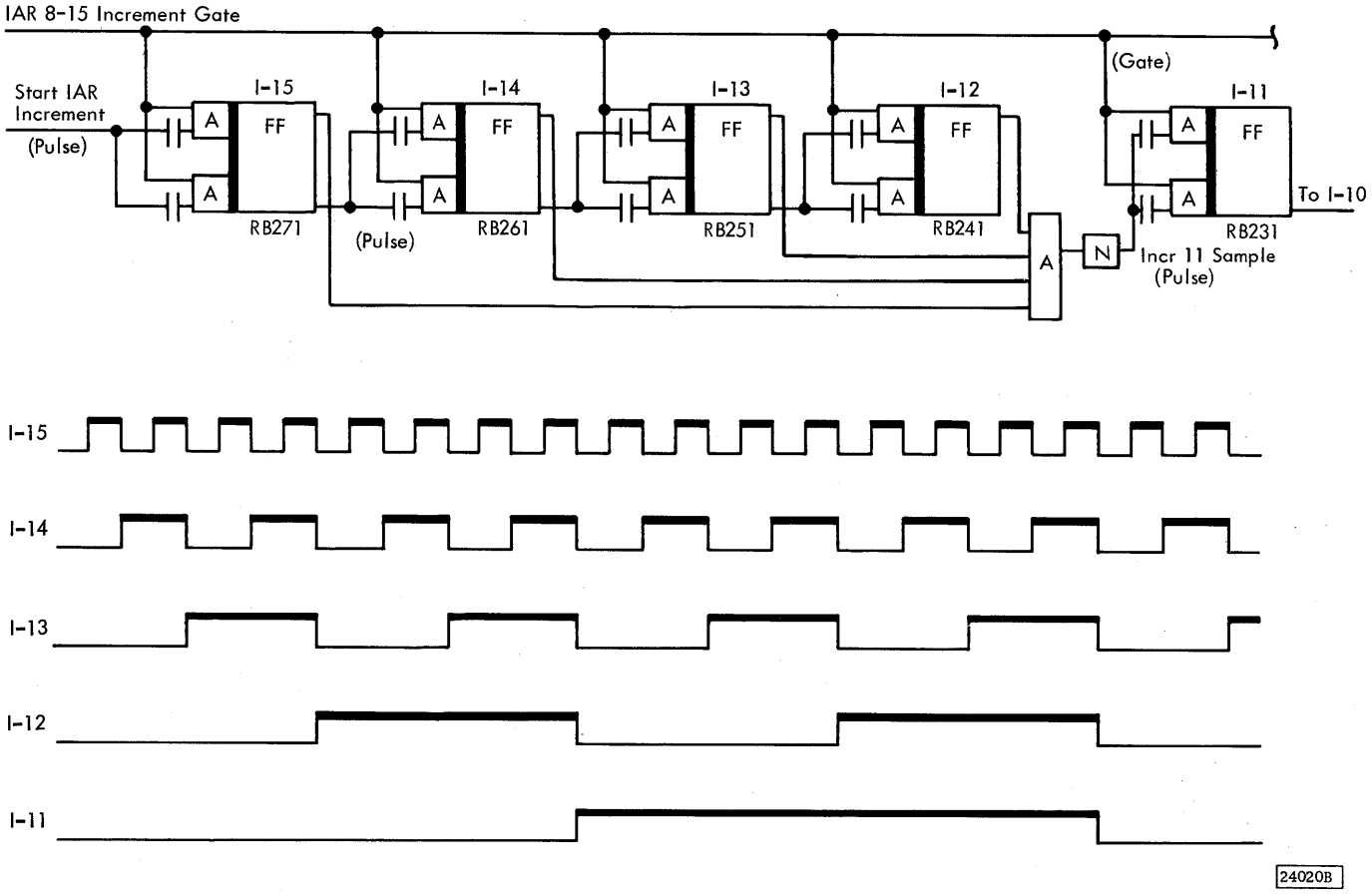



Figure 2-6. I Register Incrementing

The operation of a magnetic core can best be described by reference to the hysteresis curve, Figure 2-7. This curve is a plot of the relationship between a magnetizing current (Im) and the flux density.

A magnetic core is capable of maintaining indefinitely one of two stable magnetic states, either at point A or at point D on the hysteresis curve. Because the core has two stable states, it can be used as a binary storage device. At point A the core has a residual flux in a negative direction, and at point D a residual flux in a positive direction. The two directions can be arbitrarily assigned as binary "0" and binary "1" respectively.

The amount of current necessary to change the state of the core is called Im. Plus Im is that amount of current flowing in one direction, and minus Im the same amount of current flowing in the opposite direction.

On the hysteresis curve (Figure 2-7), it can be observed that a magnetizing current of plus Im changes the magnetism of the core from point A, a binary 0 to the magnetic saturation value in the

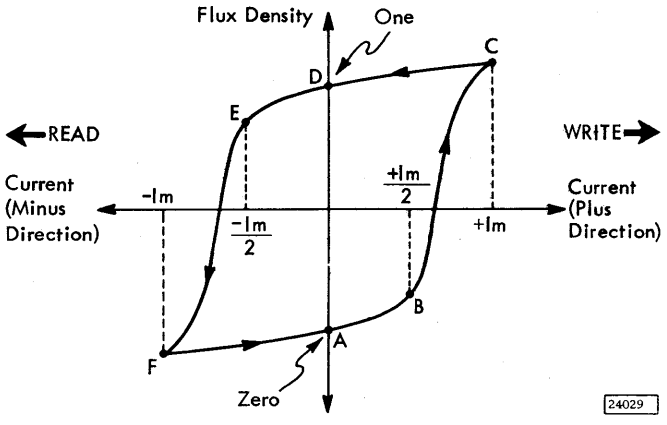



Figure 2-7. Hysteresis Curve

positive direction at point C. When the current is removed, the total amount of magnetization drops back to point D (binary one). If, instead of full plus Im, a current of plus 1/2 Im is applied, the flux change is from point A to point B on the curve, and when the current returns to 0, the flux returns to its original value at point A.

A reverse current, minus Im, develops flux of opposite polarity and changes the magnetic field of the core from point D to the magnetic saturation value in the negative direction at point F. The total amount of magnetization drops back to point A (binary 0) when the driving current is removed.

When a matrix of magnetic cores is constructed to store multiple bits of information, a specific core is selected (addressed) by the coincidence of 1/2 Im flowing through each of two wires threaded through the core. This coincidence gives a total effective current of full Im at the selected core. The state of the core is determined by the direction of the current flowing through these wires. When current Im is passed to store a 1 in the core, this is called "writing" into core. When current Im is passed in the opposite direction to change a stored value of 1 to a 0 and can be considered a current of minus Im, this is called "reading a core."

Reading a core depends on a system of sensing the state of the flux field within the core. When the core contains a 0 and a current of minus Im read current is passed through the core, the field changes from point A to point F on the hysteresis curve, which is a very small change in total flux density. If the core contains a 1 and a minus Im read current is passed through the core, the field changes from point D to point F and a large change in the flux field occurs.

A third wire, called the "sense" wire, is threaded through the core to recognize these changes in the magnetic field. Circuits are used to discriminate against low value 0 signals and amplify the large signal that results when a 1 is read from a core.

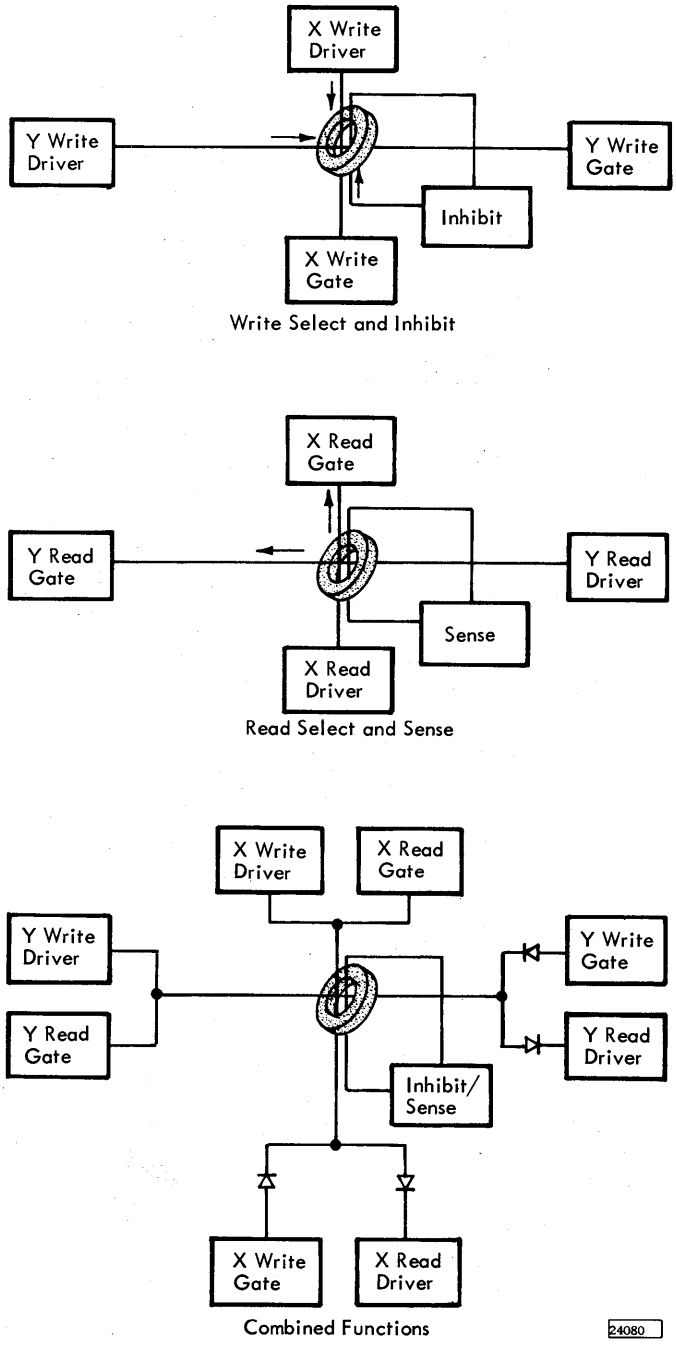



Figure 2-8. Core Storage Write and Read

## Writing Into Core

The magnetic properties of the core make it ideally suited for use in a storage matrix employing X and Y drive lines. Each core of the matrix is threaded by three windings (Figure 2-8). Two windings, the X and Y drive lines, each carry current (Im/2) in the same direction. The coincidence of current in these two windings occurs at one core storage position (18 cores) thereby "selecting" that word.

The third winding is the inhibit/sense winding. This winding is parallel to the X winding. When writing into a core storage position, each core that is not to receive a 1-bit is inhibited by a current Im/2 flowing in its inhibit/sense winding in a direction opposite that flowing in the X drive line. The magnetic field produced by the inhibit/sense winding effectively cancels the half field produced by the X

drive line. The resulting magnetic field is insufficient to flip the core to the 1 state.

Reading Out of Core

When information is to be read out of a core storage position, the currents Im/2 in the X and Y windings are reversed (Figure 2-8). Each core that flips from a 1 to a 0 state induces a pulse into its inhibit/sense winding. The inhibit/sense winding at each core does not carry inhibit current during read out and is used instead to detect the change in state of the magnetic flux. The sensed bits usually set the corresponding positions of a buffer register.

Addressing

- Data flow and control of the 3.6-microsecond core storage are shown in Figure 2-9.
- The address lines from the M-register are decoded to condition X and Y read/write drivers and gates.
- One active X line and one active Y line coincide at one core in each plane. (In a 4k array, nine planes, coincidence occurs at two cores in each plane, Figure 2-10.)
- Each active address line carries half-select current.
- Coincidence of Y half-select and X half-select currents address the 18 cores (one word).
- 3.6-microsecond reference diagrams (RD): SD011, SD012, SD041, SD042. Figures 2-9, 2-11, 2-12, and 2-13.
- 2.2-microsecond reference diagrams have an SA prefix (SA011, SA012, etc.).

Addressing 8k Array

The magnetic cores are arranged in matrices of 128 x 64 cores, called planes. (Each plane is actually 128 x 68, providing 512 positions of auxiliary storage that are not program addressable. The 1130 system does not use the auxiliary storage portion of core storage.) The 8k array consists of 18 of these planes with each plane assigned to one bit position of a core storage word. Corresponding core positions in each plane are addressed simultaneously by the X and Y drive lines to select one 18-bit word. Since there are 8,192 cores in each plane, the 8k array has a capacity of 8,192 words.

The 128 x 64 matrix in reference diagram (part of the core ALD's) SD041 represents one plane of the 3.6-microsecond core storage, 8k array. Addressing for the 2.2-microsecond 8k core storage is shown in reference diagram page SA041. It differs only in the drive circuits which are described in another section. The one core that is shown is selected by the address in the 13 positions of the address register (M register).

On side A, the 64 X drive lines, which run through all planes, are divided into eight groups of eight lines each. One group is activated by a decode of 0's (000) from M-register bit positions 3, 4, and 5. On side C of the array, one line in each group of eight is activated by a decode of 1's (111) from M-register bit positions 6, 7, and 8. Thus, a single X line is activated and carries half-select current.

At side B, the 128 Y lines which run through all planes are divided into eight groups of 16 lines each. One group of 16 lines is activated by a decode of 1's (111) from M-register bit positions 9, 10, and 11. At side D, one line in each group of 16 is activated by a decode of 0's (0000) from M-register bit positions 12, 13, 14 and 15. A single Y line is thereby activated and carries half-select current (Im/2).

In each plane, the one core at the intersection of the active X and Y drive lines receives full-current and is the only core in the plane that is selected.

Addressing 4k Array

One plane of the 3.6-microsecond core storage array is shown in reference diagram SD042. The 4k and 8k arrays occupy the same space in the SLT board. The core planes used in the 4k array are identical to those used in the 8k array, but only nine planes are used. For addressing purposes, each 128 x 64 plane is divided into two 128 x 32 half-planes. Each of the 18 half-planes is assigned to one bit position of the core storage word. Since there are 4,096 cores in each half plane, the 4k array has a capacity of 4,096 words. (An auxiliary storage of 256 words is actually part of the 4k array. However, these positions are not used in the 1130 system.) The 32 X drive lines are divided into four groups of eight lines each (one-half as many groups as in the 8k array). The X lines run through the nine half-planes on the side B of the array and then loop back through the nine half-planes on side D of the array. One group is activated by a decode of M register positions 4 and 5 (010), position 3 is not used with 4k core storage. At the other end of the X lines, one line in each group is activated by a decode of 1's (111) from M-register bit positions 6, 7, and 8. A

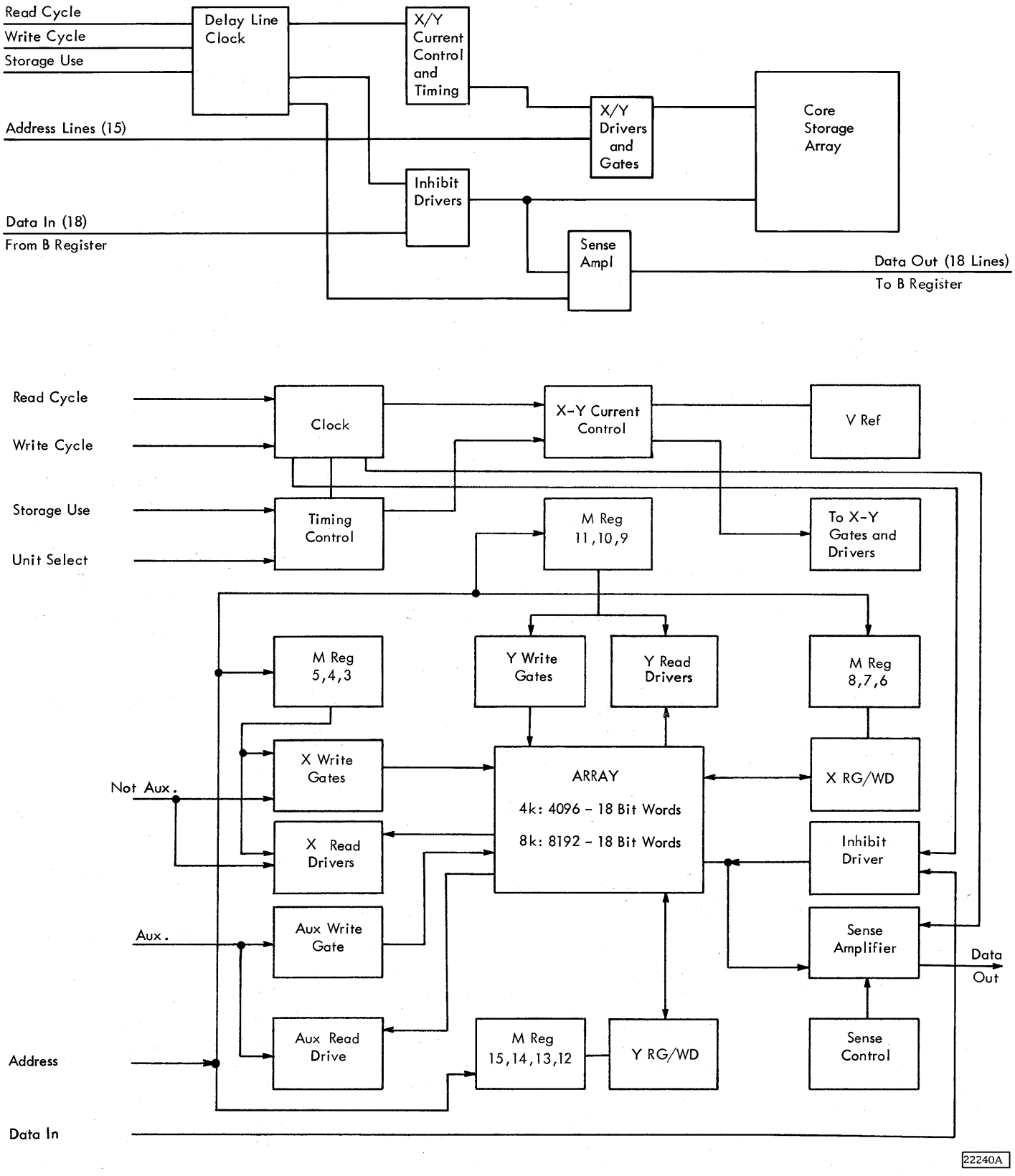



Figure 2-9. Core Storage Data Flow and Control

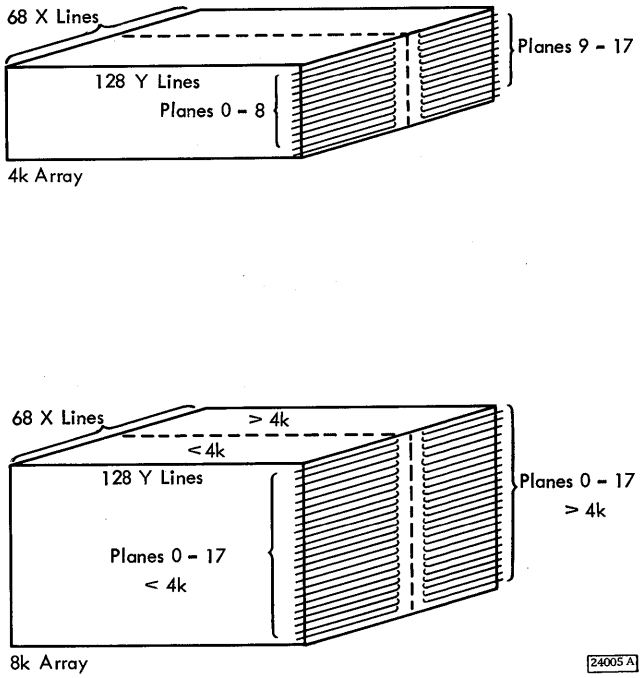



Figure 2-10. 4k and 8k Arrays

single X line is thereby activated and carries half-select current. The 128 Y drive lines are divided into eight groups of 16 lines each, as in the 8k array, and run through all nine planes. One group of 16 lines is activated by a decode of 1's (111) from M-register bit positions 9, 10, and 11. On side D of the array, one line in each group of 16 is activated by a decode of 0's (0000) from M-register bit positions 12, 13, 14, and 15. This active Y line intersects the active X line at two cores in each plane; therefore, nine planes provide 18 bits for each of 4,096 words.

Module Selection

- One module (one SLT board) of core storage can contain a 4,096- or 8,192-position array.
- Address register bit 1 and bit 2 lines are decoded to select the module in 16k and 32k core storage arrays.
- Module selection lines are connected through wired logic (Figure 2-14).

The circuits that activate the core storage clock and timing circuit, and therefore select the unit, require a specific combination of address-register bit-1 and -2 lines. These lines connect from the M-register output to the core storage circuits through wired logic. The table in Figure 2-14 shows the combinations required to activate each module.

Core Storage Arrays

- Core storage arrays are available in two sizes, 4,096 (4k) words (3.6-microsecond only) and 8,192 (8k) words (3.6- or 2.2-microsecond). (See Figure 2-10.)
- Each word consists of 18 bits.
- Both arrays consist of core planes containing a 128 x 68 matrix (128 x 64 program addressable).
- The 8k array contains 18 physical planes, each of which contains one bit position for each word.
- The 4k array contains nine physical planes, each of which contains two bit positions for each word.
- Reference diagrams: SD012, SD021, SD071, SD072, SD081, SD082 for 3.6-microsecond cycle core storage. Corresponding SA pages for 2.2-microsecond cycle.

The core storage array is mounted on an SLT large board. The 4k unit consists of 9 planes and the 8k unit consists of 18 planes.

In the 8k array, the X and Y drive lines address one bit in each plane for one word. (Figure 2-15). In the 4k array, the X and Y drive lines address two bits on each plane for one word. (Figure 2-16).

The array consists of a diode board, the core planes, a bottom board, and connecting pins.

The array is mounted on the large (SLT) board with four holding screws. Jumper blocks make the connections to the circuit pins which go through the large (SLT) board.

In a 3.6-microsecond system, a 4k array can be replaced by an 8k array. However, the 8k array requires additional SLT cards.

Differences Between the 2.2- and 3.6-Microsecond Arrays

These storage units use different address drive circuits and different inhibit/sense circuits, and the use of the array terminals is different. The 8k and 4k array descriptions given here are based on the 3.6-microsecond storage but also apply to the 2.2-microsecond storage arrays except for references to to the drive circuits. The differences can be seen

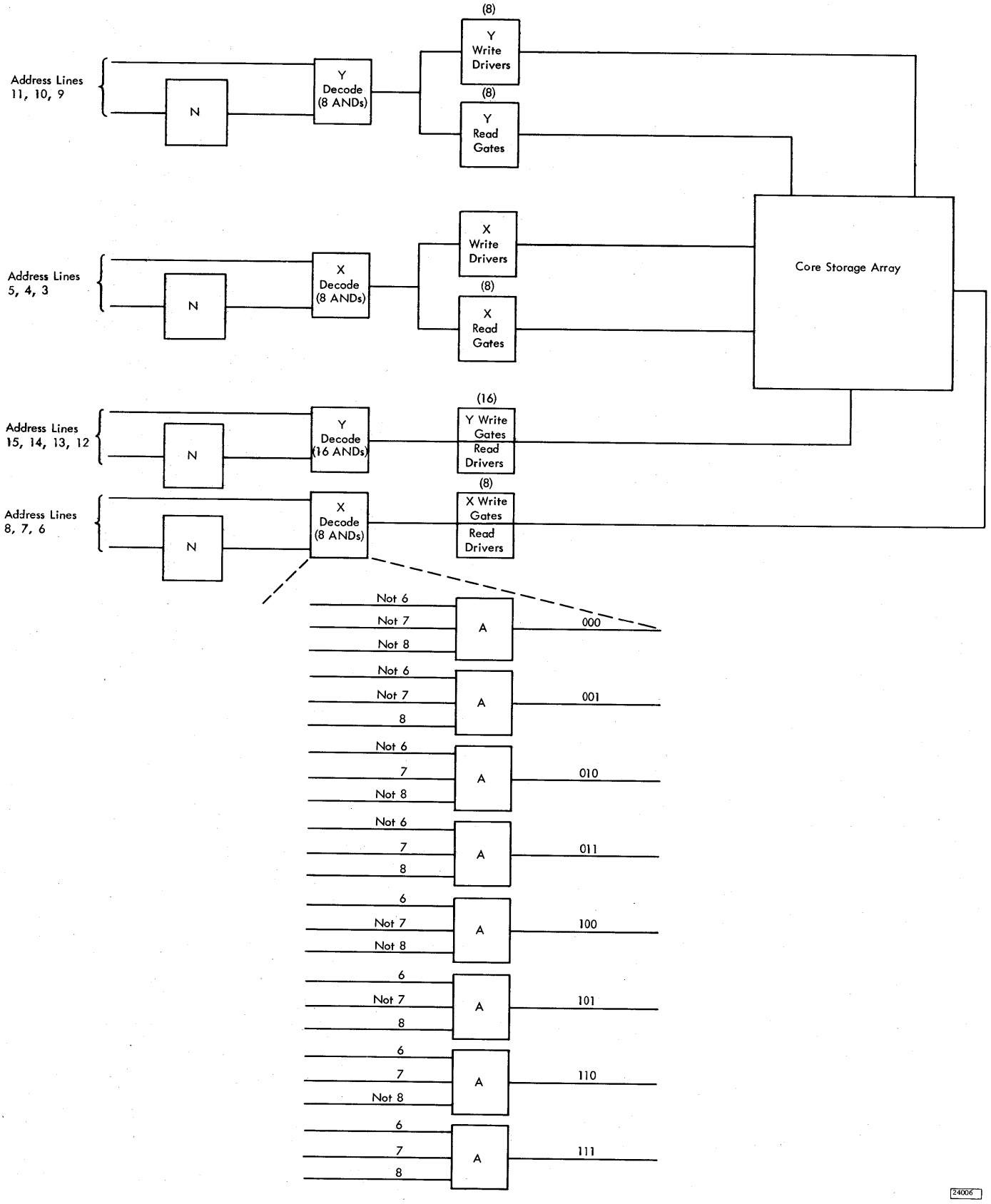



Figure 2-11. Address Decode

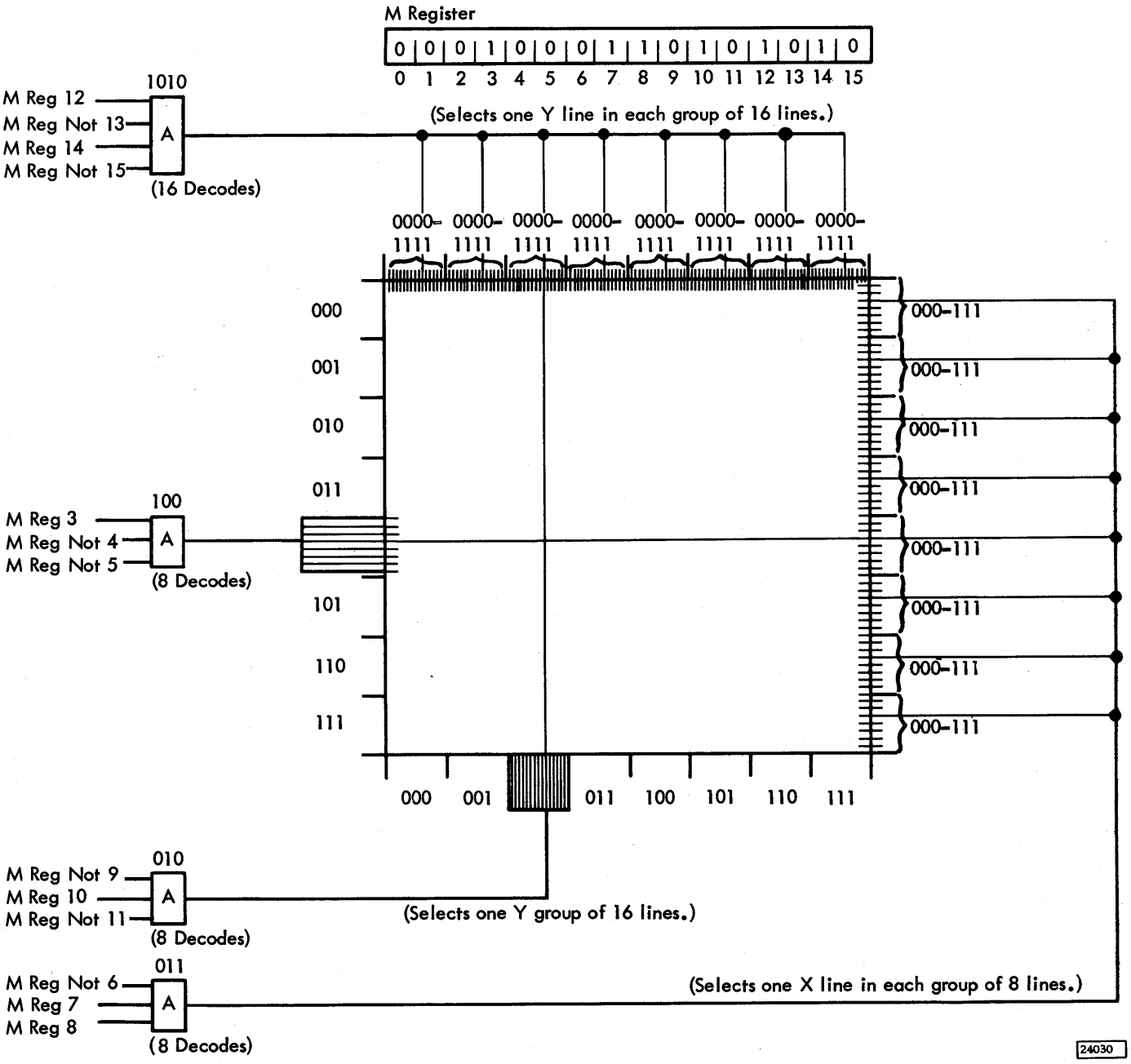



Figure 2-12. Addressing Scheme

by comparing bottom-board and diode-board diagrams for the two units. The reference diagram page numbers for the 3.6-microsecond storage begin with SD; the corresponding diagrams for the 2.2-microsecond storage have the same page number except they begin with SA. The inhibit/sense terminals for the two units are shown in SA/SD061.

8k Array

The 8k core storage array consists of a bottom board, 18 core planes, and a diode board. The bottom board plugs into the SLT board and provides connection from the SLT circuits to the X, Y, and inhibit sense lines. Reference diagram SD071 shows the

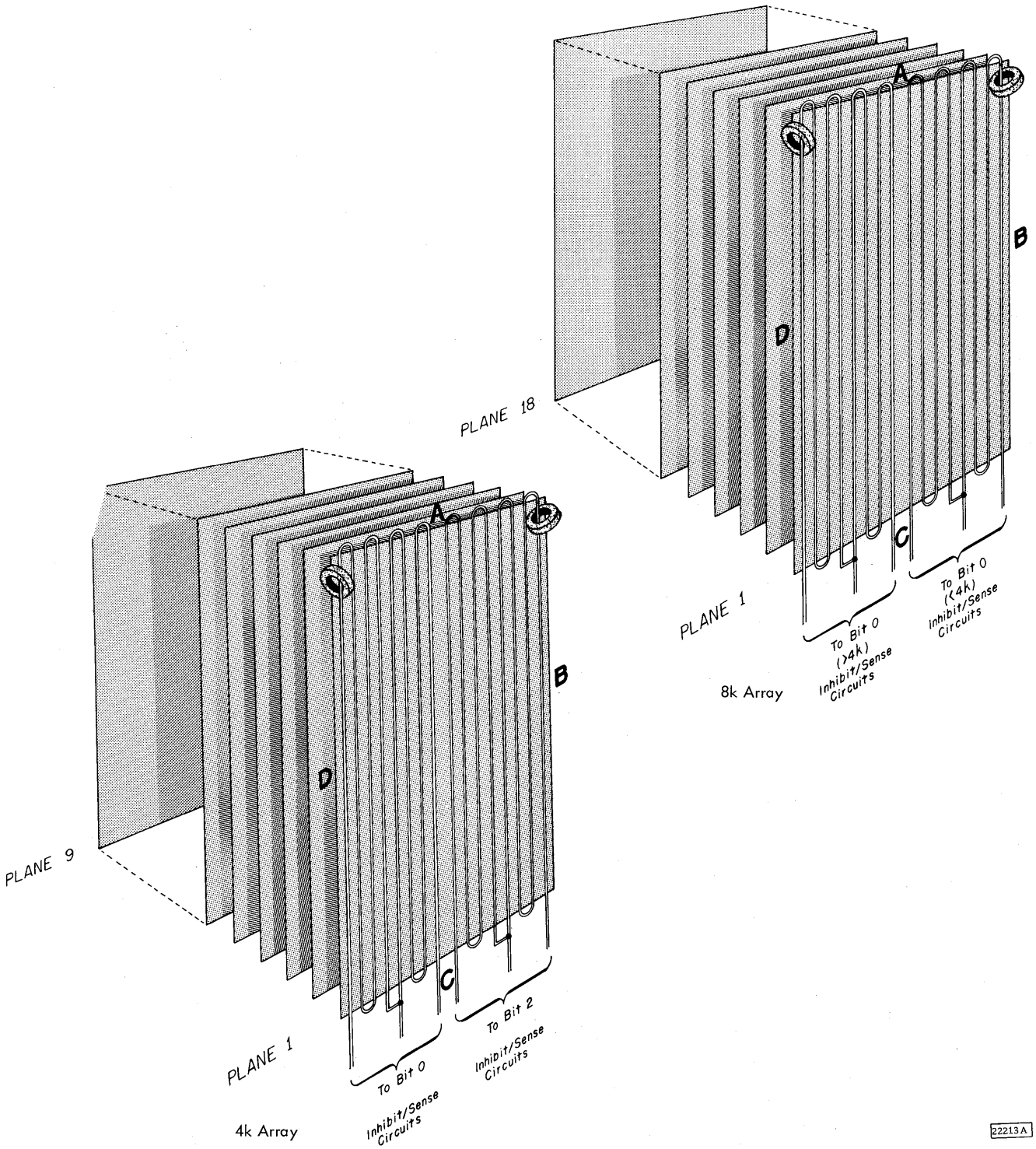
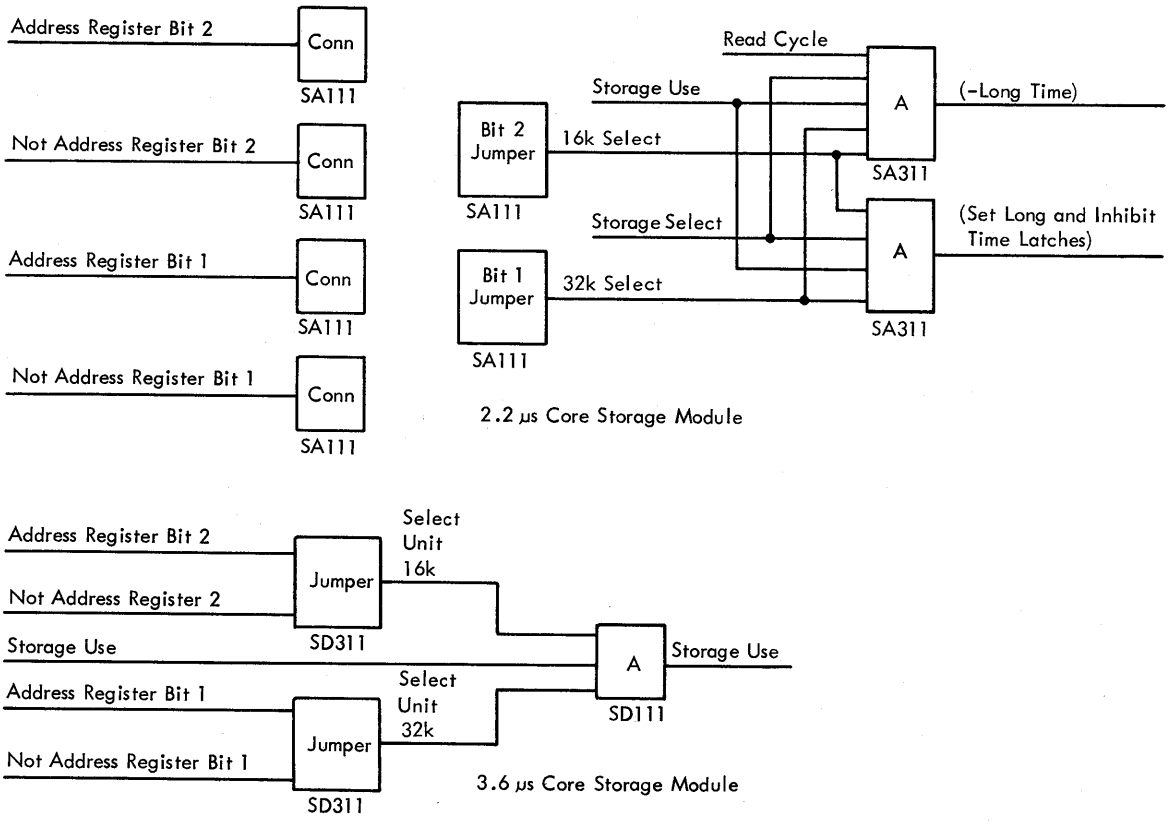



Figure 2-13. Inhibit/Sense Lines - 4k and 8k

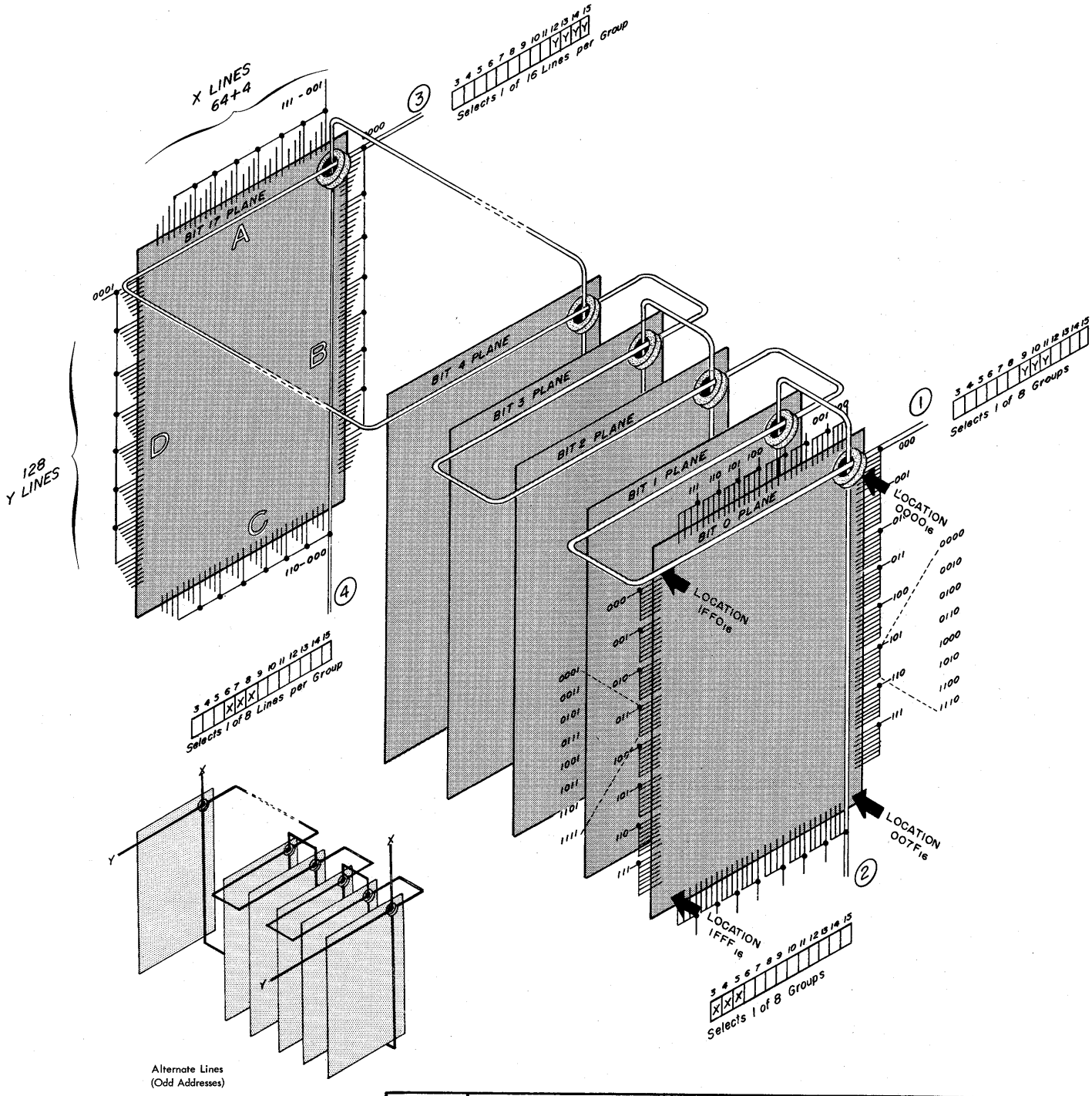



| System Storage Capacity | Module Select Jumpers | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 1 | | 2 | | 3 | | 4 | |
| | Bit 1 | Bit 2 | Bit 1 | Bit 2 | Bit 1 | Bit 2 | Bit 1 | Bit 2 |
| 4k | Not Used | Not Used | — | — | — | — | — | — |
| 8k | Not Used | Not Used | — | — | — | — | — | — |
| 16k | Not Used | 0 | Not Used | 1 | — | — | — | — |
| 32k | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 |

0 = Not Address Register Bit
1 = Address Register Bit

16201A

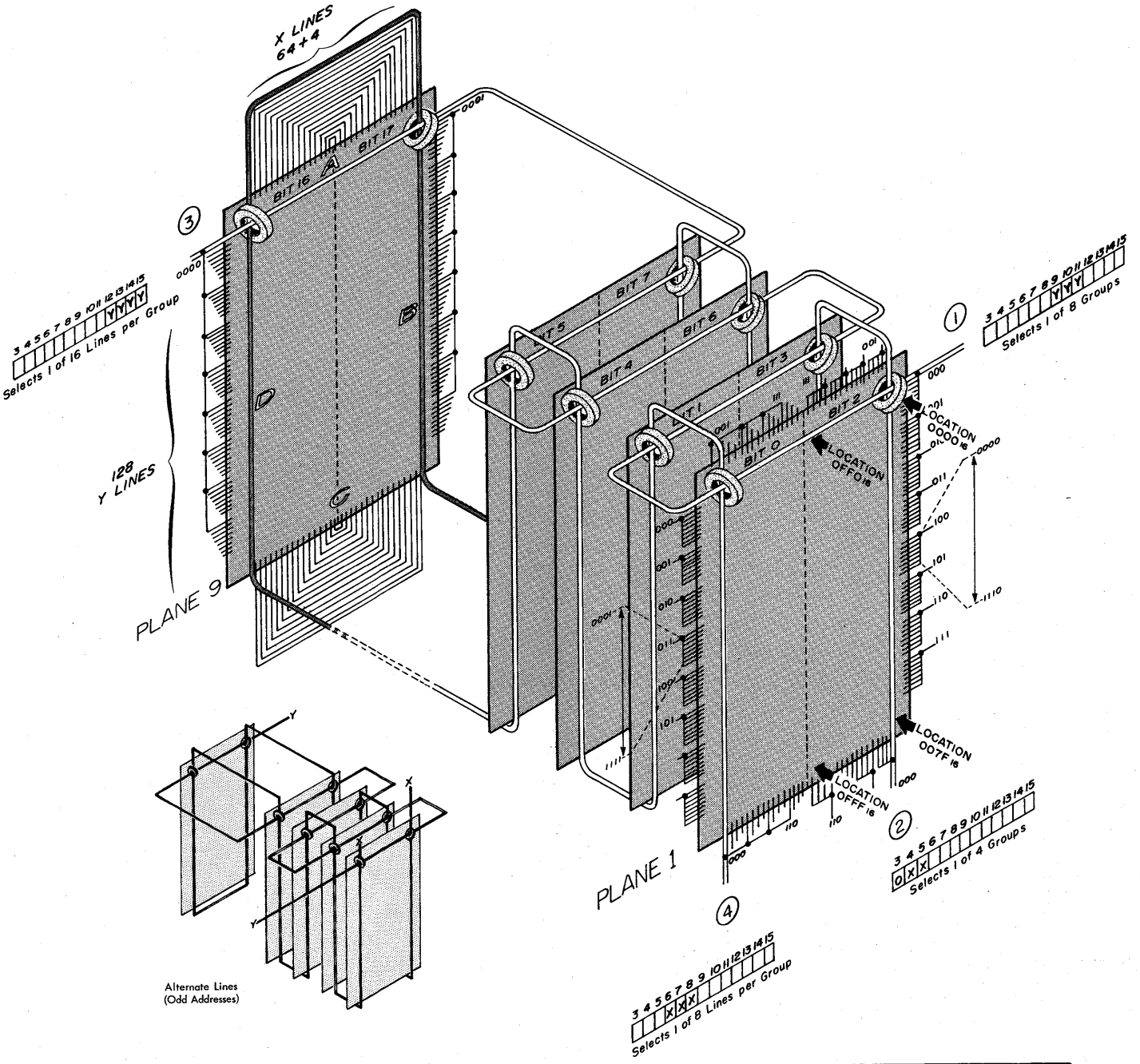
Figure 2-14. Core Storage Module Selection (16k and 32k)




| Note | 3.6 µs Storage | 2.2 µs Storage |
|---|---|---|
| ① | Y Read Driver/Write Gate | Y Read Gate/Write Driver |
| ② | X Read Driver/Write Gate | X Read Gate/Write Driver |
| ③ | Y Read Gate/Write Driver | Y Read Driver/Write Gate |
| ④ | X Read Gate/Write Driver | X Read Driver/Write Gate |

13266

Figure 2-15. X and Y Select Lines for Location /0000 (8k Array)




| Note | 3.6 μs Storage | 2.2 μs Storage |
|---|---|---|
| ① | Y Read Driver/Write Gate | Y Read Gate/Write Driver |
| ② | X Read Driver/Write Gate | X Read Gate/Write Driver |
| ③ | Y Read Gate/Write Driver | Y Read Driver/Write Gate |
| ④ | X Read Gate/Write Driver | X Read Driver/Write Gate |

Figure 2-16. X and Y Select Lines for Location /0000 (4k Array)

land pattern for the terminals; the connectors are not shown. The even-addressed Y write driver and read gate lines go from the connectors, through the land pattern, to terminals B (4) - (35) and B (46) - (77); the odd-addressed lines go to terminals D (4) - (35) and D (46) - (77). The even addressed X write driver and read gate lines go to terminals C (2) - (17) and C (20) - (35); the odd addressed lines to terminals A (2) - (17) and A (20) - (35). The write driver/read gate windings of bit 17 plane connect directly to the bottom-board terminals.

The X and Y write gate and read driver lines go from the connectors, through the bottom-board land-pattern, to the following terminals: even-addressed lines to B (1) - (3), B (36) - (45), and B (78) - (80); odd-addressed lines to D (1) - (3), D (36) - (45), and D (78) - (80). The lines then go along the side of the array to the corresponding terminals of the diode board. The diode board land pattern (reference diagram SD081) connects the lines to the common terminals of the diode packs. From the individual diodes, the land pattern connects to the terminals at the sides of the diode board.

Diode Board Terminals: In reference diagram SD081 note that the storage address register is divided into X and Y high-order and low-order positions. If the X portion of the specified address is even (position 8 = 0), the active X terminal is on side C of the array. If the X portion of the address is odd (position 8 = 1), the active X terminal is on side A of the array.

If the Y portion of the address is even (position 15 = 0), the active Y terminal is on side B of the array; if odd (position 15 = 1), the active Y terminal is on side D.

Locate the specified terminals by finding the group of lines associated with the X and Y high-order positions of the address. The relative position of the line within the group is the same for all low-order values and is identified in the enlarged drawing (within broken lines) for each side of the diode board.

Array Windings: The diode board terminals can be related to the terminals on each plane and the lines can be traced through the array by referring to Figure 2-15. This figure shows the relative location of the 18 cores that form the word at core storage address hex 0000. Note that the Y address line for this address enters bit 0 plane at side B and is activated by positions 9, 10, and 11 (Y high order) of the address register. The X line enters the bit 0 plane at side C and is activated by positions 3, 4, and 5 (X high order) of the address register. The two lines go through every plane, intersecting at location /0000 of each plane.

From the bit 17 plane, the Y line exits at side B and is activated by positions 12-15 (Y low order) of the address register. The X line exits at side C and is activated by positions 6, 7, and 8 (X low order) of the address register.

The X and Y lines for an odd address enter bit 0 plane and exit bit 17 plane at the sides of the array opposite those for an even address, as shown in the inset in Figure 2-15. The bottom board connects the lines to the inhibit/sense cards through the land patterns and connector blocks (reference diagram SD012).

Bottom Board Terminals: From the terminals of bit 17 plane, the lines go to the bottom-board terminals (reference diagram SD071), through the land pattern to the connectors (not shown), and through the SLT board land pattern to the drivers and gates (reference diagram SD012).

The bottom board also provides connection from the inhibit/sense connectors (not shown) that plug into the SLT board to terminals (1S) - (54S) at side A and C.

4k Array

The 4k array consists of a bottom board, nine core planes, and a diode board. As described in the "Addressing" section, the core planes used in the 4k and 8k arrays are the same. The Y windings in the two arrays are connected in the same way, but the X windings are connected differently. Because each physical plane must provide two bit planes, the X windings go through the B half of each plane in the array and then through the D half. Reference diagram SD072 shows the land pattern of the 4k bottom board.

The bit 16 and 17 X lines shown in Figure 2-16 connect to the bottom board terminals at side A and C. Side A of the bottom board land pattern thus provides the "jumpers" for the even-addressed X lines between bit 16 and 17, and side C provides the "jumpers" for the odd-addressed X lines. The reference diagram SD072 shows the land pattern for the terminals; the connectors are not shown.

The X and Y write driver and read gate lines go from the connectors, through the bottom-board land-pattern, to terminals at the sides of the array. The write driver/read gate windings of bit 16 and 17 planes connect directly to the bottom board terminals.

The X and Y write gate and read driver lines go from the connectors, through the bottom-board land-pattern to the indicated terminals, and along the sides of the array to the corresponding terminals of the diode board. The diode-board land pattern (reference diagram SD082) connects the lines to the common terminals of the diode packs (reference diagram

SD012). From the individual diodes, the land pattern connects to the terminals at the sides of the diode board.

Diode Board Terminals: The function of the 4k diode board (reference diagram SD082) is the same as the previously described 8k diode board. One-half of the X read driver/write gate diode pacs are not used because there are half as many X drive lines in the array. The terminals are numbered differently and sides A and C are used differently.

The B half of sides A and C provide connection from the diodes to the X read driver/write gate lines in groups of eight lines as do the corresponding terminals of the 8k diode board. The D half of sides A and C provide connection from the other end of the lines to the X read gate/write driver lines within the groups.

Array Windings: Figure 2-16 illustrates the 4k array bit-plane layout, the paths of the X and Y windings, and the relation of the windings to the terminals of the bottom board and diode boards. The windings within the planes are the same as those in the planes of the 8k array; it is the bottom board and diode board that cause the array to function as 4096 eighteen-bit words.

The even-addressed Y lines, from the read driver/write gate circuits, enter plane 1 through the bit-2-plane half (side B), go through every plane, and exit plane 9 at the bit-16-plane half (side D). Odd-addressed Y lines (Figure 2-16 inset) enter plane 1 at side D and exit plane 9 at side B. Even-addressed X lines, in four groups of eight each, enter plane 1 at the B half of side C, go through the B half of every plane, and loop back (by the bottom-board land pattern) to the D half of plane 9. The lines then go through the D half of every plane and exit plane 1 at side C to connect to the diode board. The lines are connected through the diode board and the bottom board, to the X read gate and write driver circuits.

Odd-addressed X lines (Figure 2-16 inset) enter plane 1 at side A, go through every plane, loop back (by side C of the bottom board), and exit plane 1 at side A.

The inhibit/sense lines for a 4k array are shown in Figure 2-13. The lines for plane 1 (bit 0 and 2) connect to bottom board terminals at side C; alternate planes connect to sides A and C. The bottom board connects the lines to the inhibit/sense cards through the land patterns and connector blocks (reference diagram SD012).

Drive Current Generation -- 3.6 Microsecond Storage

- Gate and driver circuits direct the drive current through the array.
- Core storage is temperature sensitive.
- A temperature-compensated voltage reference is applied to the current control circuit to ensure optimum core storage operation.
- The current control circuit provides a constant current (current sink) for the array.

The direction in which the half-select current flows in the X and Y address drive-lines is controlled by the driver and gate circuits. These circuits are conditioned by timing, for the read and write portions of the cycles, and by the address lines. The block diagram of the 3.6-microsecond core storage (reference diagram SD011) shows the read and write drivers and gates controlled by the M-register lines. Timing control is implemented through the current control block at the top of the diagram.

Reference diagram SD042 shows more detail of the drive line circuits. Looking at the one X drive line shown, note that address lines 3, 4, and 5 condition eight decode circuits. One decode circuit (decode 000 is shown) conditions both an X write gate circuit and X read driver circuit. During read time, X read driver control further conditions the X read driver to allow current to flow through the read driver to the X read sink. (Electron current flow is indicated in the core storage reference diagrams.) This X read driver activates a group of eight X lines.

One of the eight active lines is also activated at the other end of the array by an X read gate circuit. The read gate is conditioned by a decode of address lines 6, 7, and 8. (Decode 111 is shown.) The read gate is further conditioned by X read gate control and current is allowed to flow from the 8.3-volt source through the X drive line. This read gate activates one line in each group of eight, but only one line is activated at both sides.

During write time, the same address decodes are still conditioned. (The address lines do not change between read and write time.) The X read driver control is no longer active; the X write gate control becomes active and conditions the X write gate to activate the same group of eight lines. At the other end of the array, the X write driver is

conditioned by the same address decode and by X write driver control to allow current to flow to the X write sink. Half-select current now flows in the same line as during read time but in the opposite direction.

Reference diagram SD043 shows still more detail of the driver and gate circuits. In this diagram, read current (electron current direction, negative to positive) is shown by the heavy lines. The controlling circuits, identified by broken lines, must be conducting to activate the read gate and read driver circuits.

When the read or write timing pulses are not conditioning the control circuits (quiescent state), current is in the driver control and gate control circuits only. The center transistors of these two circuits are forward biased by the 1.4 volts at their bases; the only current is through these transistors.

The read gate circuit conducts when all the address-line (low-order bits) inputs are positive and the read gate control circuit is conditioned by the read-timing inputs.

The read driver circuit conducts when all the address-line (high-order bits) inputs, located in the write gate portion of the diagram, are positive and the read driver control circuit is conditioned by the read-timing inputs.

The current in the read gate control circuit places a bias on the emitters of the center transistors of the driver control and gate control circuits, cutting them off.

Half-select current is now flowing through the selected X and Y drive lines in a direction to flip the cores from the 1 state to the 0 state or read the addressed position of core storage.

After the read timing pulses fall, the circuits return to the inactive state, and current flows in the gate control and driver control circuits only. When the write timing pulses rise, current is switched to the write gate and write driver circuits and therefore flows through the array in the same drive line but in the opposite direction. This X and Y half-select current causes the selected cores to flip to the 1 state, thus writing into the addressed position of core storage.

The V-reference voltage applied to the current control circuit is temperature compensated, causing the X-Y drive current to track along its optimum value over a specified temperature range.

Drive Current Generation -- 2.2-Microsecond Storage

- Gate and driver circuits direct the driver current through the array.
- The read source circuit provides a constant current for the array during the read cycle.
- The write sink circuit provides a constant current for the array during the write cycle.
- The temperature-compensated voltage reference is applied to both the read source and write sink circuits to ensure optimum core storage operation.

As in the 3.6-microsecond core storage, the direction in which the half-select current flows in the X and Y address line is controlled by the driver and gate circuits. These circuits are conditioned by timing, for read and write portions of the cycle, and by address lines.

Comparison of the 3.6-microsecond addressing diagrams with the 2.2-microsecond storage addressing diagrams (SA041 and SA042) shows two basic differences.

The driver and gate circuit locations are reversed; that is, where the 3.6-microsecond storage uses the Y write gate, the 2.2-microsecond storage uses the Y read gate. The other basic differences that can be seen in these diagrams are the current source and sink. The 3.6-microsecond storage current (electron current negative to positive) is from -3V array current sink to +8.3V for both read and and write currents. The 2.2-microsecond storage read current is from -15V to the +12V read source and the write current is from the -15V write sink to ground.

Reference diagram SA043 is a detailed diagram of the 2.2-microsecond storage drive circuits. Electron current for the read portion of the cycle is from the selected read gate to the read source circuit (+12V), under control of read timing signals. The current is through an isolation diode and the drive line that is connected to the active read driver at the other end of the array. The read driver is conditioned by the decode of the address lines and by a read timing pulse. The current is from -15V to the driver.

Write current is from the active write gate to ground and is activated by a write timing pulse and an address decode. Current is through the array line that is in the group of lines connected to the active write driver. Current is from the write sink circuit and -15V to the isolation diode to write driver and is activated by write timing pulses and an address decode.

The magnitude of current is controlled by a temperature-controlled reference voltage, V-Ref, that is applied to both the read source and write sink circuits.

The array termination, consisting of a resistor at the common of each group of isolation diodes, provides a characteristic impedance to the array lines. These circuits absorb the discharge pulse from the nonselected array lines and any reflected waves that build up on the lines.

Inhibit/Sense -- 3.6-Microsecond Storage

- The inhibit/sense winding is common to every core in one bit plane in the 4k array, common to every core in one-half of one bit plane in the 8k array.
- During read time, the inhibit/sense windings conduct pulses, caused by the 'flipping' of selected cores, to the sense amplifier circuits.
- During write time, the inhibit/sense winding carries inhibit current to prevent the 'flipping' of selected cores.

Each inhibit/sense winding passes through 4,096 cores of one plane. The 8k array has two inhibit/sense windings per bit plane; the 4k array has one per bit plane. Figure 2-13 shows the inhibit/sense scheme for the 4k and the 8k arrays.

The functions of the inhibit/sense winding are explained in the "Magnetic Core Theory" section of this chapter.

Inhibit Driver

Reference diagram SD051 shows the 3.6-microsecond storage inhibit/sense circuits; SD051 shows some of the timing pulses of these circuits. The inhibit driver has two outputs that drive a center-tapped winding. Each output supplies approximately 200 ma of inhibit current. Data from the CPU B-register is gated to the inhibit drivers by a storage timing pulse. When the inhibit driver is conducting, inhibit current prevents the writing of a 1 in the selected core storage location by cancelling the X half-select current.

Sense Amplifier

The sense amplifier has a differential input and a single-ended output. The first stage of this three stage amplifier is deactivated during the write cycle by the emitter-strobe pulse.

The first stage is kept out of saturation by a 0.7-volt offset voltage that is applied to the second stage. During the read cycle, the output of the second stage is gated to the final stage by the sense strobe. When a 1 is sensed, the output of the final stage is a negative-pulse.

Inhibit/Sense -- 2.2-Microsecond Storage

- The 2.2-microsecond storage inhibit/sense windings serve the same purpose as the 3.6-microsecond storage inhibit/sense windings.
- The connections on the array are different for the two units as shown in reference diagrams SA/SD061.
- The 2.2-microsecond storage inhibit/sense circuits are shown in reference diagram SA051.

Inhibit Driver

Three inputs condition the inhibit drivers, as follows: data, timing, and an address line. The data inputs are the 'not' lines from the B-register that allow inhibit current to flow in the bit planes in which a 1 bit is not to be written. The timing input is the inhibit time pulse, described in the timing section that follows. The address line is the 'M-register bit 3' line that separates inhibit drivers into less than 4k and greater than 4k circuits.

The inhibit driver output attaches to the inhibit/sense lines at the center point of 4,096 cores. The two end points of these lines connect through diodes to ground. Each inhibit circuit provides control for one-half a bit plane (less than 4k or greater than 4k) in an 8k array.

Inhibit current is from ground, through 2,048 cores, through the inhibit driver that is conditioned, to -V inhibit (Vz). Vz is a temperature-responsive voltage that ranges from -9 to -12 volts.

Sense Amplifier

As in the 3.6-microsecond storage, the 2.2-microsecond storage sense amplifier has a differential input from the two ends of the inhibit/sense line for 4,096 cores. The center point of this line connects to the corresponding inhibit driver.

The first-stage collectors contain a transformer circuit that eliminates any dc imbalance to provide an equal detection threshold for both polarities of signal. This stage does not require the emitter strobe that the 3.6-microsecond sense amplifier requires. The 0.8 volts at the emitter of the second stage keeps the first stage out of saturation.

The sense-strobe pulse gates the output of the second stage to the final stage. A negative pulse output of the final stage results when a 1 bit pulse

is conducted from the array to the sense amplifier input.

Timing -- 3.6-Microsecond Storage

- Reference diagram SD031 shows the timing pulses.
- X read gates and drivers are conditioned at T0 time for a duration of approximately 1 microsecond.
- X write gates and drivers are conditioned at T4 time for a duration of approximately 1 microsecond.
- The conditioning of Y read and write gates and drivers is delayed to allow X drive line noise to subside.
- Timing-pulse generation is initiated by the transition of the CPU read/write cycle.

Figure 2-17 represents the 3.6-microsecond storage "clock" and timing circuits. The CPU provides the following signals to the storage timing circuit: read cycle, write cycle, storage use, and the M register 3 address line. The following timing pulses are generated by the "clock" circuit: long time, short time, and strobe. Combinations of the CPU signals and the generated timing pulses provide read and write gate and driver control, strobe pulses for less than and greater than 4k, emitter strobe pulse, and inhibit pulses for less than and greater than 4k.

Note in the timing chart (SD031) that Y read current and Y write current rise after X read current and X write current. The delay of the Y currents minimizes the effect of excess noise at the rise of the X current pulse. This noise is caused by the discharge of array capacitance.

3.6-Microsecond Storage Clock

- A time delay "clock" circuit (Figure 2-17) develops the necessary timing pulses for addressing, reading from, and writing into core storage.
- 'Long time': a time delay circuit provides a 1-microsecond pulse at T0 (read cycle) and at T4 (write cycle).
- 'Short time': a latch provides a pulse that begins after the rise of 'long time' and ends with the fall of 'long time'.
- Strobes: A singleshot, activated by 'short time', provides a pulse that is ANDed with 'read cycle' to activate one of two strobe pulses.
- Inhibit lines: 'long time' and 'write cycle' are ANDed to activate one of two inhibit lines.
- 'Address register bit 3' line selects the strobe pulse and inhibit line that is activated.

Time Delay Circuit

Read cycle and write cycle are connected to opposite ends of a 1 microsecond delay line (Figure 2-18). The low resistance of the delay line causes the levels at the inputs (points 1 and 2) to balance. For example, if 'read cycle' is 0 volts and 'write cycle' is +3 volts, points 1 and 2 balance at +1.5 volts.

When 'read cycle' and 'write cycle' change levels (at T0 and T4 time), the input points reflect the respective levels for 1 microsecond, then balance again. The same effect is evident at the taps (points 3 and 4) except for a shorter duration.

Timing -- 2.2-Microsecond Storage

- Reference diagram SA031 shows the timing pulses.
- Figure 2-19 shows the "clock" and pulse generation circuits.
- X read gates and drivers are conditioned at T0 time for approximately 750 ns.
- X write gates and drivers are conditioned at T4 time for approximately 750 ns.
- Y read drivers and write gates are conditioned at the same time as the X drivers and gates.
- The conditioning of Y read gates and write drivers is delayed 100 ns after the X drivers and gates to allow X drive line noise to subside.

The 2.2-microsecond storage "clock" consists of a time delay circuit, three latches, and two singleshots. The timing pulse generation is initiated by

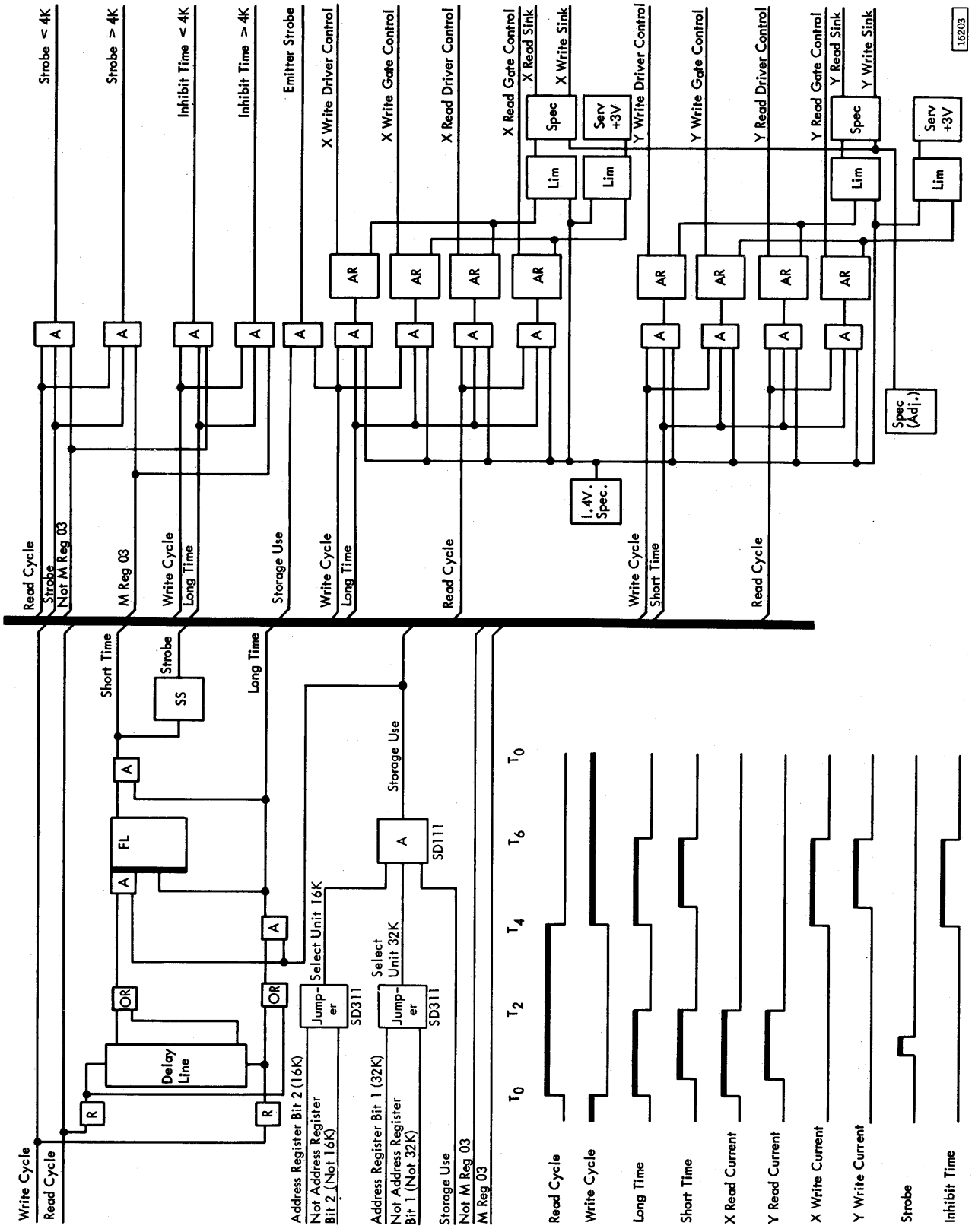



Figure 2-17. 3.6 Microsecond Core Storage Clock

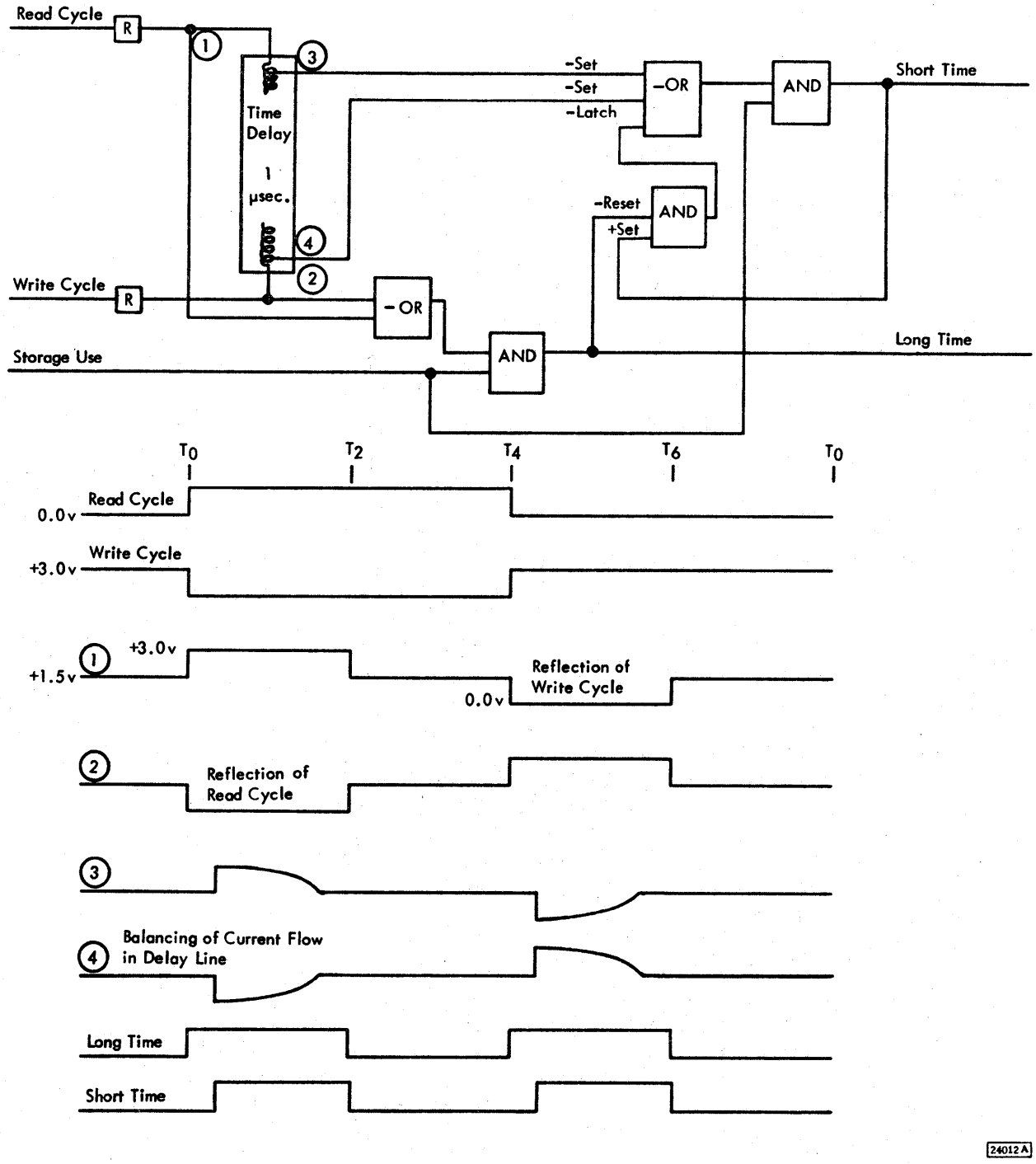



Figure 2-18. Time Delay Circuit, 3.6 Microsecond Storage Clock

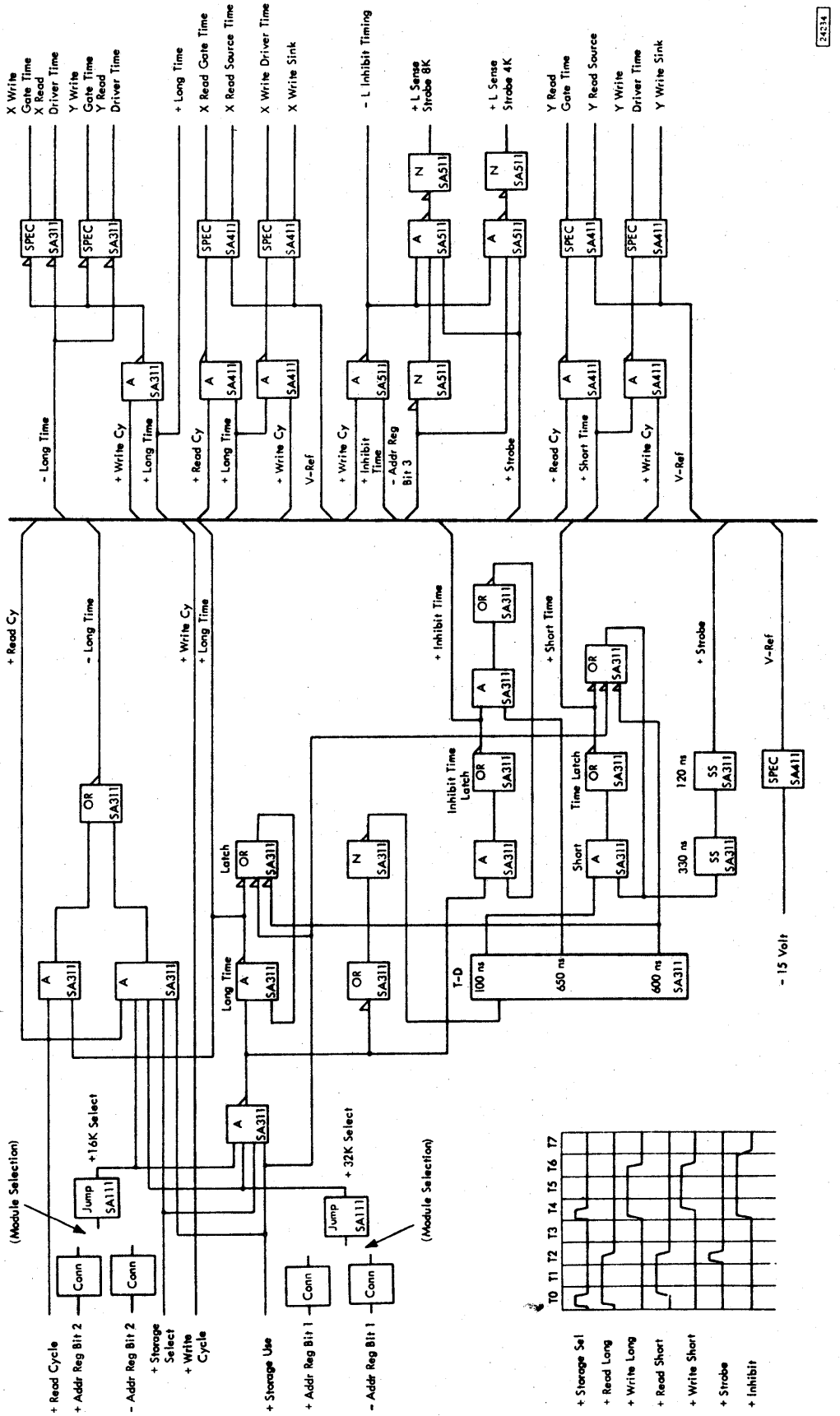



Figure 2-19. 2.2 Microsecond Core Storage Clock

the 'storage select' pulse from the CPU. 'Storage select,' which is a 100-ns pulse at T0 and T4 or X0 and X4, turns on the 'long time' latch and the 'inhibit time' latch. 'Storage select' is also connected to the input of a time delay circuit for the following: the 100-ns delayed outputs turns on the 'short-time' latch; the 600-ns outputs turns off 'long time' and 'short time' as well as 'inhibit time.'

The timing chart shows theoretical and actual timing for each signal. It shows 'long time' coming up soon after T0 time, falling at the 750-ns point, rising again soon after T4 time, and falling 750 ns later. The 'inhibit time' latch is turned on at the same time that 'long time' is, but the timing shown is that of 'inhibit timing,' after it has been ANDed with 'write cycle.'

'Short time' rises 100 ns after 'long time' and falls at the same time as 'long time.' The strobe pulse is generated by two singleshots, the first of which is started by 'short time.'

'Long time' and 'short time' are ANDed, in various combinations, with 'read cycle' and 'write cycle' to produce X and Y gate/driver timing. The X read and write gate and driver circuits are conditioned by 'long time.' The Y read driver and write gate is also conditioned by 'long time'; the Y read gate and write driver are conditioned by 'short time.'

Strobe is ANDed with 'not inhibit timing' and 'address register bit 3' to produce 'sense strobe 8k'; 'not address register bit 3' used to produce 'sense strobe 4k.' In the timing chart, the Y current (low-order end) shows a pulse at T0 and T4 time. This pulse is present because of the initial charging of array capacitance, although the Y line is not conditioned at the other end at this time. The Y read current (high-order end) and Y write current (high-order end) begin at 'short time,' when both ends of the lines are conditioned.

'Inhibit time' is ANDed with 'write cycle' to produce 'inhibit timing,' shown in the timing chart. 'Inhibit timing' is ANDed with the data lines and 'address register bit 3' or 'not address register bit 3' (not shown in the diagram) to activate the inhibit drivers for less than or greater than 4k.

## ADD-SUBTRACT CIRCUITS

- Add-subtract circuits include the D-register, the A-register, their interconnections, and controls.
- Factor in the D-register is added to or subtracted from factor in the A-register.
- All positions are affected simultaneously (parallel).
- Negative factors must be in 2's complement form, and negative results are developed in 2's complement form.
- The A-register contains the result, and D-register contains all 0's at the end of add or subtract operations.

Although not used exclusively for add or subtract operations, the D-register and the A-register are interconnected and can be controlled to perform these operations. The add-subtract circuits are sometimes used during effective address generation as well as during execution of arithmetic and logical instructions. All positions of the D-register are added to or subtracted from the corresponding positions of the A-register in parallel. When negative factors are used, they must be in the register(s) in the 2's complement form. At the end of add or subtract operations, the D-register contains all 0's and the A-register contains the result.

Add or subtract can start after the 'arithmetic control trigger' FF is turned on at T3, activating both 'add/sub/or/EOR gate' and 'add/sub/EOR gate'. Dependent on the bit configuration of the factors, the operation may or may not be completed by T7. If 'arith control' is not deactivated by completion of the operation before T7, the clock advance from T7 to T0 is prevented. T7 time is then extended until 'arith control' is deactivated.

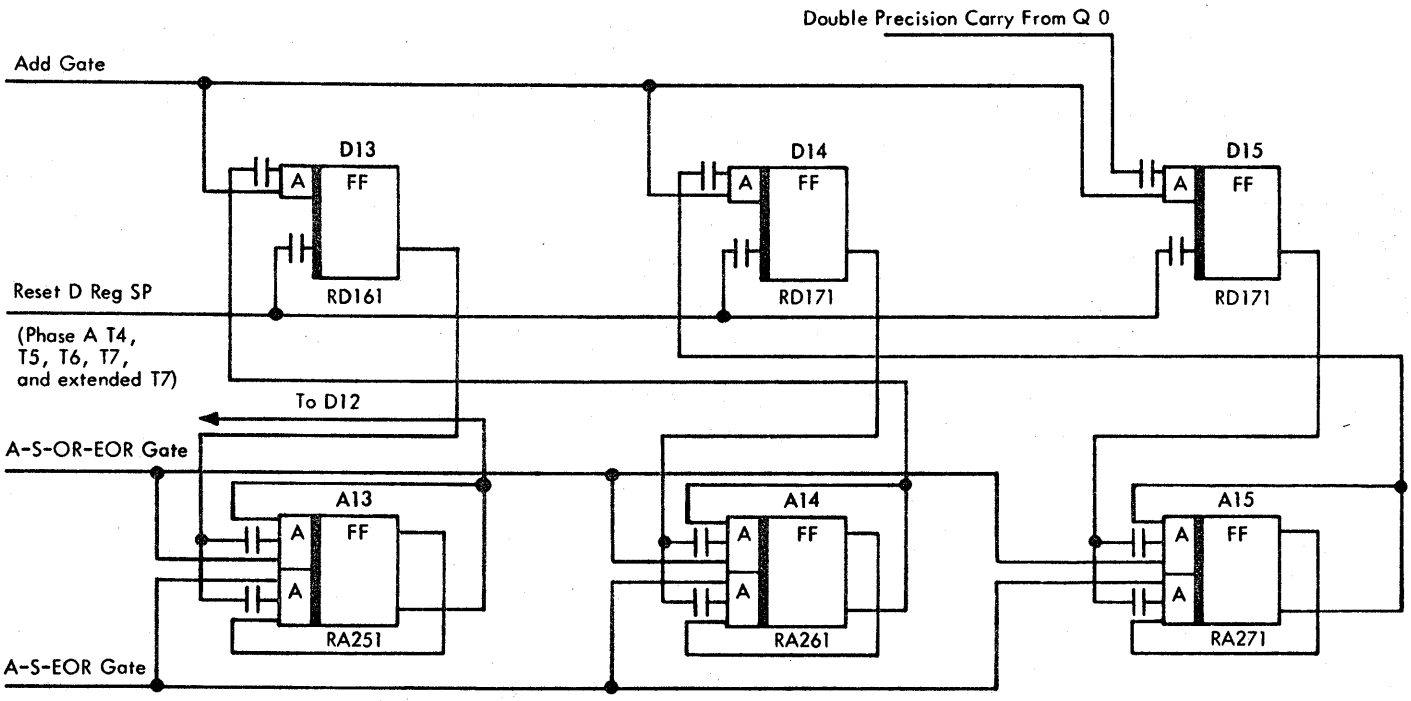
### Add Circuits

- 'Add gate' must be active.
- Phase A of T4 through T7 (or extended T7) provides 'reset D reg SP,' which turns off all D-register FF's.
- Turning off a D-register FF changes the state of the A-register FF in the same bit position.
- Turning off an A-register FF turns on the next higher order D-register FF.

- All D-register FF's off at phase B indicates end of add.

The 'add gate' line must be activated by turning on the 'add' FF with 'arith control' active. The shift at the end of the next phase B pulse (start of T4, phase A) activates the 'reset D reg SP' line. All D register FF's are reset off (Figure 2-20). Resetting a D-register FF which was on (representing a 1) causes a shift at the output. The shift changes the state of the A-register FF in the same bit position. If the change is from on (1-state) to off (0-state) the D-register FF in the next higher order position is turned back on. The D-register FF's which are on at this time represent carries.

At T4, phase B time, the D-register is checked for a 'D reg equal zero' condition (all FF's off). As long as the D-register still contains a 1 in any position, the process of adding at phase A time and checking for 0 at phase B time continues. When the D-register equals 0 at phase B time, the 'arithmetic control trigger' FF is turned off, and the add operation is complete.




Examples:

15 + 1 = 16

| | | |
|---|---|---|
| Starting value in A reg | 001111 | |
| Read from core to D reg | 000001 | |
| Reset D reg sets A reg | 001110 | T4 |
| A reg from 1 to 0 sets D | 000010 | |
| Reset D reg sets A reg | 001100 | T5 |
| A reg from 1 to 0 sets D | 000100 | |
| Reset D reg sets A reg | 001000 | T6 |
| A reg from 1 to 0 sets D | 001000 | |
| Reset D reg sets A reg | 000000 | T7 |
| A reg from 1 to 0 sets D | 010000 | |
| Reset D reg sets A reg | 010000 | Ext T7 |
| No 1 is set into D reg | 000000 | |

End operation, sum in A reg

(-8) + 5 = (-3)

| | | |
|---|---|---|
| A reg | 111000 | |
| D reg | 000101 | |
| | 111101 | T4 |
| | 000000 | |

End-T5,6, and 7 are not used

5 + (-7) = (-2)

| | | |
|---|---|---|
| A reg | 000101 | |
| D reg | 111001 | |
| | 111100 | T4 |
| | 000010 | |
| | 111110 | T5 |
| | 000000 | |

End-T6 and T7 are not used

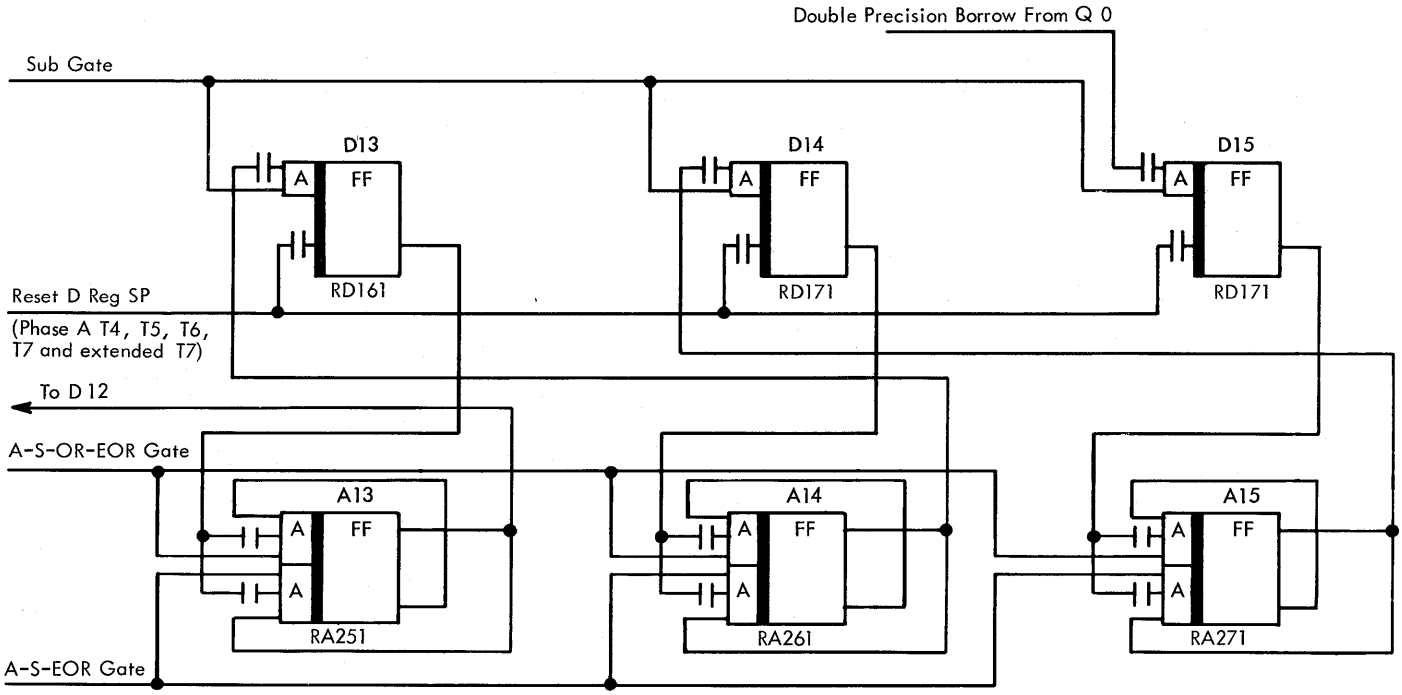
22239A

Figure 2-20. A and D Registers -- Add Operation

Subtract Circuits

- 'Subtract gate' must be active.
- Interconnection between A- and D-registers is changed.
- Turning on an A-register FF turns on the next higher order D-register FF.

When 'arith control' is active with the 'add' FF off, the 'sub gate' line becomes active. The only difference from adding is in the method of turning on a D-register FF (Figure 2-21). The shift resulting from turning on an A-register FF turns on the next higher order FF in the D-register.




Example:

5 - (-7) = 12

| | | |
|---|---|---|
| Starting value in A reg | 000101 | |
| Read from core to D reg | 111001 | |
| Reset D reg sets A reg | 111100 | T4 |
| A reg from 0 to 1 sets D | 110000 | |
| Reset D reg sets A reg | 001100 | T5 |
| No 1 is set into D reg | 000000 | |
| End operation, sum in A reg | | |

22238A

Figure 2-21. A and D Registers -- Subtract Operation

## CYCLE STEAL CIRCUITS

Certain high speed devices request cycle steal cycles, because the devices have data for or need data from CPU core storage. A cycle steal (CS) cycle is inserted between successive T-clock cycles to accomplish the data transfer. The only CPU register used is the B-register; so there is no disruption of program execution. More than one such CS cycle can be inserted if two or more devices request to cycle steal at one time.

Each cycle stealing device is assigned to CS level 0, 1, 2, or 3. A device on CS level 0 is allowed to cycle steal before a device on level 1, 2, or 3, provided the request is received by the CPU before the lower level cycle steal cycle starts. A device on level 1 has the same type of priority over a device on level 2 or 3.

Timing pulses required for the CS cycle are provided by the X-clock.

### X-Clock

- X-clock operation is similar to T-clock operation (Figure 2-2).

- X-clock advance pulses are produced by phase A pulses while 'CS0', 'CS1', 'CS2', or 'CS3' FF is on and 'T7' FF is on.

The eight FF's of the X-clock provide timings for cycle steal cycles and are called 'X0' through 'X7'. When the X-clock is stopped, 'X7' is on, and all others are off. The X-clock can only be started when both 'X7' and 'T7' are on. (The T-clock is stopped.)

If the cycle steal request occurs during a cycle in which T7 time is extended, starting of the X-clock is not delayed. Instead the X-clock starts when the T-clock would have advanced to T0 time had T7 not been extended (Figure 2-22). This overlap of the X-clock cycle and the T-clock cycle is allowable because the only CPU register used for cycle stealing is the B-register. The B-register is not used in the T-clock cycle during extended T7 time.

### Cycle Steal Controls

- Maintenance diagram AA611, sheet 2.

When a cycle steal device requires data transfer, the device activates the 'req CS level' line for the level to which the device is assigned (Figure 2-23). The next T7 and phase B turn on the corresponding 'CS trigger' FF, if no cycle steal cycle is already in progress. The X-clock starts advancing with each following phase A pulse. At X7 and phase B, the 'CS trigger' FF, which is on, turns off if the 'req CS level' line has been deactivated. X-clock advance pulses stop.

If a cycle steal is already in progress when a second 'req CS level' line is activated, the second 'CS trigger' FF does not turn on until X7 and phase B. At that time, one 'CS trigger' FF turns off and another turns on. The X-clock continues advancing and a second CS cycle occurs before the T-clock restarts.

The only time two 'CS trigger' FF's are turned on together is when two 'req CS level' lines are made active during the same T-clock cycle. Only the 'CS level' line for the higher priority is active during the first CS cycle. (The highest priority is level 0; next highest is level 1; and lowest priority is level 3.) Usually the device adapter circuits deactivate the 'req CS level' line at X6 of the CS cycle for that device. Therefore a phase B pulse at X7 can turn off the higher priority 'CS trigger' FF. However, the lower priority level request is still active, and a CS cycle occurs for that level.

At X7 and phase B of the cycle in which the last 'req CS level' line is deactivated, the last 'CS trigger' FF turns off. No more X-clock advances can occur until the next time a request line is activated by a device.

## INTERRUPT CIRCUITS

- Any I/O device can request an interrupt.

- Interrupt circuits force a BSI operation code, long format, calling for indirect addressing (bit 8 = 1).

- Interrupt circuits also force the indirect address (in the interrupt address table) for the level which is requested.

- Interrupt circuits force a branch to the highest (priority) level request which is active.

An interrupt is requested by any I/O device when it has a bit or several bits of status information to

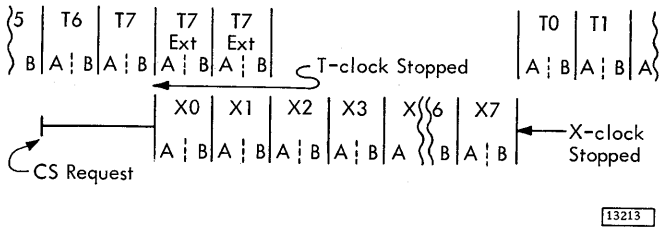



Figure 2-22. Cycle Steal and Extended T7

transfer to the CPU. Also, an interrupt is requested by any of the 'direct program control' group of I/O devices when the device requires a data transfer. In either case the actual transfer is accomplished by the execution of an XIO instruction within an 'interrupt servicing subroutine.'

The CPU interrupt circuits have two primary functions:

1. Force a BSI long format instruction calling for indirect addressing when a device requests an interrupt. Also, force the indirect address (an address in the interrupt address table).

2. Establish a priority for servicing interrupt requests when two or more requests are active simultaneously.

Execution of the forced branch directs program flow to the start of the interrupt servicing subroutine for the interrupt level requested. The content of the subroutine is the programmer's responsibility. In general, the subroutine contains instructions to transfer data words or to transfer and analyze bits of status information.

Interrupt Forced 'BSI'

- When 'gate interrupt' and 'request interrupt level (x)' are active, a 'set interrupt' pulse turns on 'level (x)' FF if it is off. Turning on 'level (x)' FF then turns on 'request (x)' FF.

- During I1, instruction word is set in B-register. 1's are set in bit 1, bit 5, and bit 8 positions.

- During I2, address word (indirect address, from interrupt address table) is set in B-register.

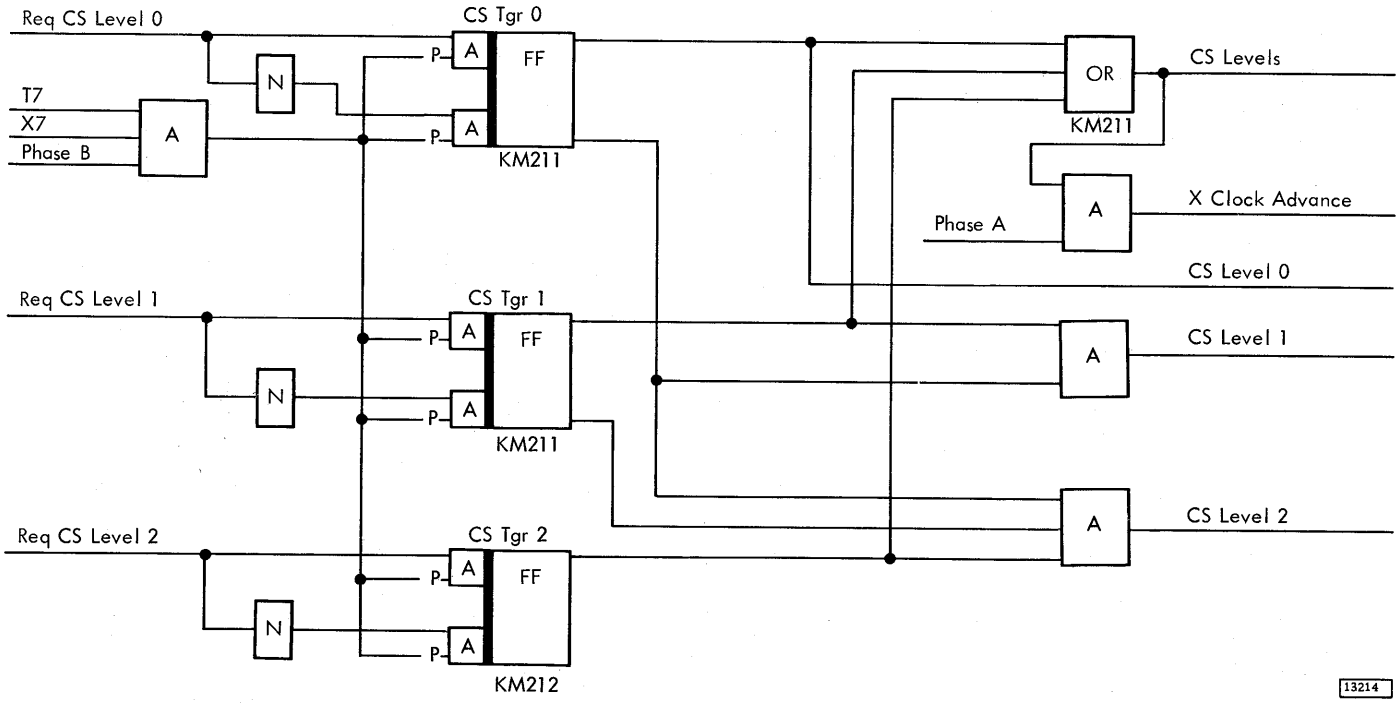



Figure 2-23. Cycle Steal Controls

- 'Request (x)' FF is turned off at I2 T6 time, and execution of the forced BSI instruction is standard.
- 'Level (x)' FF is turned off when a branch out of the interrupt servicing subroutine occurs as a result of a BOSC instruction.
- Maintenance diagram AA611, sheet 1.

When a device requests an interrupt, the CPU interrupt circuits force a 'BSI indirect' instruction during the next two I-cycles. Core storage use is prevented; so no instruction is read from core storage. During the I1 cycle, bits 1, 5, and 8 are set to 1's in the B-register and transferred to the operation, format, and modifier registers. The instruction word thus formed is:

0100 0100 1000 0000

The operation code is BSI, long format is specified, and indirect addressing is called for. During the I2 cycle, an address from the interrupt address table is forced. The address depends on the interrupt level requested. Execution of the forced instruction is the same as execution of any BSI instruction (Figure 2-24).

Figure 2-25 shows the interrupt circuits (level 0 is used as an example) which force the BSI instruction. The interrupt delay CE switch must be off to allow interrupts. 'CCC count 0' is active near the end of the last cycle required for any instruction. Phase B, with both 'T7' and 'X7' on, activates 'set interrupt SP' and turns on 'level 0' FF, which turns on 'request 0' FF. 'Request 0' stays on until T6 of the I2 cycle and is ANDed with 'level 0' to gate circuits which set the B-register. The 'interrupt request' line inhibits storage use so that the B-register can be set from the 'I/O bit' lines instead of from core storage. The 'run' FF is turned on, if it is off, so that the T-clock is started. Thus an interrupt can occur while the CPU is in a 'wait' state or has been stopped by the program stop key. Incrementing of the IAR is blocked so that the correct address can be stored by the BSI instruction. The address to be stored is the address of the instruction which would have been sensed and executed next had the interrupt not occurred.

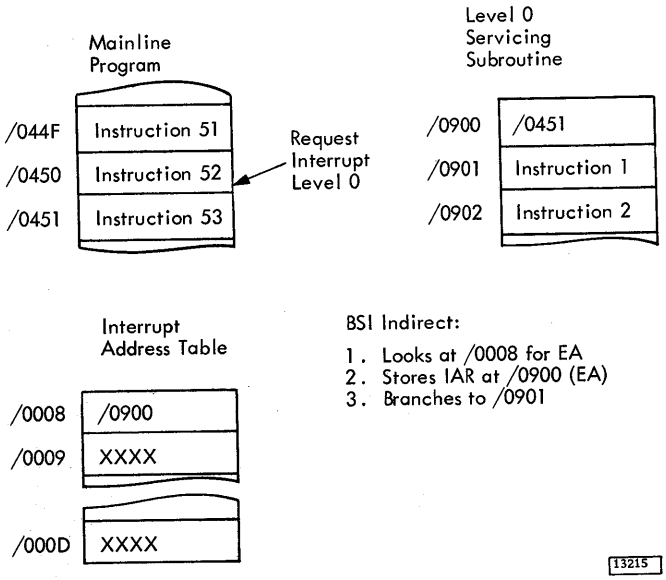



Figure 2-24. Interrupt Forced BSI

Early in the subroutine, an instruction must be given to reset the condition which activated the request. The fact that the request has been recognized by the CPU is indicated by the fact that 'level 0' FF is on.

After the CPU has completed execution of the interrupt level 0 servicing subroutine, the 'level 0' FF must be turned off. A BOSC instruction, which is a BSC with modifier bit 9 set to a 1, is used as the last instruction in the subroutine. Only when conditions cause a branch is the 'branch out' line activated. At T7 time of that cycle, the 'level 0' FF turns off.

## Interrupt Priority

- Each condition in an I/O device which can cause an interrupt request is assigned to one of the six interrupt levels.
- Levels 0 through 5 have the highest through the lowest priorities, respectively, for being serviced.
- The highest level of interrupt for which there is a request is serviced first.
- A higher level request can interrupt a lower level servicing subroutine.
- Return to the mainline program from a subroutine occurs only when no interrupt request is active.

Each interrupt request is assigned a level according to the promptness with which the request must be serviced. The interrupt priority circuits (Figure 2-26) assure that the highest level request is

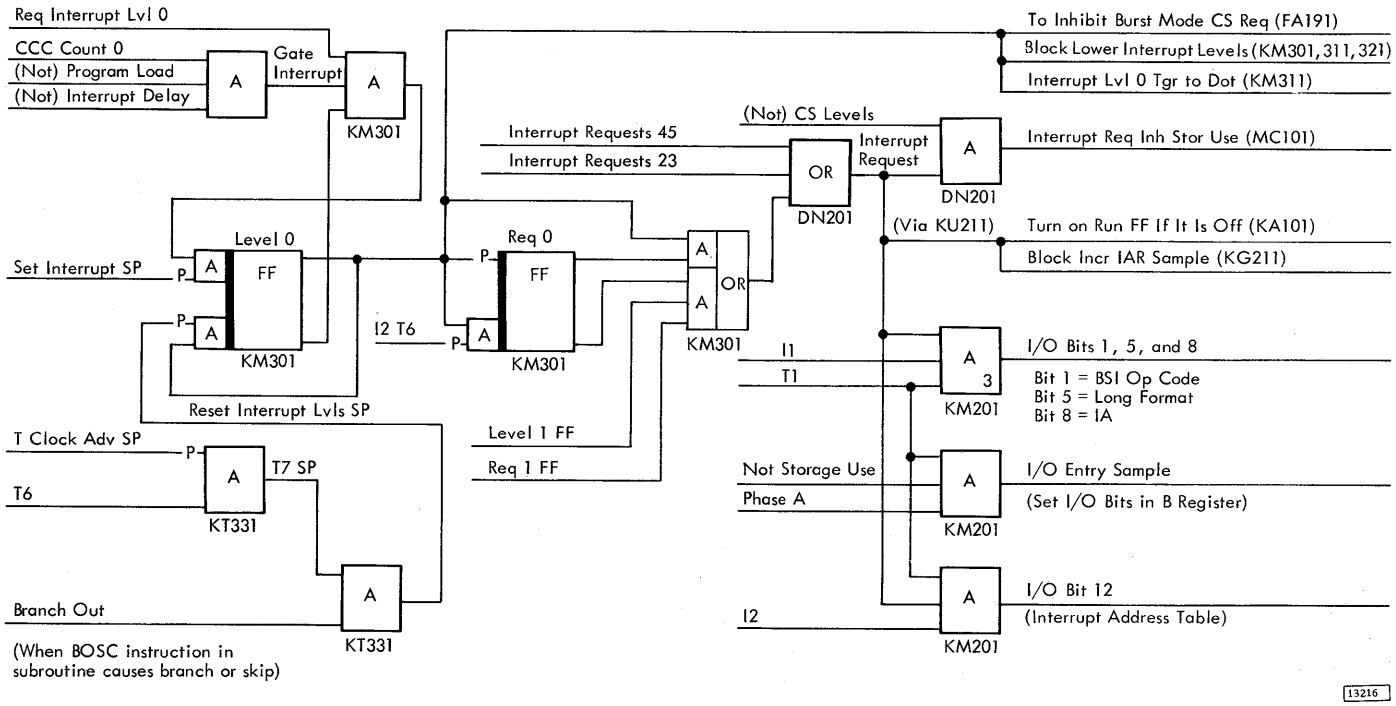



Figure 2-25. Interrupt Logic for Level 0

serviced first when multiple requests are active. The highest level is level 0; levels 1 to 4 have successively lower priorities; and the lowest level is level 5. Figure 2-26 shows only levels 0 through 3, but it is sufficient to show circuit operation. Operation is described for the following sequences of interrupt requests.

Summaries of Sequences

Sequence 1: Multiple requests activated at the same time.

Sequence 2: Higher level request activated while lower level request is being serviced.

Sequence 3: Lower level request activated while higher level request is being serviced.

Detailed Descriptions of Sequences

Sequence 1: Assume that both 'request interrupt level 0' and 'request interrupt level 1' are activated between the start of I1 and the end of the last cycle required for that instruction. Effectively, the requests are occurring at the same time (Figure 2-27).

Both 'level 0' FF and 'level 1' FF are turned on by the 'set interrupt SP.' When these two FF's are turned on, they turn on 'request 0' FF and 'request 1' FF, respectively. However, neither the 'interrupt request 1' line nor the 'interrupt level 1' line can be activated because both are blocked by 'level 0' FF being on. Only 'interrupt request 0' is activated, and the forced branch is to the highest level (level 0) servicing subroutine. 'Request 0' is turned off at I2T6 time because the 'level 0' FF is on. 'Level 0' FF remains on until the BOSC instruction at the end of level 0 subroutine causes a branch out. In the meantime, 'request interrupt level 0' should have been deactivated by an instruction in the subroutine.

'Level 0' FF turns off at T7 phase A time. Note that it could turn back on at phase B time if 'request interrupt level 0' is active because:

1. No instruction in the subroutine reset the condition in the device which activated 'request interrupt level 0.'
2. A second condition has reactivated 'request interrupt level 0.'

Either condition forces another BSI to level 0 subroutine in the same manner as the first time.

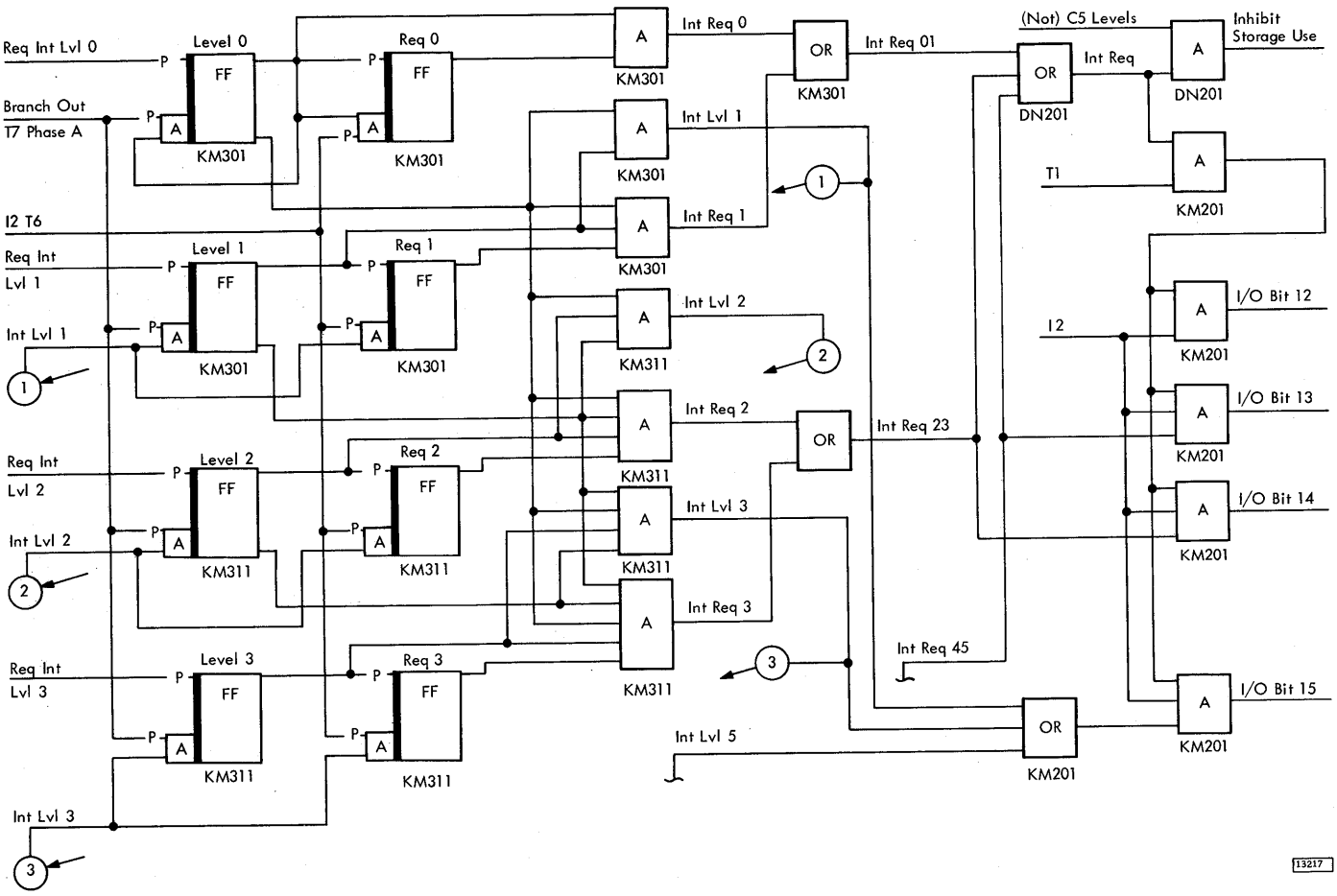



Figure 2-26. Interrupt Priority

If 'level 0' is not turned on again, 'level 1, and 'request 1' FF's can activate the 'interrupt level 1' and 'interrupt request 1' lines. Another forced BSI now directs program execution, via the interrupt address table, to level 1 interrupt servicing subroutine. At I2 and T6 time the 'request 1' FF is turned off, but the 'level 1' FF remains on. Provided no further request for level 0 is activated, execution of the level 1 servicing subroutine continues to completion. The BOSC at the end of level 1 subroutine turns off the 'level 1' FF. Unless another request is active, the mainline program is reentered. Reentry is made at the address which was set into IAR by the BOSC instruction.

Figure 2-28 shows the program flow of this sequence of interrupt requests. The first forced BSI uses the indirect address /0008 from the interrupt address table to obtain an EA of /0500. The address from IAR (/1004) is stored at /0500, and a branch occurs to /0501. During execution of the BOSC, which is usually indirect as shown, /1004 is set back into IAR. The second forced BSI follows immediately and uses indirect address /0009 from the interrupt address table. The address /1004 is set into core storage at /0600, and the branch is to /0601. Another indirect BOSC at the end of level 1 subroutine sets /1004 back into IAR, causing a return to the mainline program.

Sequence 2: Assume that the CPU is executing the servicing subroutine for interrupt level 1 when an interrupt condition activates the 'request interrupt level 0' line (Figure 2-29). The next time 'gate interrupt' is active, 'set interrupt SP' turns on the 'level 0' FF, blocking the 'interrupt level 1' line. 'Level 0' FF turning on also turns on the 'request 0' FF and activates the 'interrupt request 0' line. The resulting forced 'BSI indirect' causes a branch to the level 0 servicing subroutine via the interrupt address table. Although the 'level 1' FF remains on, it has no further effect until the 'level 0' is

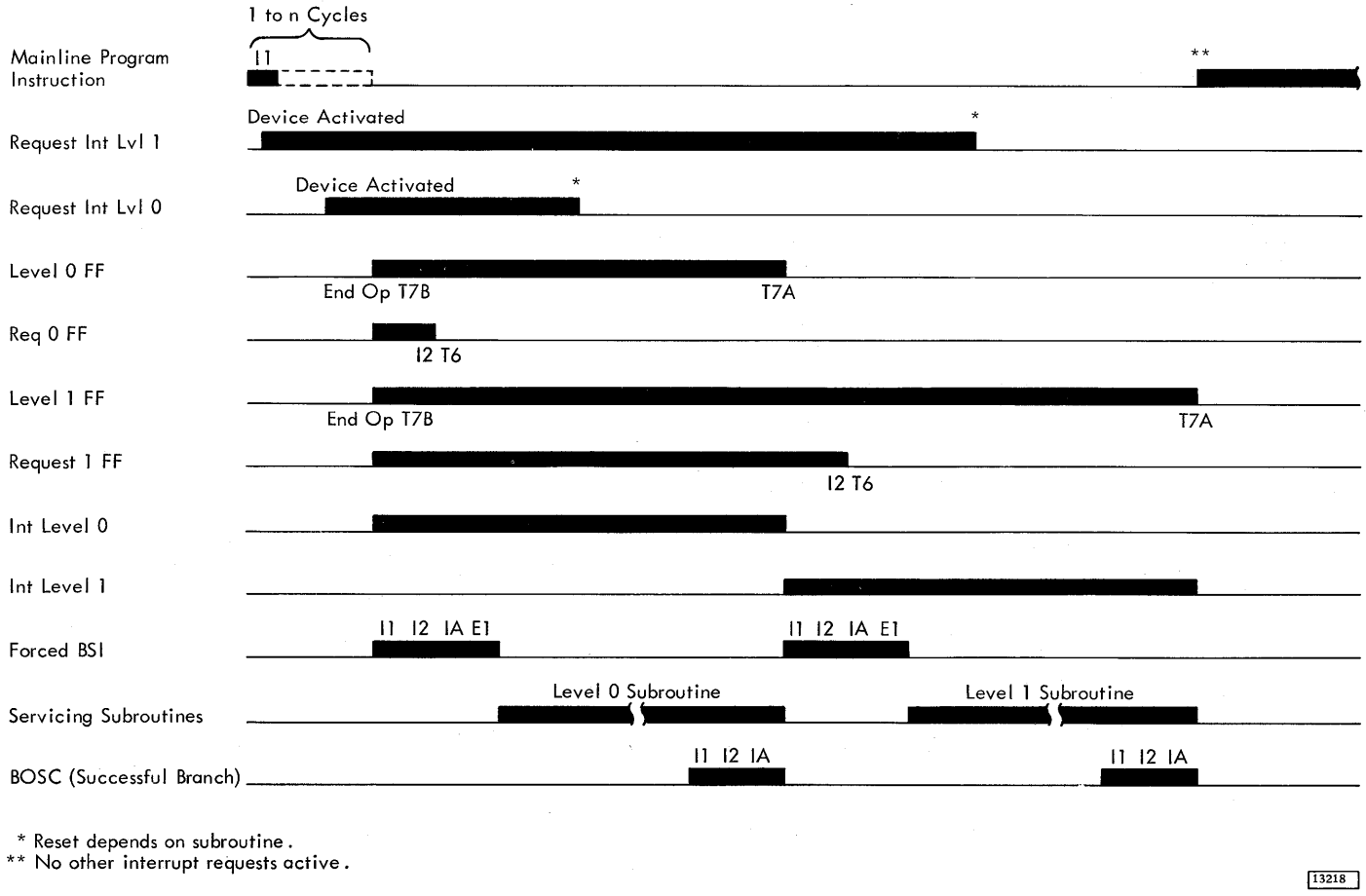



Figure 2-27. Multiple Interrupt Requests at Same Time

turned off. Thus the level 0 servicing subroutine must be completed before the level 1 subroutine can be reentered.

Once the interrupt 0 request has been recognized by the CPU, the program (interrupt servicing subroutine) must reset the condition in the device which activated the 'request interrupt level 0.' At the end of the level 0 subroutine, if the 'request interrupt level 0' has not been reactivated, the BOSC causes reentry into the level 1 subroutine. On completion of the level 1 subroutine, provided no more interrupt requests have been recognized, the BOSC causes return to the mainline program. Any lower level requests (2 through 5) would be serviced before return to the mainline program.

Figure 2-28 (sequence 2) shows the program flow for this sequence of interrupt requests. The first BSI, forced by the 'request interrupt level 1,' uses interrupt address table (IAT) address /0009 to obtain an EA of /0600. The mainline return address /1006 from IAR is set into /0600, and the program branches to /0601. When 'request interrupt level 0' forces the second BSI, IAR contains /0606. Level 0 selects the indirect address /0008. Thus the BSI sets /0606 in address /0500 and causes a branch to /0501. When level 0 subroutine has been completed, the 'BOSC indirect' sets /0606 back into IAR. The level 1 subroutine is reentered where the level 0 request interrupted. The BOSC at the end of level 1 subroutine sets the address /1006 back into IAR. If no further interrupt requests are to be serviced, mainline program execution is resumed at /1006.

Sequence 3: Assume that the CPU is executing the servicing subroutine for interrupt level 0 when a device interrupt condition activates the 'request interrupt level 1' line. The next time 'gate interrupt' is active, the 'set interrupt SP' can turn on 'level 1' FF, which then turns on the 'request 1' FF. However, 'interrupt level 1' cannot be activated because the 'level 0' FF remains on. The sequence followed in branching to and executing the level 1 servicing

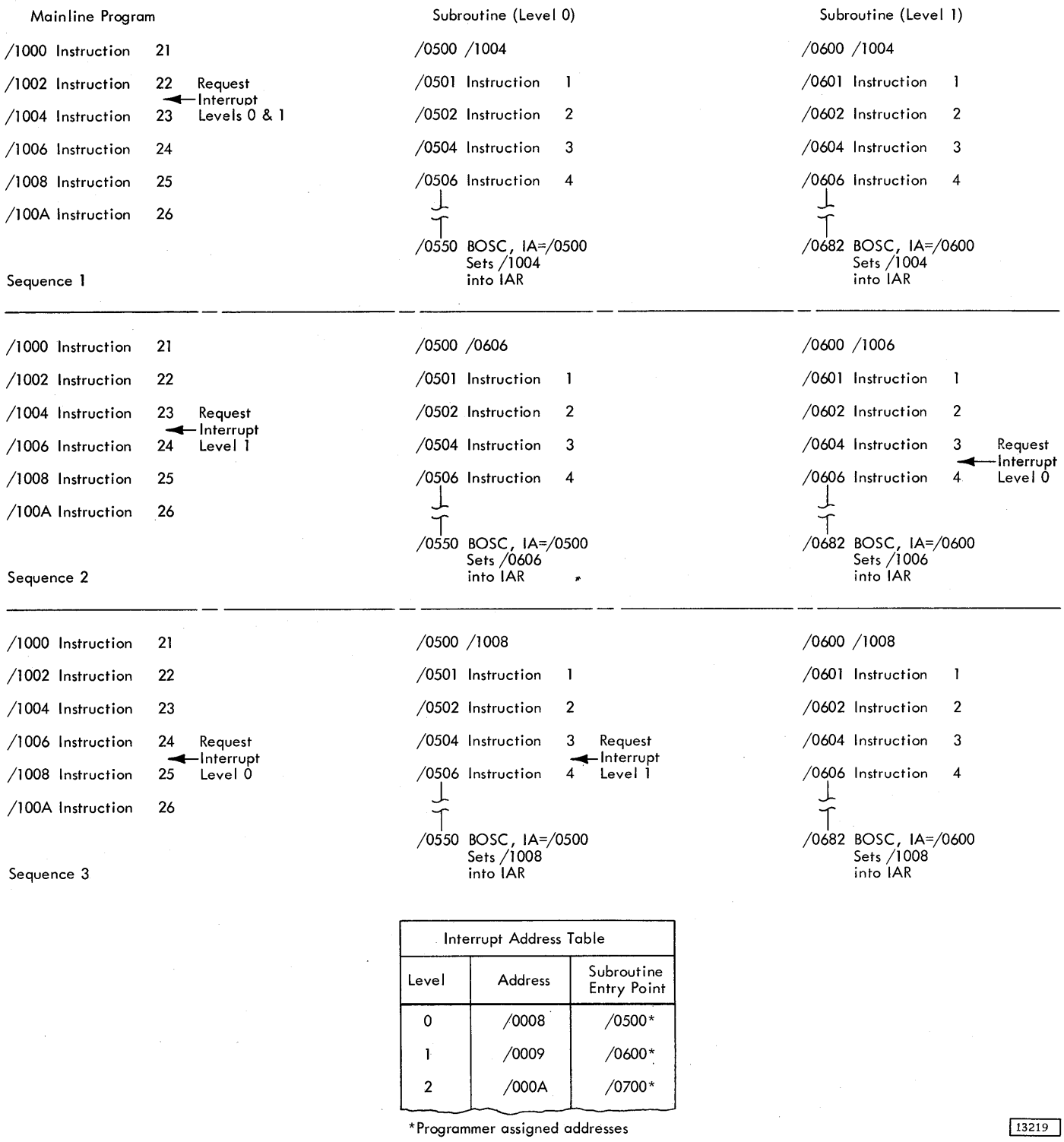



Figure 2-28. Interrupt Sequences

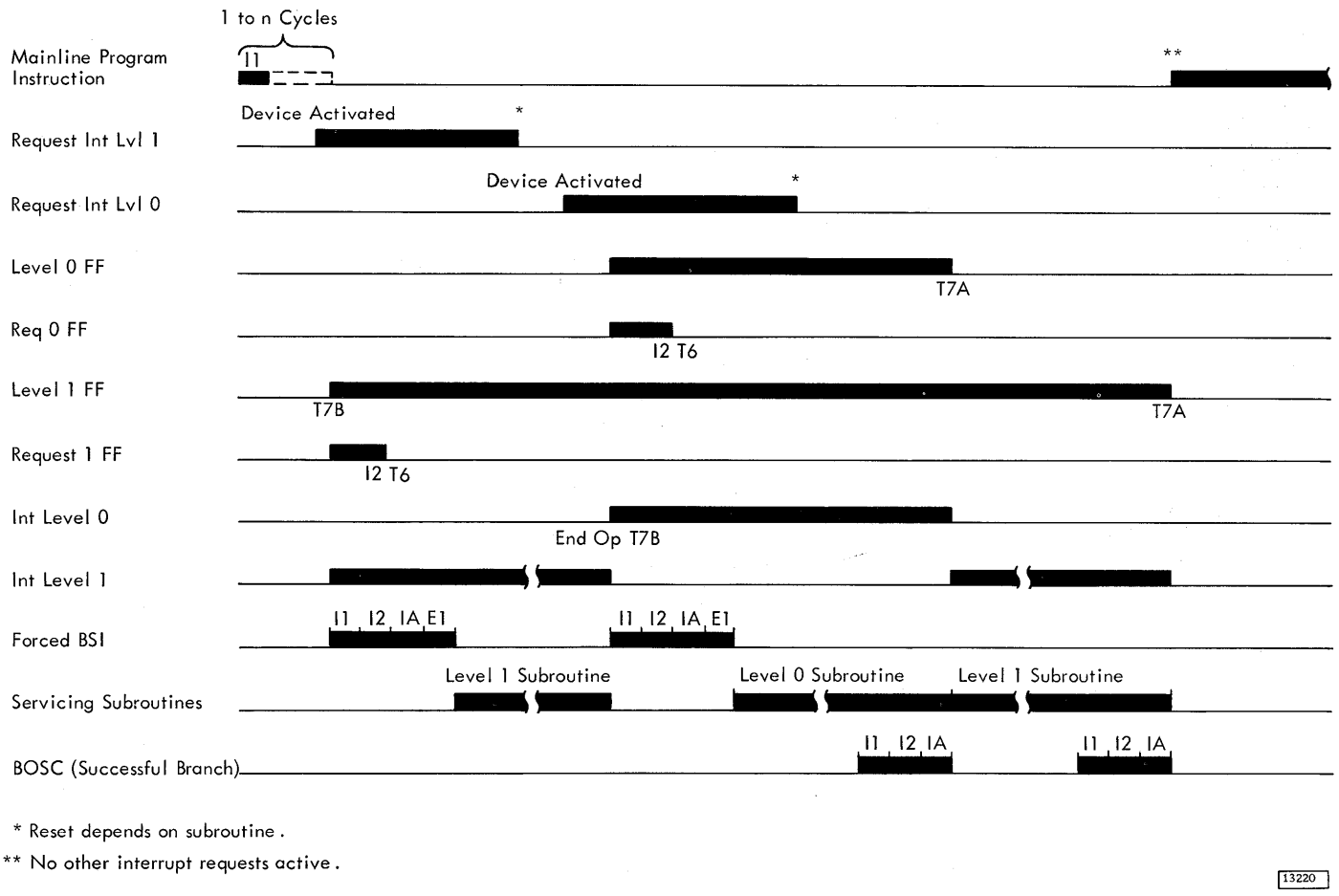



Figure 2-29. Higher Level Interrupting Lower Level

subroutine is the same as when multiple requests occurred at the same time (Figure 2-27).

Note in Figure 2-28 that program flow for this sequence of interrupt requests is the same as that for sequence 1. The request for level 1 cannot interrupt the level 0 servicing subroutine. Execution of the level 1 subroutine follows execution of level 0 subroutine, with no intervening return to the main-line program.

## CONSOLE KEYBOARD

### Keyboard Mechanics

- Pressing keyboard keys closes electrical circuits.
- Some keys also perform a mechanical function.
- The keyboard interlocks allow only one key to operate at a time.
- The two basic units in the keyboard are the permutation unit and the key components unit.

Figure 2-30 shows the combination keyboard with its cover removed and the keyboard disassembled into its two major units (key components unit and permutation unit). The key components unit contains all the keys. The numeric key and restore keyboard key perform only electrical functions by operating switches when they are pressed. (Early machines had an alphabetic key which operated similarly.) All other keys operate mechanical linkages when they are pressed. The keys are interlocked so that only one can be pressed at a time. Operating each mechanical linkage closes a contact or contacts in an electrical encoding network. Figure 2-31 shows

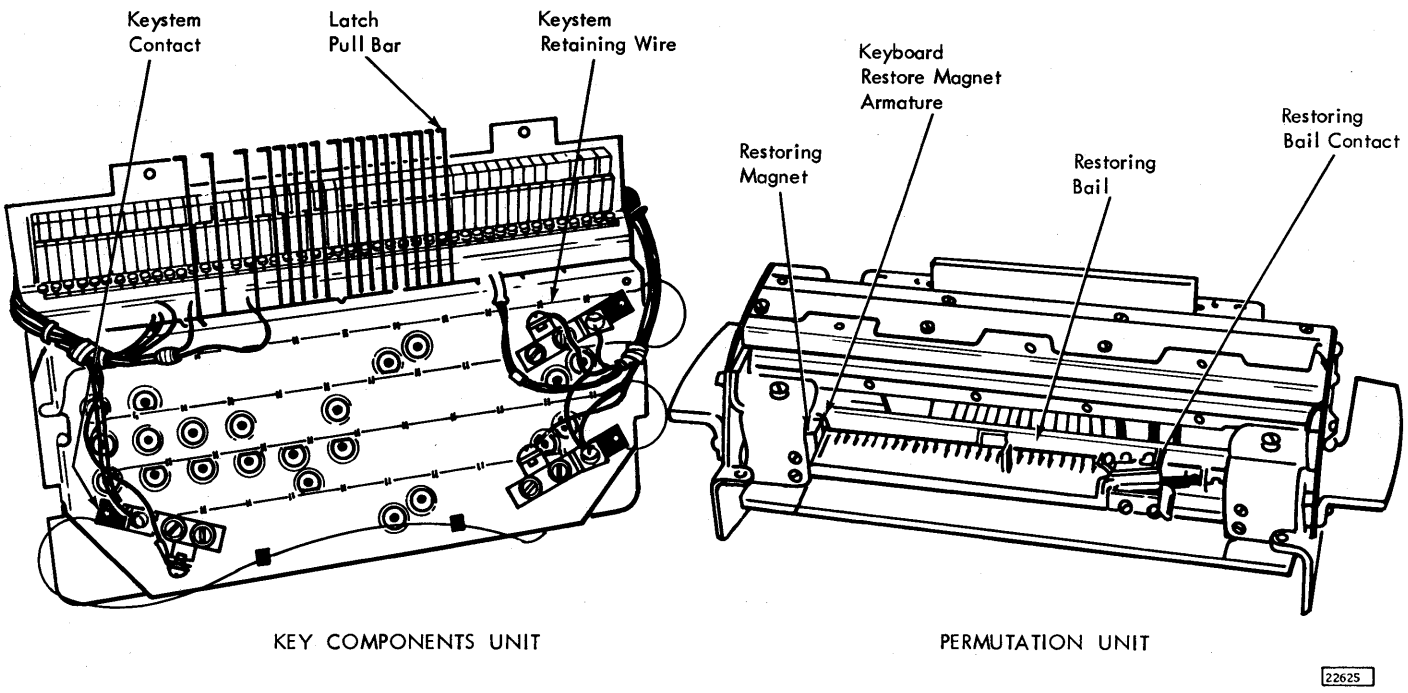



Figure 2-30. Keyboard Units

the latch pull bar, bellcrank, latch, and permutation bar comprising the mechanical linkage for one key.

The key button rubber bumper, a rubber washer under the key button, is associated with the keystem. The amount of downward travel on the key when it is pressed is partly determined by the thickness of the rubber bumper. As a key is pressed, the bellcrank pivots and moves the latch pull bar toward the key stem. The hooked portion of the latch pull bar rests in a notch in the top of the latch. Moving the latch pull bar toward the key stem pulls the latch off the latch bar. The permutation bar is then free to move downward. Near the end of the operation the keystem is restored by its spring.

The mechanical operation upon pressing a key follows (Figures 2-31, 2-32, and 2-33):

1. Keystem bellcrank moves its latch pull bar forward.
2. Latch assembly drops off latch bar.
3. Permutation bar moves downward.
4. Individual keystem spring restores key and pull bar to normal.
5. Separate flat springs hold each pull bar against its latch assembly and make sure it relatches in notch of latch.

As the permutation bar moves downward, it pivots any contact bails it touches, thus closing the corresponding bail contacts. The permutation bar also transfers a latch contact if one is mounted on the lower frame directly beneath the permutation bar.

Bail and Latch Contacts

- These contacts are operated by permutation bars or contact bails to complete circuits in the encoding network.
- Any of the 15 contact bails, spanning the width of the keyboard, can be pivoted by a permutation bar.
- Each bail, when pivoted, transfers a bail contact.
- A latch contact is transferred by its permutation bar.

A contact bail is pivoted by a tab attached to the bail and resting in a notch that is cut in the edge of a permutation bar. A permutation bar has fifteen

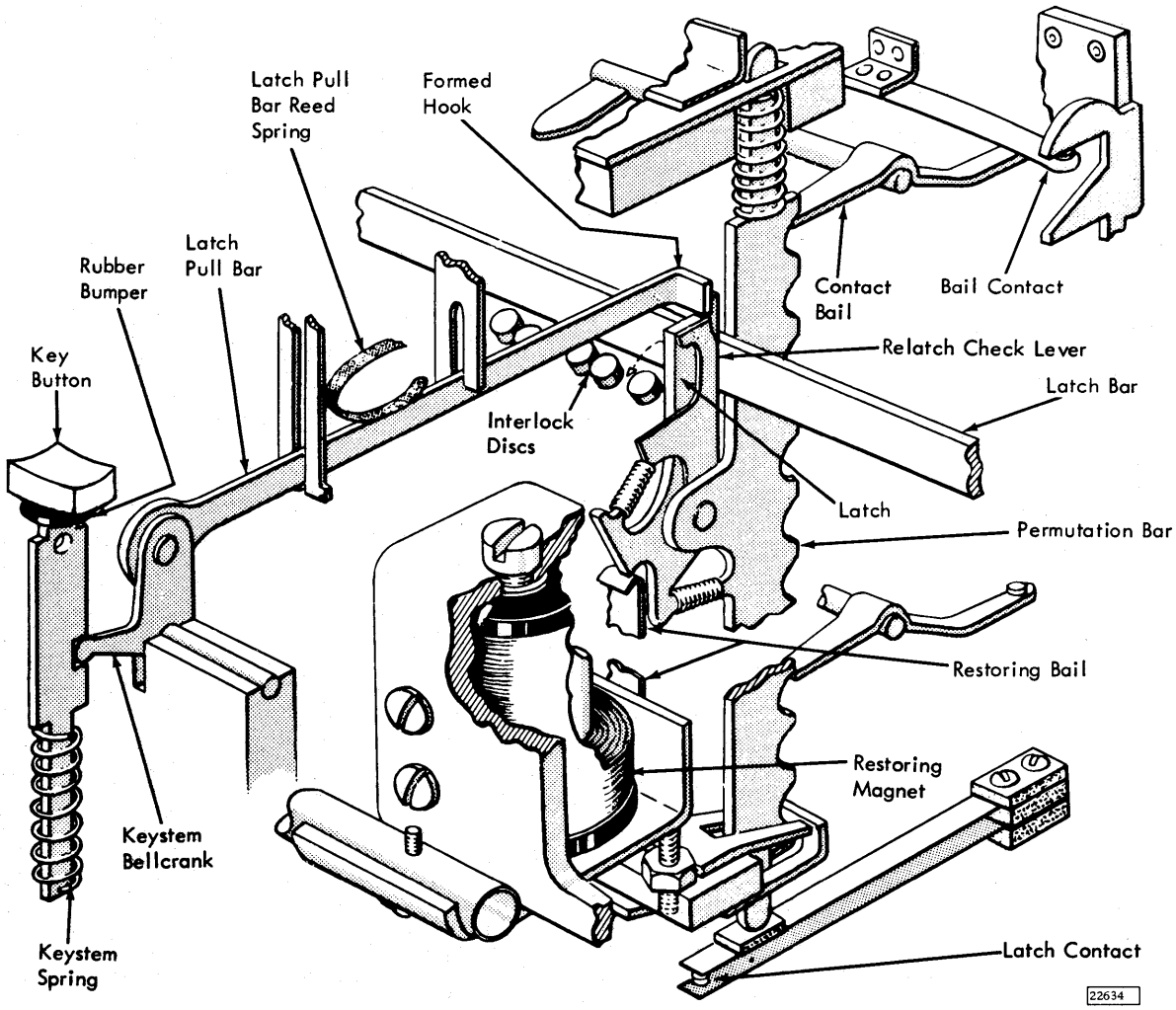



Figure 2-31. Key Position

notches cut in its edge for operating any of the possible 15 contact bails (Figures 2-32 and 2-33).

A keyboard reference chart, on ALD ZK121, shows the combination of latch contacts and bail contacts needed to encode any character. Odd-numbered bail contacts are on the right side of the keyboard, and even-numbered bail contacts are on the left side of the keyboard (as viewed from front of keyboard).

The chart in Figure 2-34 identifies the keystem numbering of the combination keyboards. Pressing any one key can never transfer more than one latch contact, but more than one bail contact may be transferred.

Keyboard Restore

- Keyboard restore consists of relatching any unlatched permutation bars.
- The restoring bail restores the keyboard when the restoring magnet is energized.

When the latch was pulled off the latch bar (Figure 2-35, inset), the relatch check lever pivoted toward the latch bar. The pull bar disengaged from the latch because it could not follow the latch downward. The pull bar cannot engage in the slot in the latch again until the latch and pull bar have both been

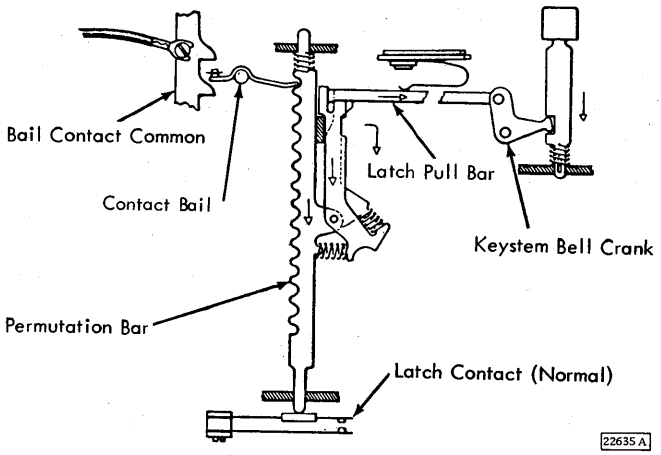



Figure 2-32. Permutation Bar and Keyboard Latch (Latched)

restored. A key can be pressed and held pressed while the latch is pulled off the latch bar and restored to the latch bar. The pull bar reengages in the latch only if the key is released.

Energizing the restoring magnets operates the restoring bail upward, pushing against the latch which has been pulled off the latch bar. At this time, the keyboard restoring bail contact (normally closed) is operated to open the circuit to the latch and bail contacts. When the permutation bar latch relatches on the latch bar, restore action is complete. All latch and bail contacts have returned to their normal conditions. The slot in the latch has moved to a position clear of the relatch check lever.

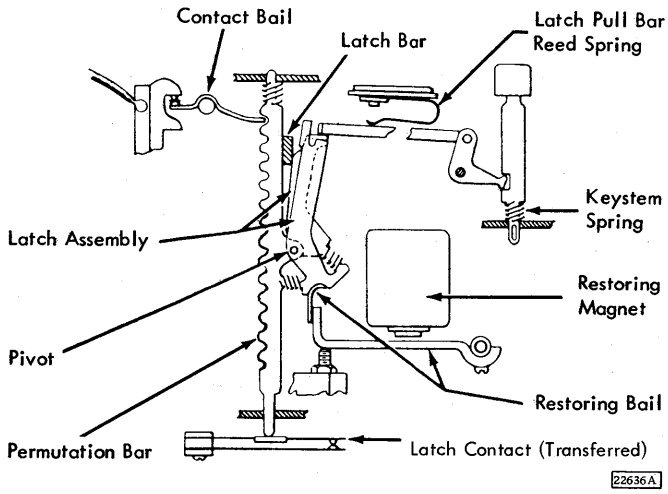



Figure 2-33. Permutation Bar and Keyboard Latch (Unlatched)

The latch pull bar can drop into the slot in the latch if the key stem is in its normal position.

## Keyboard Interlocks

- Interlock disks prevent more than one latch at a time from being pulled off the latch bar.
- Consequently, only one key stem can be operated at one time.

Figure 2-36 shows how the interlock disks prevent unlatching of more than one permutation bar at one time. A permutation bar can only be unlatched when the latch pull bar can pull its latch off the latch bar. Only a part of the interlock assembly is used to show the operation.

Pressing a key results in moving latch B into the position shown and unlatching its permutation bar. Before latch B is restored, the key for latch D is pressed. The interlock disks prevent latch D, or any other latch, from moving clear of the latch bar. If no other latch can be pulled off the latch bar, no other key can be fully operated. Only when the first latch has been restored on the latch bar can another latch be pulled from the latch bar.

## Keyboard Electrical Functions

The keyboard reference chart, on page ZK121 of the ALD's, shows all the combinations of contacts, the card code, and the symbol associated with each keystem. Page ZK101 of the ALD's shows the wiring of the keyboard.

The control switches and indicators, which are mounted at the sides of the keyboard, are described in "Chapter 6, Console and Maintenance Features."

## CONSOLE PRINTER

The console printer used on the 1131 CPU is a SELECTRIC® I/O Keyboardless Printer. The theory of operation and maintenance of that unit are described in FE manuals as follows:

| | |
|---|---|
| FETO | 225-3353 |
| FEMM | 225-3207 |

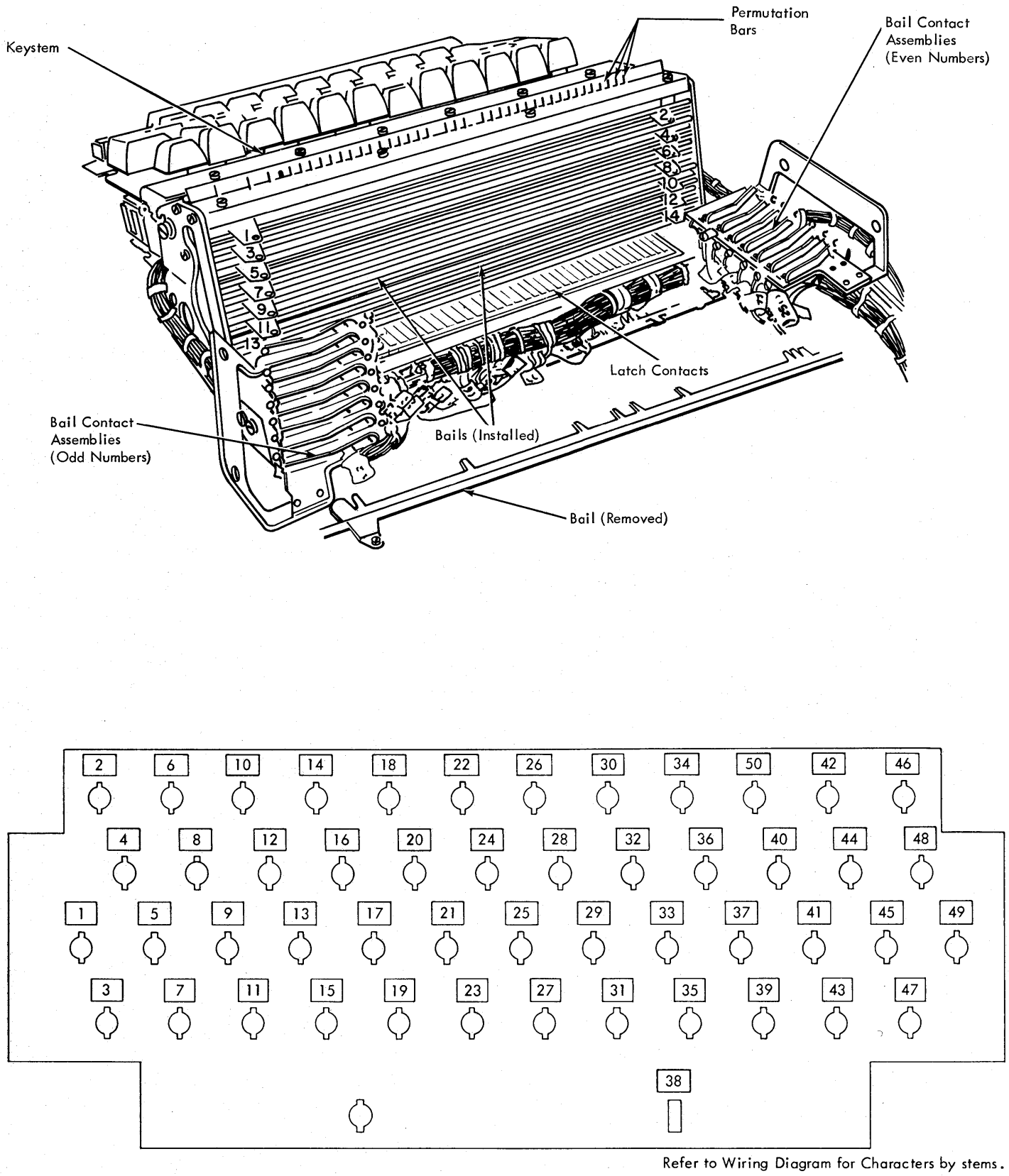



Figure 2-34. Permutation Unit and Keystem Numbering Chart

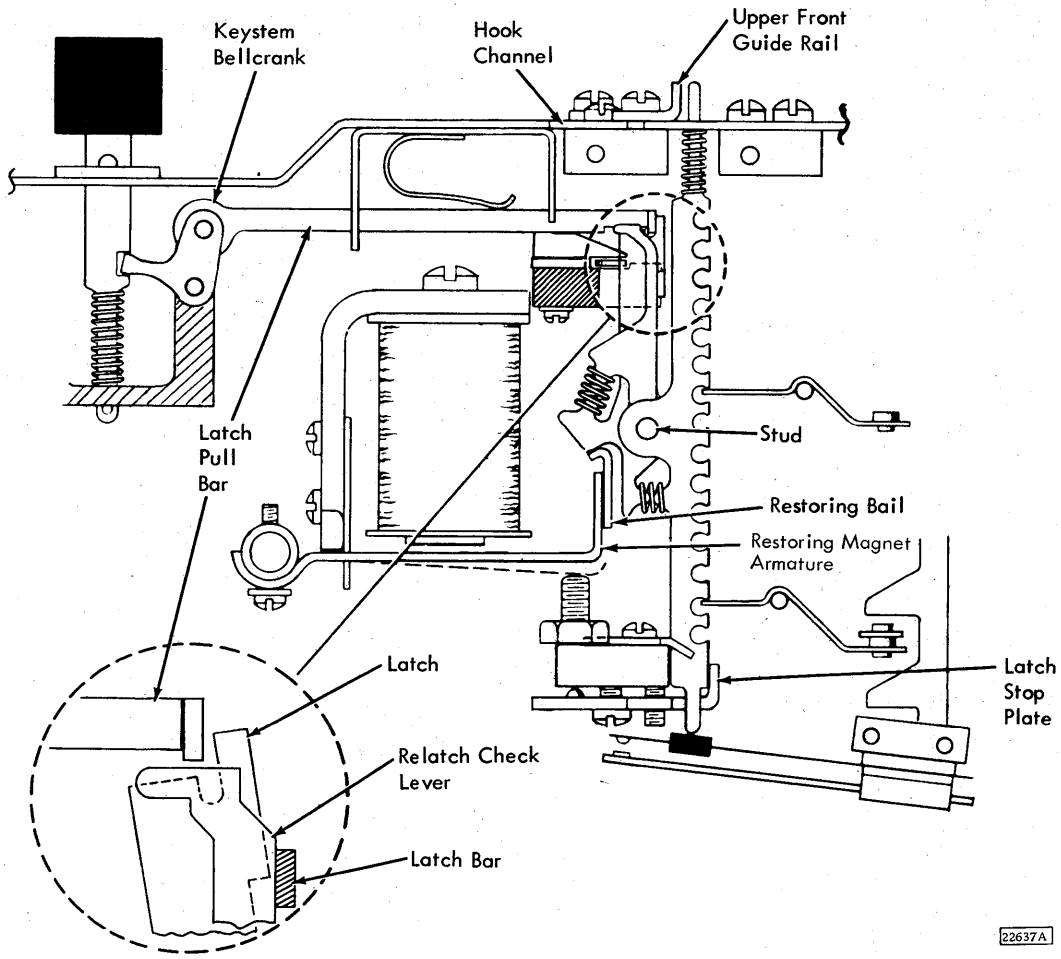



Figure 2-35. Key Position (Latched and Unlatched)

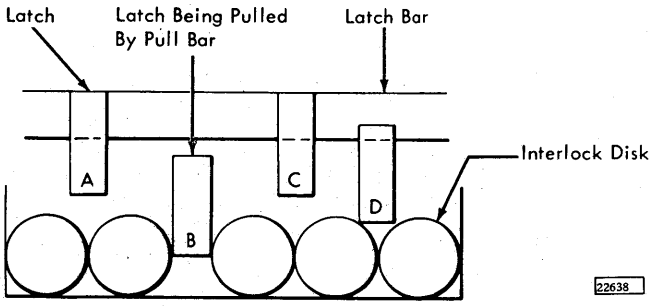



Figure 2-36. Interlock Disks

## INSTRUCTION CYCLES

- Instruction cycles interpret the instruction and develop an effective address (EA).
- There are six instruction types (Figure 3-1);
  Single-word format, without indexing.
  Single-word format, with indexing.
  Double-word format, without indexing, without indirect addressing.
  Double-word format, with indexing, without indirect addressing.
  Double-word format, without indexing, with indirect addressing.
  Double-word format, with indexing, with indirect addressing.
- Format (bit 5) position of instruction word indicates whether the instruction is one or two words in length.
- Tag (bit 6 and bit 7) positions indicate which index register to use when the instruction calls for the use of an index register.
- IA (bit 8) position indicates whether or not indirect addressing is used.
- Instruction cycle 1 (I1) occurs for all instructions.
- Instruction cycle 2 (I2) occurs for all double-word format (F = 1) instructions.
- Indirect addressing cycle occurs for instructions using indirect addressing (IA = 1).
- Indexing cycle occurs when tag = 1, 2, or 3 (Tag≠0) except when instruction is LDX, STX, or MDX.
- Reference maintenance diagram: AA6 01.

Instruction cycles (I-cycles) read the instruction from core storage, store the instruction in control registers, and decode the control registers to specify an operation. Also, when data is to be used in the operation, I-cycles either obtain the address of the data from core storage or compute the address of the data. This address is called the effective address (EA). See Figure 3-2. The sequence of I-cycles that occurs is controlled by the cycle timer. The types of I-cycles in addition to I1 (I2, IX, and IA) that are taken for any instruction depend on the bits of the instruction word. When both indexing and indirect addressing are called for, the indexing cycle occurs first.

Normal turn-on time and normal turn-off time of all I-cycle FF's is T0. A slight overlap of I-cycle FF's may occur at T0. This overlap extends from the turn-on of one FF to the turn-off of the previous FF. For example, 'I1'

| Type Of Instruction | Cycle Required | | | |
|---|---|---|---|---|
| | I-1 | I-2 | IX | IA |
| Single - Word | x | | | |
| Single - Word With Indexing | x | | x | |
| Double - Word | x | x | | |
| Double - Word with Indexing | x | x | x | |
| Double - Word With Indirect Addressing | x | x | | x |
| Double - Word With Indexing And Indirect Addressing | x | x | x | x |

22211A

Figure 3-1. Instruction Cycles

| Tag | F=0 (Direct Addressing) | F=1, IA=0 (Direct Addressing) | F=1, IA=1 (Indirect Addressing) |
|---|---|---|---|
| T=00 | EA=Disp+IAR | EA=Add | EA=C/Add |
| T=01 | EA=Disp+XR1 | EA=Add+XR1 | EA=C/(Add+XR1) |
| T=10 | EA=Disp+XR2 | EA=Add+XR2 | EA=C/(Add+XR2) |
| T=11 | EA=Disp+XR3 | EA=Add+XR3 | EA=C/(Add+XR3) |

Disp=Contents of Displacement field of instruction.
Add =Contents of Address field of instruction.
C=Contents of Location specified by Add or Add+XR.

Note: This table does not apply to the MDX, LDX, STX, LDS, Shift or Wait instructions.

20111D

Figure 3-2. Effective Address Generation

conditions the circuits required to turn on 'I2.' As 'I2' is being turned on, 'I1' is being turned off. An overlap of the two FF's can occur, depending upon the timing of the internal circuits of the FF.

Only functions common to all I-cycle FF's are performed during T0. The turn-on of successive FF's is interlocked by control gates to prevent more than one cycle ('I1,' 'I2,' 'IX,' or 'IA') FF from being on simultaneously during instruction time.

INSTRUCTION CYCLE 1 (I1)

- I1 cycle occurs for every instruction.
- Instruction word is read from core storage.
- Bit positions 0-7 are set into the operation-format-tag register.
- Bit positions 8 and 9 are gated to modifier FF's.
- Bit positions 8-15 are gated to set the D-register.
- Reference maintenance diagram: AA601.

The I1 cycle reads the instruction word from core storage at the address specified by the M-register at read time. The instruction word is set into the B-register, distributed from the B-register to the various control registers (Figure 3-3), and decoded to indicate the operation to be performed.

Further functions of the I1 cycle depend upon the decoding of the instruction word (operation code, format, tag, IA, modifiers, and displacement).

| B Register Bits | Register |
|---|---|
| 0 to 4 | Operation ( OP ) |
| 5 | Format ( Flag ) |
| 6 and 7 | Index Address ( Tag ) |
| 8 and 9 | Modifier ( Mod ) |
| 10 to 15 | Cycle Control Counter ( CCC )* |
| 8 to 15 | Displacement to D Register |
| * Shift instructions only | |

22212 A

Figure 3-3. B-Register to Control Register Transfer - I1

Certain operation codes (op codes) do not require that an EA be computed. These op codes are:

1. 'Shift left' or 'shift right.'
2. 'Load status.'
3. 'Wait.'

Note that these instructions are valid in short format only (F = 0).

All other instructions can require that some or all of the EA computation occur during the I1 cycle, depending on the format and tag bits. Any special functions required during I1 by an op code are described in the principles of operation for that op code.

Setting Control Registers

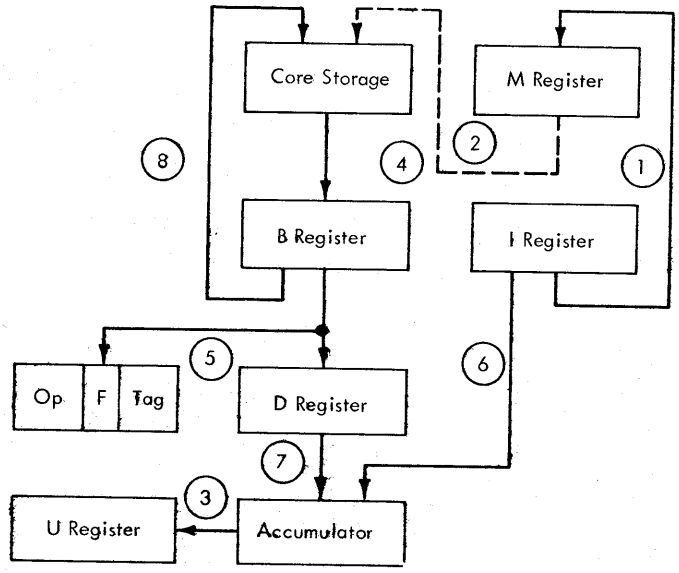
T0:
1. 'End op T0 SP' turns on 'I1' FF and turns off any other cycle timer FF.
2. Transfer I-register to M-register.
3. Increment IAR (phase B time).
4. Reset D, operation, format, tag, and modifier registers.

T1:
1. Transfer A-register to U-register. (Save contents of A-register for future use, particularly in arithmetic operations.)
2. Reset A-register.

T2:
1. Read word at location specified by M-register. Set into B-register (B-register is usually already set before T2.)
2. Gate B-register bits 0-4 to operation register 0-4.
3. Gate B-register bit 5 to format register. (F = 0 means single-word format.)
4. Gate B-register bits 6 and 7 to tag register. These bits must be off when no index register is to be used.
5. Gate B-register bits 8 and 9 to 'mod 8' and 'mod 9' FF's.
6. Gate B-register bits 8-15 to D-register bits 8-15 for use in computing EA, if required.
7. If B-register bit 8 = 1 (negative displacement), set 1's in D0-7.

Format = 0, Tag = 00

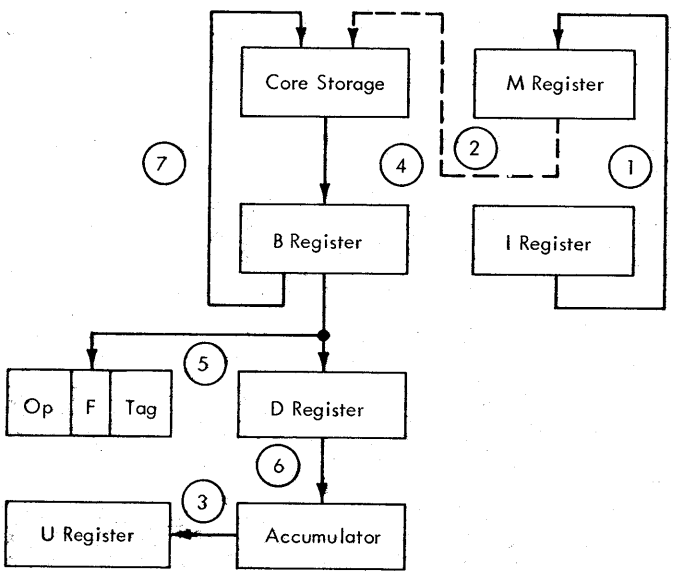
The EA is to be computed by adding the displacement to the contents of IAR (Figure 3-4).




1. Transfer instruction address from I to M.
2. Address core storage from M register.
3. Transfer accumulator contents to U register. Reset accumulator.
4. Read instruction word from core storage; load into B register.
5. Load operation, format, and tag registers with B register bits 0-7. Load D register with B register bits 8-15. Extend displacement sign (bit 8) in D register.
6. Transfer I register to accumulator.
7. Add D register to accumulator. Effective address is now in accumulator.
8. Write back instruction word into core storage.

13221

Figure 3-4. I1 Cycle, Format = 0, Tag = 00




1. Transfer instruction address from I to M.
2. Address core storage from M register.
3. Transfer accumulator contents to U register. Reset accumulator.
4. Read instruction word from core storage; load into B register.
5. Load operation, format, and tag registers with B register bits 0-7. Load D register with B register bits 8-15. Extend displacement sign (bit 8) in D register.
6. Add D register to accumulator. Displacement is now in accumulator, including sign.
7. Write back instruction word into core storage.

13222

Figure 3-5. I1 Cycle, Format = 0, Tag ≠ 0

T3:
1. Set 'arithmetic control' FF.
2. Turn on 'add' FF.
3. Transfer IAR to the A-register if the op code is not 'load index' or 'shift.' (IAR is one factor to be used in computing the EA.)

T4: Start adding D-register to A-register.

T5:
1. Continue adding.
2. Set CCC to 1 (to 16 for multiply, to 18 for divide).

T6: Continue adding.

T7: Complete addition of D-register to A-register (extend T7 if required).

End of T7: Cycle timer determines type of next cycle.

Format = 0, Tag ≠ 00

The EA is not computed completely during the I1 cycle. However, the displacement is added to the A-register which was reset at T1 (Figure 3-5). The contents of the specified index register are to be added to the displacement during the IX cycle.

T3:
1. Set arithmetic control.
2. Turn on 'add' FF.
3. I-register to A-register transfer is prevented when tag ≠ 00.

T4: Add D-register to A-register (no carries). Displacement is now in A-register.

T5: Set CCC to 1 (to 16 for multiply, to 18 for divide).

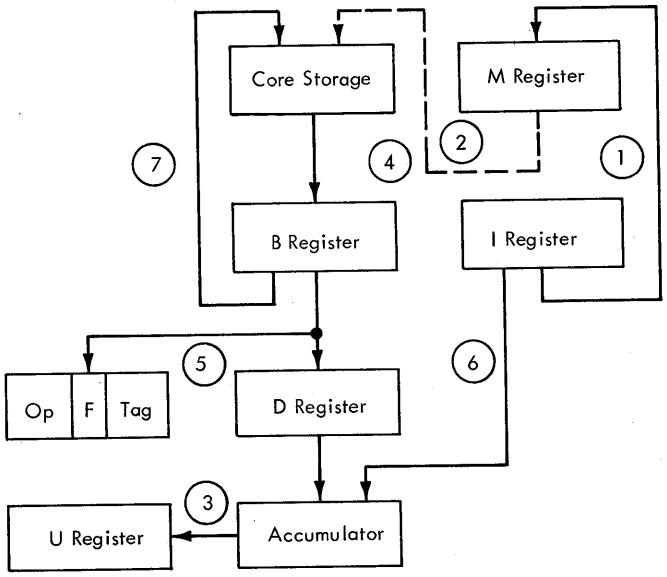
End of T7: Cycle timer determines type of next cycle.

Format = 1

The EA is computed completely by I-cycles other than I1. 'Arithmetic control' FF cannot be turned on at T3; so no computation of an EA can occur (Figure 3-6).

T3:
1. IAR, which contains address of instruction plus 1, is transferred to A-register if tag = 00. However, no use is made of this fact.
2. 'Arithmetic control' FF is blocked from turning on.

T5: Set CCC to 1 (to 16 for multiply, to 18 for divide.




1. Transfer instruction address from I to M.
2. Address core storage from M register.
3. Transfer accumulator contents to U register. Reset accumulator.
4. Read instruction word from core storage; load into B register.
5. Load operation, format, and tag registers with B register bits 0-7. Load D register with B register bits 8-15. Extend displacement sign (bit 8) in D register.
6. Transfer I register to accumulator.
7. Write back instruction word into core storage.

13223

Figure 3-6. I1 Cycle, Format = 1

T6: For MDX when tag = 00 only, turn on 'add to storage interlock' FF. This turns on the 'modifier 8' FF, forcing later IA cycle.

End of T7: Cycle timer determines that next cycle is an I2 cycle.

INSTRUCTION CYCLE 2 (I2)

- I2 cycle occurs for all double-word format instructions, including the interrupt-forced 'BSI indirect.'
- The 16 bits of the address word are set into the A-register via B-register and D-register.
- The address word contains the effective address unless it is to be modified by indexing or indirect addressing.
- Special I2 functions when the instruction is MDX and tag = 00 are described with MDX principles of operation.
- Reference maintenance diagram: AA601.

T0:
1. 'Not shift instruction,' 'format 5,' and 'I1 cycle' gate turn-on of 'I2' FF by 'not end op T0 SP.' Same pulse turns off 'I1' FF.
2. Transfer IAR to M-register. (IAR was incremented +1 during previous I1 cycle.)
3. Increment IAR (phase B time).

T2:
1. Read word at location specified by M-register. Set into B-register (usually before T2).
2. Transfer B-register to D-register.

T3: Although 'add/sub/or/EOR' is activated, 'arithmetic control' FF is not turned on.

T4: Transfer D-register to A-register by resetting D-register FF's at start of T4 while T3 gate is still active.

T7: Cycle timer determines type of next cycle.

INDEXING CYCLE (IX)

- Reads the contents of the index register (XR) selected by the tag (tag = 01, 10, or 11).

- Computes an address by adding the contents of the selected XR to the accumulator. The computed address may be the EA, or the indirect address of the EA.

- Indexing takes precedence over indirect addressing (occurs first when the instruction calls for both).

- IAR is not changed.

- Reference maintenance diagram: AA601.

The IX cycle reads the contents of the index register specified by the tag bits of the instruction (bits 6 and 7 now in the tag register). The contents of the XR are set into the D-register and added to the contents of the accumulator. The accumulator contains the displacement if the instruction is single-word format, and the address word if the instruction is double-word format. The instruction as it appears in core storage is not changed by indexing.

When an address is indexed and indirect, the indexing procedure has precedence. The resulting address after indexing is the indirect address of the effective address. The address at the indirect address must be the direct effective address. (Only one indexing and one indirect addressing operation are allowed for each instruction.)

The three index registers are located in the three core storage addresses /0001, /0002, and /0003. The normal output of the M-register is blocked. Decoding the contents of the tag register activates lines which substitute for the M-register output lines for bits 14 and 15.

T0:
1. 'Not end op T0 SP,' when properly gated, turns on both 'IX' FF and 'IX inhibit SAR' FF.
2. Tag register outputs activate 'IX address SAR 14' or/and 'IX address SAR 15' to substitute for corresponding M-register outputs in addressing core storage.

T1:
1. IX cycle causes no A-to-U transfer. U-register can contain accumulator data developed during previous instruction. Accumulator contains address that must be indexed by adding contents of XR.

T2:
1. Read word at addressed location (XR1, 2, or 3) and set into B-register (usually set before T2).
2. Transfer B-register to D-register for adding.

T3:
1. Turn on 'arithmetic control' FF, if not shift op.
2. Turn on the 'add' FF.

T4: Start adding D-register to the A-register.

T5, T6: Continue adding.

T7: Complete addition of D-register to A-register. (Extend T7 if required.)

End of T7: Cycle timer determines type of next cycle.

## INDIRECT ADDRESSING CYCLE (IA)

- IA cycle follows either an I2 cycle or an IX cycle.

- The indirect address is in the A-register at the start of the IA cycle.

- The address is read from the core storage location specified by the indirect address.

- By the end of the IA cycle, the EA replaces the indirect address in the accumulator.

- Only long-format instructions can use indirect addressing.

- Reference maintenance diagram: AA601.

The effective address (EA) has been defined as the address of data to be processed by the instruction. An indirect address can then be defined as the address of the EA.

The presence of a 1 in bit 8 position of a long-format instruction calls for indirect addressing. The address which is present in the accumulator at the end of the I2 cycle (or the IX cycle, if indexing has occurred) is an indirect address. This indirect address is the address of the core storage location where the EA is stored. The IA cycle obtains the EA from core storage and sets it into the accumulator.

Only one level of indirect addressing is available in the 1130 system. This means that the indirect address of an instruction always addresses the EA and never addresses another indirect address.

T0:
1. 'Not end op T0 SP,' when properly gated, turns on 'IA' FF.

2. Transfer accumulator (indirect address) to M-register. 'A to M SPD gate' remains active long enough into T0 to activate 'A to M SP 0-7' and 'A to M SP 8-15' lines.

T2:
1. Read EA from core location specified by M-register (indirect address). B-register is usually set before T2.
2. Transfer B-register to D-register.

T4:
1. Transfer D-register to A-register.
2. Note: Indirect address is cleared out of A-register.

End of T7: Cycle timer determines type of next cycle.

## EXECUTION CYCLES

- Execution cycles (E-cycles) may be required to execute the instruction read from core storage and interpreted during I-cycles.
- The cycle timer controls the types and sequence of E-cycles as well as I-cycles.
- Some instructions do not require any E-cycles.
- Reference maintenance diagrams: AA211, AA601.

The usual sequence of 1131 operation (Figure 3-7) is controlled by the cycle timer, depending on the format of each instruction and the operation to be performed by each instruction. I-cycles for each instruction are immediately followed by any E-cycles required for that instruction.

The function of E-cycles is to complete the execution of the instruction if execution cannot be completed during I-cycles. Execution of any of the following instructions is complete at the end of its I-cycle(s) and requires no E-cycles:

1. 'Wait' (00110 or any other unassigned op code).
2. 'Shift left" (00010), except when operation is 'shift left and count' and A-register bit 0 position contains a 1.
3. 'Shift right' (00011).
4. 'Load status' (00100).
5. 'Branch and store IAR' (01000) if no branch occurs.
6. 'Branch or skip on condition' (01001).
7. 'Load index" (01100) if no index register is specified.
8. MDX (01110) if flag = 0 and tag = 00.

When the execution of one instruction is completed, the computer enters I-cycles for the next instruction.

## E-CYCLE FLIP-FLOPS

The E-cycle flip-flops are the 'E,' 'E1,' and 'E3' FF's in the cycle timer circuits. Each E-cycle FF, in conjunction with the op code, controls the functions that are accomplished during the cycle for which it is on. The 'E' FF and the 'E1' FF are turned on for the cycle following the I-cycle(s) when one or more E-cycles are required to complete the execution of an instruction.

When only one E-cycle (E1 cycle) is required, both 'E' and 'E1' are turned off by the 'end op T0 SP.' When an E2 cycle is required, only the 'E1' FF is turned off by a 'not end op T0 SP.' While 'E' is on and both 'E1' and 'E3' are off, the 'E2 cycle' line is active. The 'E' FF is turned off later by an 'end op T0 SP,' after the required E2 cycle(s) (multiple E2 cycles are required for multiply or divide).

The 'E3' FF is only turned on following the E2 cycle of an XIO instruction, when the function code is read or write. Only one E3 cycle is taken per XIO instruction, so the 'E3' FF is turned off after one cycle by the 'end op T0 SP.'

E1 cycles occur for many op codes, and the functions accomplished depend on the op code. A typical example is the E1 cycle which occurs for a 'load accumulator' instruction, described in the following paragraphs. Different and additional functions required during an E1 cycle for other instructions are described in the principles of operation for that instruction.

### Load Accumulator E1 Cycle

The EA is in the A-register at the end of whatever I-cycles were required. At the next T0 time, the 'E' and 'E1' FF's in the cycle timer are turned on, and all B-register FF's are reset off. Also during T0, the EA is transferred from A to M so that the

EA selects a core location for the remainder of the cycle. At T1, the word which had been stored in the temporary accumulator (U-register) during EA computation is returned to the A-register. The CCC, which had been set to 1 during the I1 cycle, is decremented.

By T2 the core storage sense outputs have been strobed and the B-register set with the word from core storage (EA location). At T2, the 'gate B to D 0-7' and 'gate B to D 8-15' lines are activated. These lines AND with each B-register FF output to turn on the corresponding D-register FF. At this time, both the B-register and the D-register contain the same bit configuration as the addressed storage location had when it was read.

In order to transfer the contents of the D-register into the A-register (Figure 3-8), the 'add/sub/OR/EOR gate' and the 'AND SPD gate' must be active. The op code and T3 activate these gates, as well as the 'reset D SPD gate.' The gates remain active, because of circuit delays, beyond the end of T3 phase B, when 'reset D reg SP' is

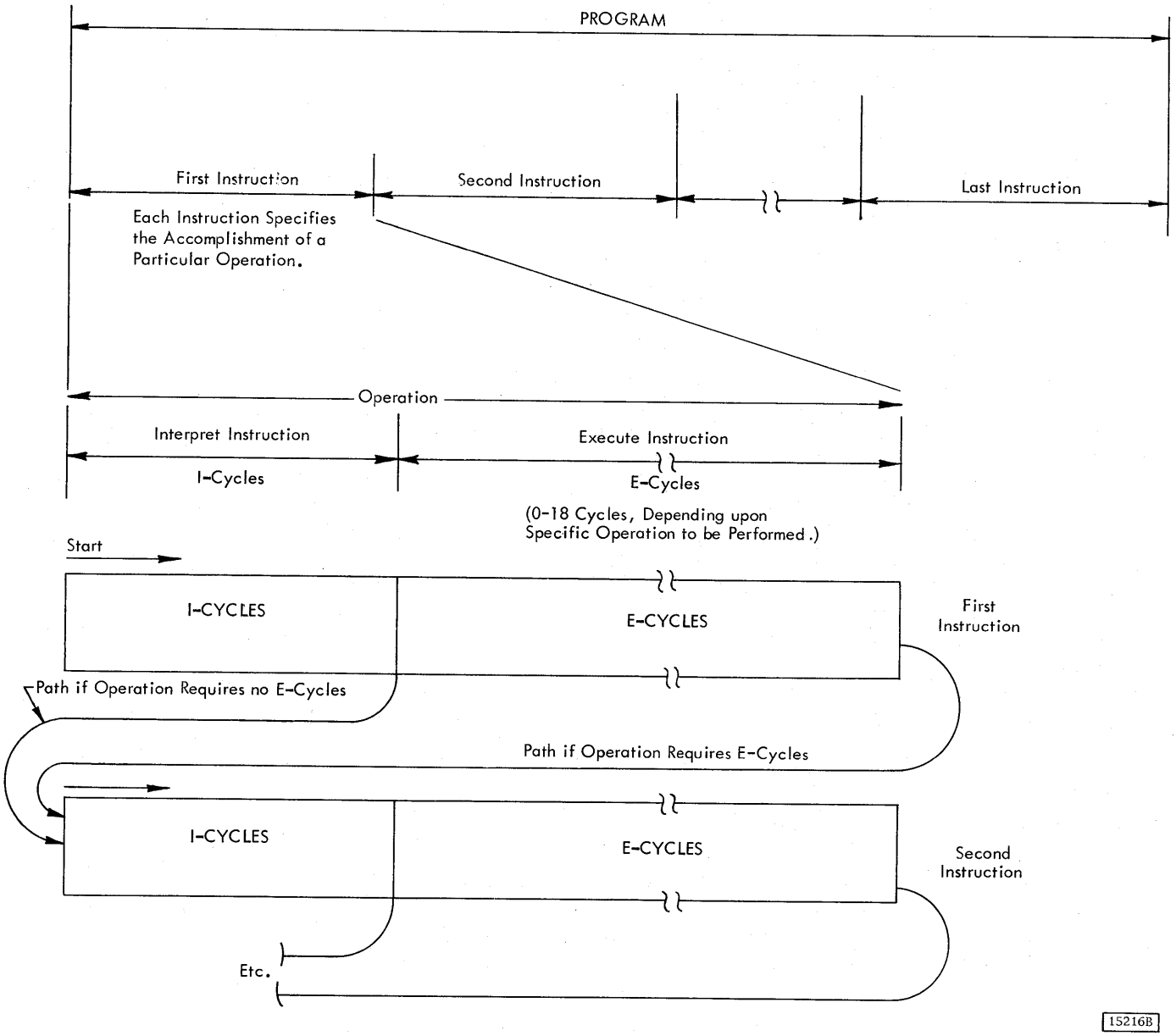



Figure 3-7. Sequence of Operation

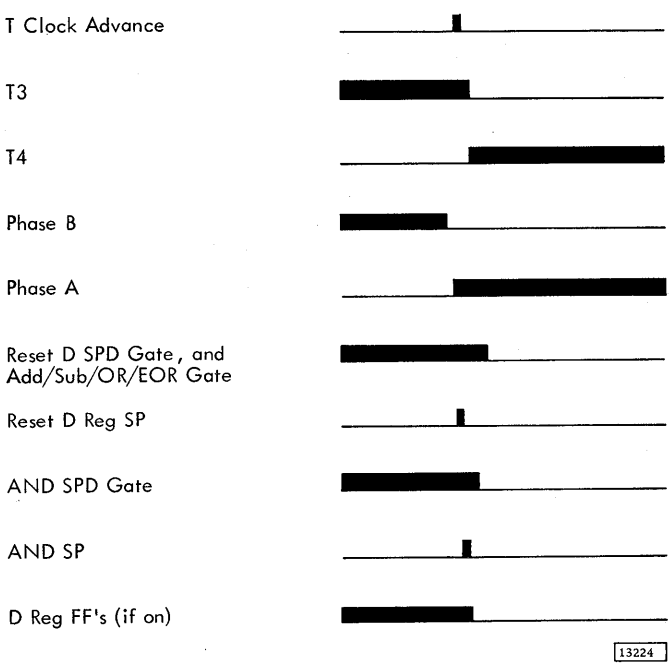



Figure 3-8. D-Register to A-Register Transfer Load Accumulator

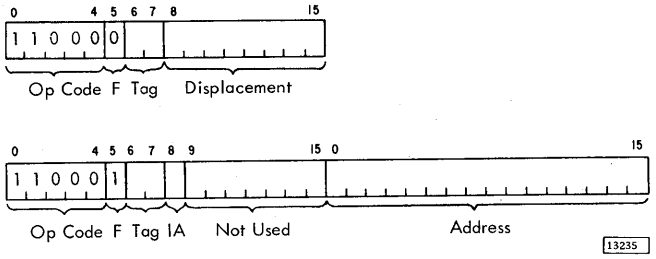
activated. The reset pulse turns off all D-register FF's just as the T-clock is advancing to T4. Each D-register FF which turns off causes the corresponding A-register FF to turn on. Each D-register FF which was already off gates the turn-off of the corresponding A-register FF. The 'AND SP' pulse turns off the A-register FF's that are gated by D-register FF's which are already off. Thus, the same bit configuration which was read from storage is set into the A-register.

The B-register still contains the word as it was read from storage. During the write portion of the cycle, the B-register controls the inhibit circuits so that the word is written into core storage unchanged.

## LOAD AND STORE INSTRUCTION GROUP

- Load instructions read the core storage word at the EA and load the information into other circuit components.
- Store instructions write information from other circuit components into the core storage word at the EA.
- Besides core storage, the other circuit components used by this instruction group are the accumulator, accumulator and extender, index registers, or 'carry' FF and 'overflow' FF.

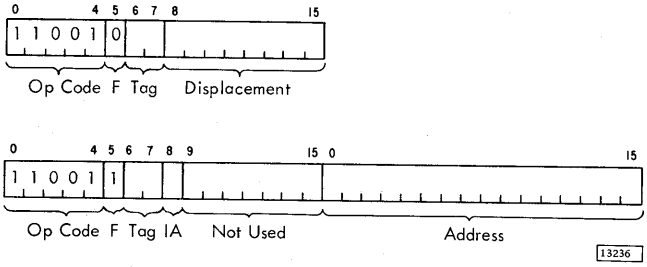
## LOAD ACCUMULATOR (LD)




- Short format can be indexed. EA is computed during I1 cycle and IX cycle when called for.
- Long format can be indexed, indirectly addressed, or both. The address word provides the EA or is used in EA computation.
- LD sets the accumulator (A-register) to the same bit configuration as the addressed word.
- The word is written back at the addressed location unchanged.
- 'Carry' and 'overflow' FF's are not affected.
- Maintenance diagram AA681.

Execution of this instruction requires only an E1 cycle in addition to the I-cycles called for to compute the EA. A detailed description of the E1 cycle is included in the description of E-cycle flip-flops in this chapter.

LOAD DOUBLE (LDD)




- Short format can be indexed. EA is computed during I1 cycle and IX cycle when called for.
- Long format can be indexed, indirect addressed, or both. The address word provides the EA or is used in EA computation.
- LDD sets the accumulator (A-register) and its extension (Q-register) to the same bit configuration as the two words at EA and EA + 1.
- Both words are written back into storage unchanged.
- Low-order word of double-precision word (word at EA + 1) is in Q-register at end of operation. High-order word (from EA) is in accumulator.
- EA must be even. Otherwise, the word at EA + 1 sets both the A-register and the Q-register.
- 'Carry' and 'overflow' FF's are not affected.
- Maintenance diagram AA682.

Execution of this instruction requires both an E1 cycle and an E2 cycle. Op code decoding activates the 'double word addressing' line and develops a gate so that the 'DPW odd address' FF can be turned on at the same time 'E' and 'E1' turn on. The 'DPW odd address' FF stays on for only one cycle, being turned off by a 'not end op SP' when E1 is also turned off. Consequently, the 'DPW odd address SAR 15' line is active throughout the E1 cycle, and substitutes for the 'CPU address 15' line. Thus, the EA + 1 location is addressed without the 'M bit 15' FF being on. (The EA is even.)

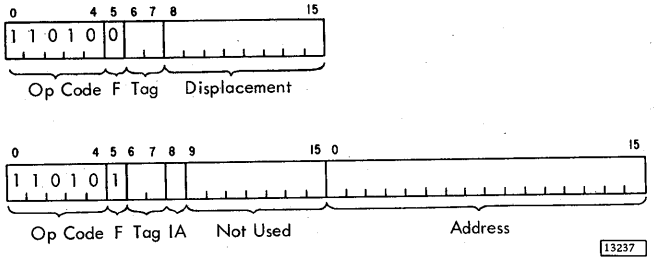
The 'double word addressing' line also gates 'exchange A and Q SPD sample' which is activated at T2. Although this exchange occurs during the E1 cycle, no use is made of this fact. The contents of both A and Q are to be replaced later in the operation.

Other functions of the E1 cycle are the same as for a LD instruction, except that the CCC is not decremented. At the end of E1 cycle, the word from EA + 1 is present in the A-register as well as written back at EA + 1 location unchanged. Because 'double word addressing' has blocked the 'CCC decrement sample,' the CCC still contains the 1 that was set during I1. No end operation condition can occur, and the cycle timer causes entry into an E2 cycle.

When the E2 cycle is entered, the EA still remains in the M-register. However, 'DPW odd address SAR 15' is no longer active, because the 'DPW odd address' FF has been turned off. The word at EA is read and set into the B-register. At T2, the B-register sets the D-register, and A and Q contents are again exchanged. When the 'exchange A and Q SP' lines are activated, each A-register FF is turned on or off depending on the state of the corresponding Q-register FF. Conversely, the A-register FF's gate the corresponding Q-register FF's in the same manner. The exchange can occur because of the requirement that the gate must be present before the sample pulse and because of inherent circuit delays.

After the exchange of A and Q, the word from EA + 1 is present in the Q-register. At T4, a D-register to A-register transfer occurs similar to the one during the E1 cycle. Then the normal write-back into core storage (at the EA) occurs, and the operation is complete. At T1 of the E2 cycle, the CCC was decremented to 0. 'T7' and CCC count 0 activate the 'I to M SPD gate' in preparation for the I1 cycle which follows. The next 'T clock advance SP' activates the 'end op T0 SP.'

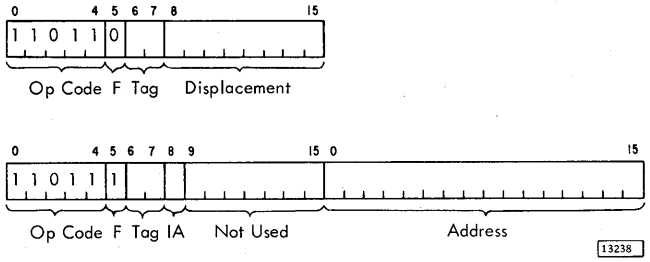
## STORE ACCUMULATOR (STO)




- Short format can be indexed. EA is computed during I1 cycle and IX cycle when called for.
- Long format can be indexed, indirectly addressed, or both. The address word provides the EA or is used in EA computation.
- STO causes the word at the EA to be written with the same bit configuration as the accumulator (A-register).
- Word in the accumulator is not changed.
- 'Carry' and 'overflow' FF's are not affected.
- Maintenance diagram AA683.

Execution of this instruction requires only an E1 cycle in addition to the I-cycles required to compute the EA. Operation is standard through T2 time, even including the setting of both B- and D-registers with the word from core storage. The op code activates the 'store E' line when E1 is entered. During T2 time, the 'A to B 0-7 SPD gate' and 'A to B 8-15 SPD gate' are activated. Very early in T3 time, the 'A to B SP 0-7' and 'A to B SP 8-15' occur. These sample pulses set each B-register FF according to the state of the corresponding A-register FF. As a result, the B-register contains the same bits as the A-register. Normal write-back circuits then write the same word in the EA location. Decrementing CCC to 0 allows the next T0 to cause entry into an I1 cycle.

## STORE DOUBLE (STD)



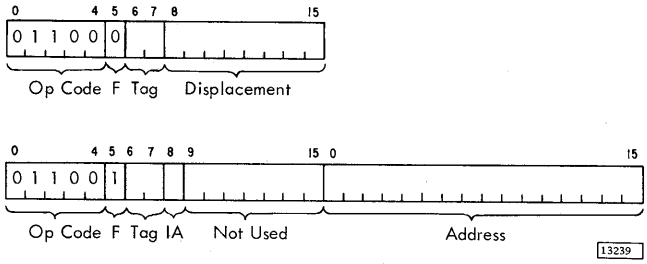

- Short format can be indexed. EA is computed during I1 cycle and IX cycle when called for.
- Long format can be indexed, indirectly addressed, or both. The address word provides the EA or is used in EA computation.
- STD causes the two words at EA and EA + 1 to be written with the same bit configuration as the double word in the accumulator (A-register) and its extension (Q-register).
- Low order positions of the double precision word (Q-register) are stored first at EA + 1. Then A-register bits are stored in EA.
- EA must be even. Otherwise, the odd-numbered location is written on two successive cycles and finally contains the A-register word.
- Q- and A-register contents are not changed.
- 'Carry' and 'overflow' FF's are not affected.
- Maintenance diagram AA684.

Execution of this instruction requires an E1 cycle and an E2 cycle in addition to the I-cycles required to compute the EA. Operation during E1 is like that for a STO instruction except that the 'double word addressing' line is active. This results in addressing core storage location EA + 1 in the same manner as for an LDD instruction. Also, at T2, A- and Q-

registers are exchanged. Consequently, when the A-register sets the B-register at T3 time, the B-register contains the word which was in Q at the start of the operation. At write time of the E1 cycle, the original Q-register word is written at EA + 1.

Again during the E2 cycle, the A- and Q-registers are exchanged at T2, restoring the words to their original registers. After A is duplicated in B at T3 time and then written back into the EA location, the operation is complete. CCC has been decremented to 0 so at the next T0, the 'end op T0 SP' can be activated.

LOAD INDEX (LDX)




- LDX is one of the index instructions. No IX cycle can occur.
- No EA is generated for short format. Displacement is the data to be loaded.
- In long format, the address word is the data unless IA bit 8 = 1. Then the address word is the address of the data to be loaded.
- Tag selects the register to be loaded. T = 00, LDX loads IAR; T = 01, 10, or 11, LDX loads XR1, XR2, or XR3, respectively.
- When T = 00 and LDX loads IAR, an unconditional branch occurs.
- Maintenance diagram AA661.

For an LDX instruction, an EA is never computed during the I1 cycle. Moreover, the I- to A-transfer is blocked. The displacement is set into the D-register 8-15 positions, and the displacement sign (bit 8) is extended throughout positions 0-7, as usual during I1. Adding the displacement to the A-register, which had been reset to all 0's, sets the displacement, including sign, into the A-register.

An indexing cycle never occurs for an index instruction, but a long format LDX may call for an IA cycle (bit 8 = 1). Regardless of what I-cycles are required, by T5 time of the last I cycle, the A-register contains the data to be loaded into the selected register. The manner in which execution of the instruction is completed depends upon the tag code.

Tag = 00

When T = 00, the data is loaded into the IAR, and an unconditional branch occurs (Figure 3-9). The negative shift at the end of T5 time turns on the 'branch 1' FF. T7 phase A turns on the 'branch 2' FF when 'branch 1' is on. Both FF's being on activate the 'A to M SPD sample' and fire the 'A to M SP' lines, placing the data in the M register. 'Phase B' and 'branch 2' FF reset the CCC to 0, which activates the 'T7 end op' line. This line ANDs with 'branch 2' FF and 'not shift instruction' to activate the 'U to A SPD sample' and 'U to A SP' lines. The word which had been saved in the temporary accumulator (U-register) is returned to the A-register. The next 'phase A SP A' turns off the 'branch 1' FF, activating 'M to I SPD sample,' and 'M to I SP' sets the data into the I-register (IAR). The next 'phase B' turns off the 'branch 2' FF, and the operation is complete. IAR contains the "branch to" address for use on the next I1 cycle.

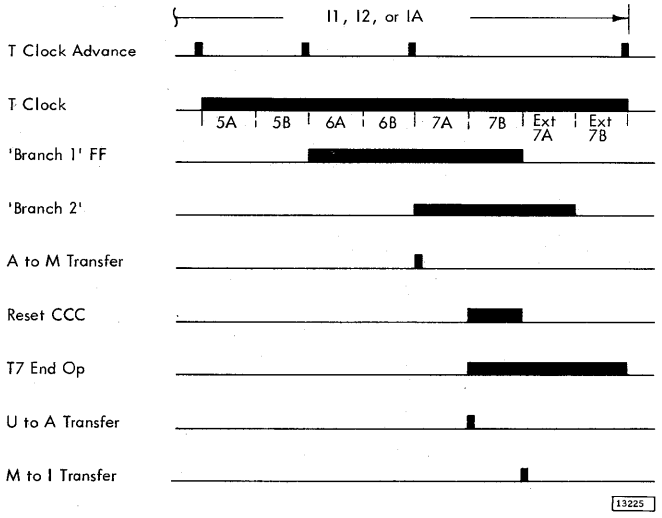
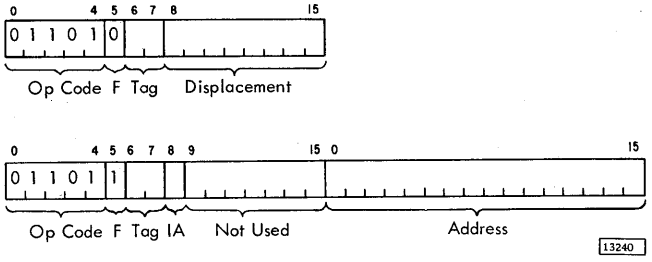



Figure 3-9. Branching on LDX, Tag = 00

Tag ≠ 00

When the tag does not equal 00, the data which is in the A-register at the end of the last I-cycle (may be I1, I2, or IA) is to be loaded into an index register (XR). Execution of the instruction then requires an E1 cycle. Normal addressing of core storage by the M register is blocked, because the 'IX inhibit SAR' FF is turned on. Instead, core storage is addressed as described for an IX cycle. During T2, the 'A to B SPD gate' lines are activated. Early in T3 time, the 'phase A SP B' then activates the 'A to B SP' lines which set the B-register FF's the same as the corresponding A-register FF's. When the B-register is written into storage, the operation is complete. Because the CCC was decremented at T1 of the E1 cycle, the next T0 activates 'end op T0 SP,' and the next I1 can be entered.

STORE INDEX (STX)




- STX is one of the index instructions. No IX cycle can occur.
- EA of short format instruction is the sum of IAR and the displacement, computed during the I1 cycle.
- The address word of a long format instruction may be an indirect address.
- Tag selects the register to be stored. T = 00, STX stores IAR at the EA; T = 01, 10, or 11, STX stores XR1, XR2, or XR3, respectively.
- Maintenance diagram AA662.

During the I1 cycle of a STX instruction, an I-to-A transfer occurs at T3. The displacement is added, starting at T4, if the instruction is short format. Thus the EA (IAR + displacement) is in the accumulator at the end of I1. For a long-format instruction, the EA is set into the accumulator during either I2 or the IA cycle, if called for. Execution of the instruction, thereafter, depends on the tag register. Note that the word in the accumulator is not disturbed by this instruction. Early in I1, the A-register is transferred to the U-register. The last step in execution of the instruction is returning the contents of the U-register to the accumulator.

Tag = 00

An E1 cycle is required to store the contents of IAR at the EA. The A-register contents are transferred to the M-register and address core storage throughout the E1 cycle. During T2, the 'I to B gate' is active, and, at the end of T2, the contents of the IAR are set into the B-register. When this is written into core storage later, the operation is complete. Note that the CCC was decremented to 0 at T1 of the E1 cycle.

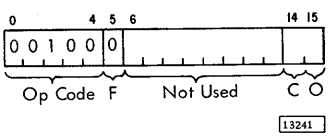
Tag ≠ 00

An E1 cycle sets the contents of the selected XR into the A-register, and an E2 cycle then stores the contents of the A register in core storage at the EA.

During the E1 cycle, the EA is transferred from A to M, but not used to address core storage. Instead, the 'IX address SAR' lines are activated according to the setting of the tag register. The operation is similar to an IX cycle. Reading from the XR sets the B-register, and at T2 the D-register is also set with the contents of the XR. At T4, the D-register is reset to 0 and the contents set into the A-register.

During the E2 cycle, core storage is addressed by the M-register, which had been set to the EA in the E1 cycle. The contents of the A-register, which were originally in the XR, replace what is read from storage in the B-register. When this is written into the EA location and the U-register transferred back to the A-register, the operation is complete. The CCC is decremented to 0 at T1, so the following T0 causes entry into the next I1 cycle.

LOAD STATUS (LDS)

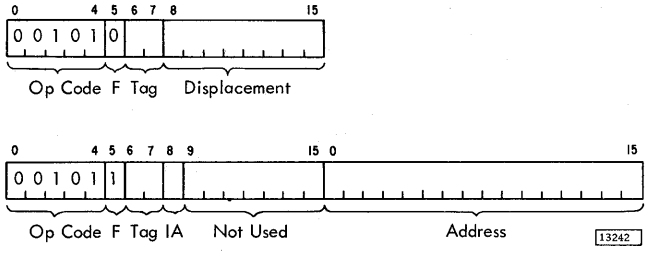



- LDS is valid in short format only.

- No EA is required, and no indexing cycle can occur.
- If instruction word bit 14 = 1, turn on 'carry' FF. If instruction word bit 15 = 1, turn on 'overflow' FF.
- Carry status and overflow status usually have been stored in this instruction word by a prior 'store status' instruction.
- Execution is complete during I1 cycle.
- Execution does not change the instruction word in core storage.
- Maintenance diagram AA641.

The LDS instruction causes certain changes in the operation of the I1 cycle. At T3, the 'arithmetic control trigger' FF is not turned on, because 'op 001x0' is active. D-register positions 14 and 15 have been set to the same state as the corresponding positions of the instruction word. Doing so is a normal function of an I1 cycle (storage to B to D). Then a 'load status T4' pulse turns on the 'carry' FF if it is gated by the 'D bit 14' line being active. 'D bit 15' gates the turn on of the 'overflow' FF in a similar manner. At T5, the usual setting of the CCC to 1 is blocked by 'op 001x0.' Thus the next T0 causes entry into another I1 cycle. Contents of the A-register have been saved in the U-register and are returned to the A-register at T7.

STORE STATUS (STS)




- Short format can be indexed. EA is computed during the I1 cycle and IX cycle when called for.
- Long format can be indexed, indirect addressed, or both. The address word provides the EA or is used in EA computation.
- STS stores the on/off status of the 'carry' and 'overflow' FF's in the core storage word at the EA. If 'carry' is on, bit 14 is set to 1, and if 'overflow' is on, bit 15 is set to 1.
- Bits 0-7 of the word at EA are unchanged, and bits 8-13 are reset to 0's.
- 'Carry' and 'overflow' FF's are reset off when their status is stored.
- The word in which status is stored is usually an LDS instruction.
- Maintenance Diagram AA642.

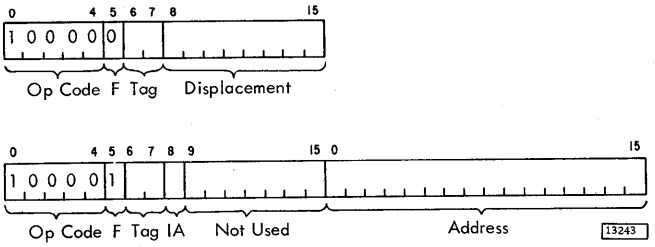
Execution of this instruction requires an E1 cycle following the I-cycles necessary to set the EA into the A-register. At T0, the EA is immediately transferred to the M-register to address core storage. Then the A-register is reset to 0 by the 'store status E' line and T0, phase B. By T2 time, the normal readout from storage sets the word from EA into the B register. Although the same word is also set into the D-register at T2, no use is made of this fact. Also at T2, the 'A bit 14' and 'A bit 15' FF's are set to the same state as the 'carry' and 'overflow' FF's. Only the 'A to B 8-15 SPD gate' is activated during T2, so only positions 8 to 15 of the B-register are set the same as positions 8 to 15 of the A-register. The sample pulse occurs at the end of T2 at about the same time that the 'carry' and 'overflow' FF's are reset off. Note that at the time of the A8-A15 to B8-B15 transfer, only 'A bit 14' and 'A bit 15' may be on, and 'B bit 8' through 'B bit 13' are always turned off by the transfer. After the B-register word is written into storage, the operation is complete. The CCC was decremented to 0 at T1, so the next T0 causes entry into an I1 cycle.

## ARITHMETIC INSTRUCTION GROUP

- All arithmetic instructions require that one factor already be present in the accumulator (A-register).
- Execution of any arithmetic instruction uses the add-subtract circuits.
- The control gates that are activated depend on the instruction (op code) and the type of cycle which is taking place.
- All arithmetic instructions are valid in either short or long format.
- I-cycle operation is standard. Op code is set into the operation register, and the EA is present in the A-register at the end of I-cycles.

Execution of any arithmetic instruction requires the use of two factors, one of which must be present in the accumulator (and its extension, in some cases), before the instruction is given. During I-cycles, this factor is stored in the temporary accumulator (U-register) to allow EA computation in the A-register. After the EA is transferred to the M-register, early in the first execution cycle (E1 cycle), the temporarily stored factor is returned to the A-register. Then, when the second factor is read from core storage and set into the D-register via the B-register, both factors are available to the add-subtract circuits. The operation of these circuits depend on which gates are active. The gates, in turn, are activated according to the op code and operation to be executed.

ADD (A)




- The word read from EA is added to the word in the A-register.
- The word at EA is written back unchanged.
- Sum is in A-register at end operation time.
- Negative factor or sum is in 2's complement form.
- 'Carry' FF is reset off and may be turned on during execution of this instruction.
- 'Overflow' FF is not reset. It may be turned on, but cannot be turned off during execution of this instruction.
- Maintenance diagram AA671.

Execution is completed during an E1 cycle which follows any I-cycles necessary to set the EA into the A-register. Before the end of T2 time, the two factors are in the D- and A-registers, ready for addition. At T3, the 'arithmetic control trigger' FF turns on, and a 'T3 SP' sets the 'add' FF to add (on). Also, at T3, if the signs of the factors are different, the 'arithmetic sign' FF turns on. Starting at T4 time, addition proceeds as described in Chapter 2. During addition, if the 'A bit 0' FF turns off, the 'temporary carry' FF turns on, indicating a carry out of the high-order A-register position. Turning on 'temporary carry' also turns on the 'carry' FF.

During addition, each phase B checks the D-register for an all-zero condition. When this condition is reached, the 'arithmetic control trigger' FF is turned off. When the D register does not reach the all-zero condition by the first T7B pulse, extended T7 time is required. The T-clock cannot advance to T0 until the 'arithmetic control trigger' FF is turned off.

When the 'arithmetic control trigger' FF is turned off, 'T7' and 'CCC count 0' can activate 'T7 end op.' At this time, the 'overflow' FF may be turned on, but it cannot be turned off. Two exclusive OR's (Figure 3-10) determine whether or not 'overflow' is turned on. A chart on maintenance diagram AA671 shows the conditions for turning on 'overflow.' A positive result is present in the A-register when A0 = 0. ('A bit 0' FF is off.)

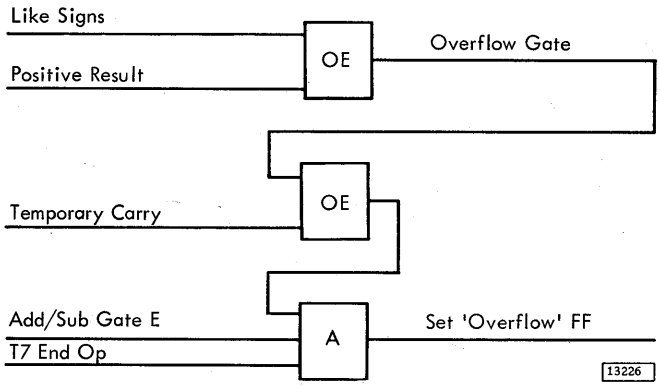
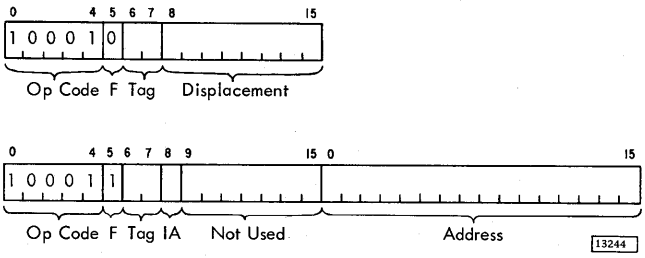



Figure 3-10. Overflow in Add or Subtract Operations

ADD DOUBLE (AD)




- Two core storage words at EA and EA + 1 are treated as a 32-bit double word (31 significant binary digits and a sign).
- The addressed double word is written back into core storage unchanged.
- The addressed double word is added to a double word already present in the A- and Q-registers (Q-register is low order).
- EA must be even; otherwise, the word at the odd EA is added to both the A- and Q-registers.
- 'Carry' FF is reset off and may be turned on later in the execution of this instruction.
- 'Overflow' FF is not reset. It may be turned on, but cannot be turned off during execution of this instruction.
- Maintenance diagram AA672.

Execution of this instruction requires two cycles, an E1 and an E2, because the add-subtract circuits are capable of handling only 16 positions in one cycle. During the E1 cycle, the contents of EA + 1 are to be added to the contents of the Q-register. To do this, the E1 cycle must perform certain functions in addition to those described for 'add: '

1. Address EA + 1 by activating 'DPW odd address SAR 15.' See "Load Double." Contents of EA + 1 are set into D-register for adding.
2. Exchange A- and Q-registers. Original contents of Q-register are set into A-register for adding. Original contents of A-register are saved for use in E2 cycle.

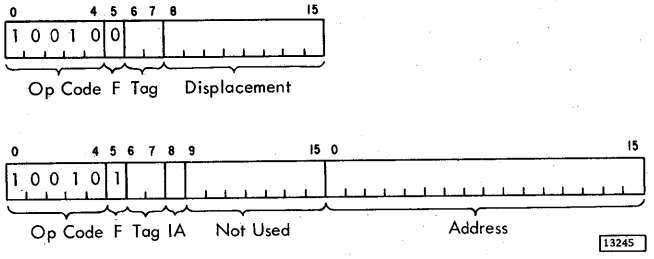
'Temporary carry' is not used in determining 'overflow' but may be turned on if a carry into the units position of the A-register is required in E2. The CCC is not decremented during the E1 cycle; so no end op condition can be activated.

In the E2 cycle, the contents of EA are to be added to the original contents of the A-register. The E2 cycle must perform certain functions in addition to those described for an 'add' E1 cycle:

1. Address EA by deactivating 'DPW odd address SAR 15.' Contents of EA are set into D-register for adding.
2. Turn on 'double precision carry' FF if 'temporary carry' FF is on. Then turn off both 'temporary carry' and 'carry' with 'T3 SP.'
3. Exchange A- and Q-registers. Original contents of A-register are set back into A-register for adding. Partial sum of contents of Q and contents of EA + 1 is set back into Q-register.
4. When other D-register FF's are being turned on to represent any carries during T4 time, turn on 'D bit 15' if 'double precision carry' FF is on. This turn-on cannot occur early enough in T4 time to allow T4 'reset D reg SP' to turn 'D bit 15' back off. Therefore, carry into A-register 15 position does not occur until T5.

The final status of 'carry' and 'overflow' is determined in E2 of an AD just as it was in E1 of an A instruction. The CCC is decremented to 0 during E2, and the operation ends at the end of T7 or any extended T7 times which occur.

SUBTRACT (S)

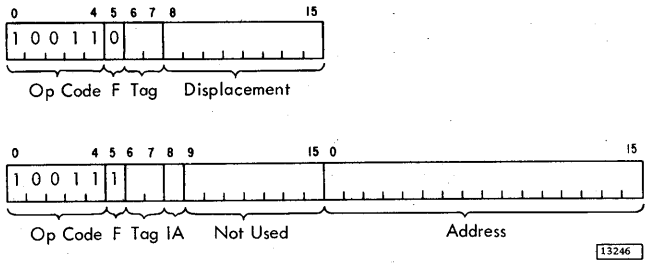



- The word read from EA is subtracted from the word in the A-register.
- The word at EA is written back unchanged.
- Result is in A-register at end operation time.
- Negative factor or difference is in 2's complement form.
- 'Carry' FF is reset off and may be turned on during execution of this instruction.
- 'Overflow' FF is not reset. It can be turned on, but cannot be turned off during execution of this instruction.
- Maintenance diagram AA671.

Execution of this instruction is very nearly the same as execution of an 'add' instruction. Differences result from the fact that the 'op bit 3' prevents the 'add' FF from being turned on. ('Add' is reset off during I1 cycle.) Operation of the add-subtract circuits is changed, because 'subtract gate' is active rather than 'add gate.' Any D-register FF which is turned on represents a borrow instead of a carry. An attempt by the A0 position to borrow from the next higher order position (nonexistent) turns on the 'temporary carry' FF. The circuits operate the same as when an attempt is made to carry from A0 position into the same nonexistent higher-order position.

The final status of the 'carry' FF depends on whether or not a borrow is attempted beyond the A-register capacity during this subtract operation. (The 'carry' FF is reset off before subtraction starts.) However, 'overflow' is not reset, and if it was previously on, this operation does not affect it. If 'overflow' is off when this subtraction starts, the final status is determined as it is in execution of an 'add' instruction. The operation ends at the end of T7, or any extended T7 times that occur.

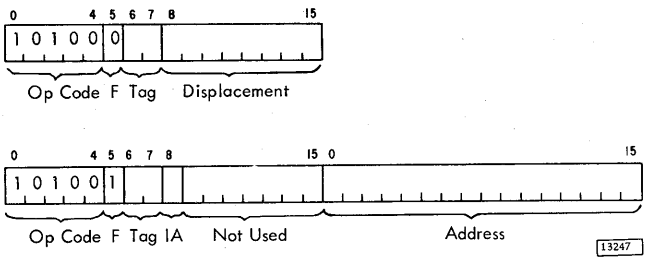
SUBTRACT DOUBLE (SD)




- Two core storage words at EA and EA + 1 are treated as a 32-bit double word (31 significant binary digits and a sign).
- The addressed double word is written back into core storage unchanged.
- The addressed double word is subtracted from a double word already present in the A- and Q-registers. (Q-register is low order.)
- EA must be even; otherwise, the word at the odd EA is subtracted from both the A- and Q-registers.
- 'Carry' FF is reset off and can be turned on later in the execution of this instruction.
- 'Overflow' FF is not reset. It can be turned on, but cannot be turned off during execution of this instruction.
- Maintenance diagram AA672.

Execution of the SD instruction is very nearly the same as execution of the AD instruction. 'Subtract gate' is active instead of 'add gate,' and borrows occur instead of carries. The final status of the 'carry' and 'overflow' is determined during the E2 cycle. The operation ends at the end of T7 or any extended T7 times that occur.

MULTIPLY (M)



- The word at the EA is the multiplicand and is multiplied by the word already in the A-register, which is the multiplier.

- Each time the word at EA is read, it is written back during the same cycle, unchanged.

- The product, a 32-bit double word (31 significant digits and sign), is in the A- and Q-registers at end operation time. The Q-register contains the low-order positions.

- Multiply operation sets the sign of the product according to algebraic rules.

- Product cannot exceed the capacity of the A- and Q-registers, so the multiply operation does not affect the 'carry' and 'overflow' flip-flops.

- Maintenance diagram AA673.

1131 Multiply Algorithm

Execution of a 'multiply' instruction uses a rapid multiplication algorithm which requires fewer add cycles than conventional methods. The algorithm is based on the fact that shifting the 1's in a binary number to the left multiplies the number by 2 raised to some power. No shift multiplies the number by $2^0$ ($2^0 = 1$). Shifting the number one position multiplies the number by $2^1$ ($2^1 = 2$). Shifting the binary number two positions multiplies the number by $2^2$ or 4. Therefore, shifting a binary number (the multiplicand) a number of positions (n) multiplies the binary number by $2^n$. Some examples:

1. 2 x 8 = 16 -- 000010 (2), shifted three positions ($2^3 = 8$), becomes 010000 (16).
2. 1 x 16 = 16 -- 000001 (1), shifted four positions ($2^4 = 16$), becomes 010000 (16).
3. 4 x 8 = 32 -- 000100 (4), shifted three positions ($2^3 = 8$), becomes 100000 (32).
4. 10 x 4 = 40 -- 001010 (10), shifted two positions ($2^2 = 4$), becomes 101000 (40).

A counter is used to accumulate partial products, necessary when the multiplier is other than 1, 2, 4, 8, 16, etc. Assume a multiplicand of 000101 (5) and a multiplier of 001001 (9). The multiplicand is multiplied by 1 (no shift) and added into the counter. Then the multiplicand is multiplied by 8 (shifted three positions) and added into the counter again. The product is in the counter after the second addition. The operation is as follows:

| | |
|---|---|
| 000000 | Counter clear at start. |
| 000101 | Add 000101 (no shift). |
| 000101 | After first addition. |
| 101000 | Shift m'cand three positions and add. |
| 101101 | After second addition. Product = 45. |

Multiplication by -1 can be accomplished by subtracting the multiplicand from the counter with no shift.

The simple example of multiplying 000010 (decimal 2) by 001111 (decimal 15) shows the savings in add cycles derived from using the algorithm. A conventional method might consist of adding the multiplicand (000010) into a counter once for each 1 in the multiplier (001111). This method requires four add cycles with the multiplicand being shifted left after each addition. Assuming a 12-position counter, multiplying would proceed as follows:

| | |
|---|---|
| 0 0 0 0 0 0 0 0 0 0 0 0 | Counter at start. |
| 0 0 0 0 0 0 0 0 0 0 1 0 | End of 1st add cycle. |
| 0 0 0 0 0 0 0 0 0 1 1 0 | End of 2nd add cycle. |
| 0 0 0 0 0 0 0 0 1 1 1 0 | End of 3rd add cycle. |
| 0 0 0 0 0 0 0 1 1 1 1 0 | End of final add cycle. |

The algorithm performs the same multiplication by multiplying by -1 and +16, in effect. (Multiplier is +15.) Stated another way, the multiplicand is subtracted with no shift, then added later. Addition occurs when enough shifts have taken place to make the value added 16 times the value of the multiplicand. Thus, only two add (or subtract) cycles are required:

| | |
|---|---|
| 0 0 0 0 0 0 0 0 0 0 0 0 | Counter at start. |
| 1 1 1 1 1 1 1 1 1 1 1 0 | End of subtraction cycle. |
| 0 0 0 0 0 0 0 1 1 1 1 0 | End of add cycle. |

Shifting the contents of the counter (partial product or product) to the right allows the multiplicand to be added to or subtracted from the proper position of the counter. The effect is the same as shifting the multiplicand to the left. The total number of shifts in a multiply operation is equal to the number of positions in the multiplier. This method is used in the 1131, with the CCC controlling the number of shifts. The counter comprises the A- and Q-registers. (The Q-register contains the low-order position.)

The 1131 does the multiplication of 000010 (2), the multiplicand, and 001111 (15), the multiplier, in the following manner. Six-position registers are assumed in the example; therefore, Q4 and A5 are the equivalent of Q14 and Q15 in the machine.

```
     A          Q
0 0 0 0 0 0 0 0 1 1 1 1   Exchange A&Q. Reset A.
0 0 0 0 1 0               Subtract multiplicand.
-----------
1 1 1 1 1 0 0 0 1 1 1 1
1 1 1 1 1 1 1 1 1 0 0 0   Shift A and Q four
                          positions.
0 0 0 0 1 0               Add multiplicand.
-----------
0 0 0 0 0 1 1 1 1 0 0 0
0 0 0 0 0 0 0 1 1 1 1 0   Shift A and Q two
                          positions.
```

During the shifting of a minus partial product, 1's are set into the high-order position of the A-register to maintain the negative sign. Remember that the multiplier is in the A-register when the 'multiply' instruction is given. Early in the operation, contents of the A- and Q-registers are exchanged. After the exchange, the A-register is reset to zero.

Determination of when to add and shift, when to subtract and shift, and when to shift only is shown in Figure 3-11. The chart shows that the Q14 and Q15 bits and the previous action (add or subtract) determine the new action to take. At the start of 'multiply,' it is assumed that the previous action was add. Each shift shifts all bits in both the A- and Q-registers one position to the right.

The sequence of actions for any multiplier can be determined from the chart. With a . used to indicate shift only, + to indicate add and shift, and - to indicate subtract and shift, some examples of different multipliers are shown:

```
                  . . . . . . . . . . + . . . -
Example 1         0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1

                  . . + . . - . + . - . - . - . .
Example 2         0 0 0 1 1 1 0 0 1 0 1 0 1 1 0 0

                  + . . . . - . + . . . . . - .
Example 3         0 1 1 1 1 1 0 0 1 1 1 1 1 1 1 0
```

The most add cycles are eliminated when the multiplier contains long strings of 1's or 0's, as in examples 1 and 3.

In the two examples of multiply operations (Figure 3-12), six-position registers are used. The two low-order positions of the Q-register are the equivalent of Q14 and Q15, to agree with the rules in Figure 3-11. The cycle control counter is set to 6, because the Q-register (multiplier) has six positions. In the machine, CCC is set to 16 at the start of a multiply operation.

## Multiply Cycles

Execution of a 'multiply' instruction requires an E1 cycle and a variable number of E2 cycles, in addition to the I-cycles required to develop the EA of the multiplicand. The multiplier must already be in the A-register. When the instruction is 'multiply' the CCC is set to 16 rather than 1 during the I1 cycle.

### E1 Cycle

Maintenance diagram AA673 shows the functions of the E1 cycle. Contents of the A- and Q-registers are exchanged so that the multiplier is set into the Q-register. Then the A-register is reset to 0. The 'B bit 0 latch' FF is turned on, if the multiplicand is negative, and cannot be turned off until the I1 cycle of the next instruction. Thus, the FF retains the sign of the multiplicand throughout the multiply operation. The 'add' FF is turned on to simulate a previous add action. The E1 cycle provides all shifting right of the A- and Q-registers that is necessary to set the lowest-order 1 of the multiplier into the Q15 position. Each time the contents of A and Q are shifted one position, the CCC is decremented by 1. Details of shifting are given in the description of shift instructions. Note that, if the multiplier contains no 1, the complete "multiply by 0" is accomplished by shifting in an E1 cycle with extended T7's. The 'shift control trigger' FF is turned off when the 'CCC count 0' line becomes active (CCC decremented to 0).

| Multiplier Q14 | Q15 | Previous Action | New Action | Explanation |
|---|---|---|---|---|
| 0 | 0 | Add | Shift | |
| 0 | 1 | Add | Add, Shift | Single One in String of Zeros |
| 1 | 0 | Add | Shift | |
| 1 | 1 | Add | Sub, Shift | Start String of Ones |
| 0 | 0 | Sub | Add, Shift | End of String of Ones |
| 0 | 1 | Sub | Shift | |
| 1 | 0 | Sub | Sub, Shift | Single Zero in String of Ones |
| 1 | 1 | Sub | Shift | |

13227

Figure 3-11. Rules for Multiplication

(-1) · (+14) = - 14

| | A Reg | Q Reg | |
|---|---|---|---|
| E1 Cycle | 0 0 1 1 1 0 | X X X X X X | Set M'cand (111111) in D Reg, B0 = 1 |
| | 0 0 0 0 0 0 | 0 0 1 1 1 0 | Set M'plier in Q Reg, reset A Reg |
| | 0 0 0 0 0 0 | 0 0 0 1 1 1 | Shift, recognize low order 1, enter E2, CCC to 5 |
| 1st E2 Cycle | 0 0 0 0 0 1 | 0 0 0 1 1 1 | Subtract, because Q14 = 1 |
| | 0 0 0 0 0 0 | 1 0 0 0 1 1 | Shift only, CCC to 4, Q15 = 1 |
| | 0 0 0 0 0 0 | 0 1 0 0 0 1 | Shift only, CCC to 3, Q15 = 1 |
| | 0 0 0 0 0 0 | 0 0 1 0 0 0 | Shift only, CCC to 2, Q15 = 0, enter E2 |
| 2nd E2 Cycle | 1 1 1 1 1 1 | 0 0 1 0 0 0 | Add, last action Subtract, Set neg arith sign |
| | 1 1 1 1 1 1 | 1 0 0 1 0 0 | Shift, CCC to 1, Q15 = 0 |
| | 1 1 1 1 1 1 | 1 1 0 0 1 0 | Shift, Q15 = 0, CCC = 0, End op |

(+31) · (-25) = -775

| | A Reg | Q Reg | |
|---|---|---|---|
| E1 Cycle | 1 0 0 1 1 1 | X X X X X X | Set M'cand (011111) in D Reg, B0 = 0 |
| | 0 0 0 0 0 0 | 1 0 0 1 1 1 | Set M'plier in Q Reg, reset A reg, Q15 = 1, Enter E2 |
| 1st E2 Cycle | 1 0 0 0 0 1 | 1 0 0 1 1 1 | Subtract, because Q14 = 1, Set neg arith sign |
| | 1 1 0 0 0 0 | 1 1 0 0 1 1 | Shift, CCC to 5, Q15 = 1 |
| | 1 1 1 0 0 0 | 0 1 1 0 0 1 | Shift, CCC to 4, Q15 = 1 |
| | 1 1 1 1 0 0 | 0 0 1 1 0 0 | Shift, CCC to 3, Q15 = 0, Enter E2 |
| 2nd E2 Cycle | 0 1 1 0 1 1 | 0 0 1 1 0 0 | Add, because Q14 = 0 |
| | 0 0 1 1 0 1 | 1 0 0 1 1 0 | Shift, CCC to 2, Q15 = 1 |
| | 0 0 0 1 1 0 | 1 1 0 0 1 1 | Shift, CCC to 1, Q15 = 0, Enter E2 |
| 3rd E2 Cycle | 1 0 0 1 1 1 | 1 1 0 0 1 1 | Subtract, Set neg arith sign |
| | 1 1 0 0 1 1 | 1 1 1 0 0 1 | Shift, CCC = 0, End op |

13228

Figure 3-12. Examples of Multiply Operations

When a 1 is present in Q15, the 'start multiply arithmetic action' line is activated. No more shifting occurs in the E1 cycle, because the low-order 1 of the multiplier is recognized. The next T0 time then causes entry into an E2 cycle. The 'E1' FF is turned off, and 'E' FF is left on.

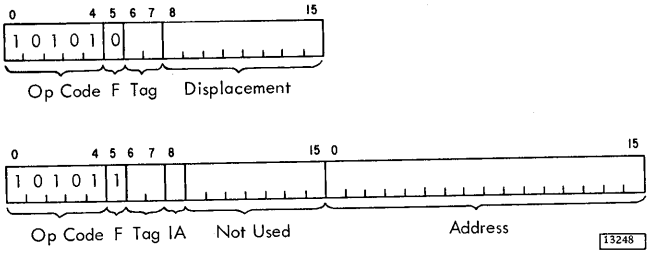
E2 Cycles

Any additions or subtractions required in execution of a 'multiply' instruction are done in E2 cycles. Furthermore, all shifting after the first addition or subtraction is done in E2 cycles. A cycle is required for each addition or subtraction, but once the add or subtract action has been completed, at least one shift occurs in the same E2 cycle. In fact, shifting continues until the next arithmetic action is called for.

The operation during an E2 cycle is dependent upon several variables as shown in maintenance diagram AA673. In any E2 cycle before the CCC has been decremented to 1, the Q14 bit decides whether add or subtract is set. When CCC = 1 the previous action determines which is set for this E2 cycle. Once add or subtract has been set, the state of the 'B bit 0 latch' FF determines whether or not the 'arithmetic sign' FF is turned on. Later, at each shift time, the 'arithmetic sign' FF determines whether 1 or 0 is set into A0 position. As the CCC is decremented for each shift, the counter is checked as to whether it has reached 0. Before the CCC reaches 0, the arithmetic action just completed and the Q15 bit determine whether to continue shifting or to enter another E2 cycle. When the CCC reaches 0, the operation is complete, and the next T0 causes entry into the I1 cycle for the following instruction.

Figure 3-12 shows two examples of multiply operations, using six-position registers. Note that CCC is set to 6 instead of 16 in these examples.

DIVIDE (D)




- The double word already in the A-register and the Q-register is the dividend, and the word at EA is the divisor.
- Each cycle the divisor is read and written back, unchanged.
- At the end of the operation, the quotient is in the A-register and the remainder is in the Q-register.
- 'Overflow' FF is turned on by divide by zero or any division resulting in a quotient which exceeds the capacity of the A-register.
- 'Carry' FF is not affected.
- Maintenance diagram AA674.

Execution of this instruction, when no divide overflow occurs, requires 18 E-cycles -- one E1 and seventeen E2 cycles. The dividend must already be present in the A- and Q-registers as a 32-bit double precision word. The A-register contains the high-order bit positions and, when the dividend is negative, a bit 0 position contains a 1. The divisor is read from EA during each E cycle of the operation. The algorithm used for the divide operation is described and illustrated in maintenance diagram AA674. At the end of the operation, the quotient is in the A-register, and the remainder is in the Q-register. Figure 3-13 shows two divide operations, one requiring quotient correction, and one requiring remainder correction.

The 'carry' FF is not affected by the divide operation. However, any time the quotient exceeds the capacity of the A-register, the 'overflow' FF is turned on. Once turned on, 'overflow' can only be turned off by a 'store status' instruction, by testing while executing a conditional 'BSI' or 'BSC,' or by a 'load status' instruction. Checks for overflow conditions are made during the E1 cycle and the first and last E2 cycles.

E1 Overflow Checks

At T2 time of the E1 cycle, the A-register is checked for a 1 in A bit 0 position and a 0 in A bit 1 position. If these conditions are true, the dividend is a 31-position negative number. Division, even by the largest 15-position divisor, would result in at least a 16-position quotient, exclusive of sign. Such a quotient exceeds the capacity of the A-register and represents an overflow condition. The 'overflow' FF is turned on by activating 'set divide overflow E1 E2,' and the same line resets the CCC to 0. The divide operation stops at the end of the E1 cycle.

When the divisor is 0, the quotient must exceed the capacity of the A-register. At phase B of T3 time, during the E1 cycle, a D-register equal to 0 condition activates the same 'set divide overflow E1 E2' line. This line turns on the 'overflow' FF and resets the CCC to 0. At the end of E1, the divide operation stops and the next I1 is entered.

E2 Overflow Checks

At T1 time of the first E2 cycle, the value in the CCC is decremented to 16. (CCC is set to 18 during I1; then it is decremented to 17 during the E1 cycle.) At T2 time, another test for overflow conditions is made. One set of conditions turns on the 'overflow' FF and resets the CCC to 0, causing the divide operation to end after the first E2. These conditions are:

1. A-register does not equal 0.
2. 'B bit 0 latch' FF is on, indicating a negative divisor.
3. 'Temporary carry' FF is on, as a result of add or subtract action during the E1 cycle.

Other conditions at T2 of the first E2 cycle may turn on the 'overflow' FF without resetting the CCC. Consequently, division continues in the usual manner, except that the 'overflow' FF is on. The conditions tested are the same as those tested at T7 end operation time in execution of an 'add' or 'subtract'

(+57) ÷ (-5) = -11, with Remainder of +2  Divisor = 111011

| | A Reg | Q Reg | CCC | A0 = B0 | Action |
|---|---|---|---|---|---|
| E1 | 0 0 0 0 0 0 | 1 1 1 0 0 1 | 8 | No | CCC to 7, Set Add & Unlike Sign |
| | 0 0 0 0 0 1 | 1 1 0 0 1 0 | 7 | | Shift, Never set Q15 to 1 in E1 |
| | 1 1 1 1 0 0 | 1 1 0 0 1 0 | | Yes | Add |
| E2 | 1 1 1 0 0 1 | 1 0 0 1 1 0 | 6 | | CCC to 6, Set Subtract & Q15 to 1, Shift |
| | 1 1 1 1 1 0 | 1 0 0 1 1 0 | | Yes | Subtract |
| E2 | 1 1 1 1 0 1 | 0 0 1 1 1 0 | 5 | | CCC to 5, Set Subtract & Q15 to 1, Shift |
| | 0 0 0 0 1 0 | 0 0 1 1 1 0 | | No | Subtract |
| E2 | 0 0 0 1 0 0 | 0 1 1 1 0 0 | 4 | | CCC to 4, Set Add, Shift |
| | 1 1 1 1 1 1 | 0 1 1 1 0 0 | | Yes | Add |
| E2 | 1 1 1 1 1 0 | 1 1 1 0 1 0 | 3 | | CCC to 3, Set Subtract & Q15 to 1, Shift |
| | 0 0 0 0 1 1 | 1 1 1 0 1 0 | | No | Subtract |
| E2 | 0 0 0 1 1 1 | 1 1 0 1 0 0 | 2 | | CCC to 2, Set Add, Shift |
| | 0 0 0 0 1 0 | 1 1 0 1 0 0 | | No | Add |
| E2 | 0 0 0 0 1 0 | 1 1 0 1 0 0 | 1 | | CCC to 1, Reset Arith Ctrl, No 0 Remainder |
| | 0 0 0 0 1 0 | 1 1 0 1 0 0 | | -- | No Shift or Arithmetic Action |
| E2 | 1 1 0 1 0 0 | 0 0 0 0 1 0 | 0 | | CCC to 0, Set Add and Arith Ctrl, Exchange A & Q |
| | 1 1 0 1 0 1 | 0 0 0 0 1 0 | | -- | Set D15 to 1, Add, End Op |

(-7) ÷ (-3) = +2, with Remainder of -1  Divisor = 111101

| | A Reg | Q Reg | CCC | A0 = B0 | Action |
|---|---|---|---|---|---|
| E1 | 1 1 1 1 1 1 | 1 1 1 0 0 1 | 8 | Yes | CCC to 7, Set Subtract |
| | 1 1 1 1 1 1 | 1 1 0 0 1 0 | 7 | | Shift, Never set Q15 to 1 in E1 |
| | 0 0 0 0 1 0 | 1 1 0 0 1 0 | | No | Subtract |
| E2 | 0 0 0 1 0 1 | 1 0 0 1 0 0 | 6 | | CCC to 6, Set Add, Shift |
| | 0 0 0 0 1 0 | 1 0 0 1 0 0 | | No | Add |
| E2 | 0 0 0 1 0 1 | 0 0 1 0 0 0 | 5 | | CCC to 5, Set Add, Shift |
| | 0 0 0 0 1 0 | 0 0 1 0 0 0 | | No | Add |
| E2 | 0 0 0 1 0 0 | 0 1 0 0 0 0 | 4 | | CCC to 4, Set Add, Shift |
| | 0 0 0 0 0 1 | 0 1 0 0 0 0 | | No | Add |
| E2 | 0 0 0 0 1 0 | 1 0 0 0 0 0 | 3 | | CCC to 3, Set Add, Shift |
| | 1 1 1 1 1 1 | 1 0 0 0 0 0 | | Yes | Add |
| E2 | 1 1 1 1 1 1 | 0 0 0 0 1 0 | 2 | | CCC to 2, Set Subtract & Q15 to 1, Shift |
| | 0 0 0 0 1 0 | 0 0 0 0 1 0 | | No | Subtract |
| E2 | 0 0 0 0 1 0 | 0 0 0 0 1 0 | 1 | | CCC to 1, Set Add, Change 0 Remainder |
| | 1 1 1 1 1 1 | 0 0 0 0 1 0 | | -- | No Shift, Add Divisor |
| E2 | 0 0 0 0 1 0 | 1 1 1 1 1 1 | 0 | | CCC to 0, Set Add & Arith Ctrl, Exchange A & Q |
| | 0 0 0 0 1 0 | 1 1 1 1 1 1 | | -- | D15 = 0, Add, End Op |

13229

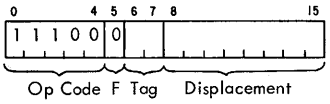
Figure 3-13. Examples of Divide Operations

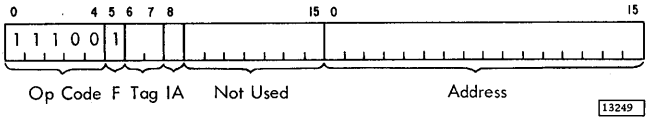
instruction. The conditions are shown in a chart in maintenance diagram AA671, and the temporary carry or borrow depends on the add or subtract action during the preceding E1 cycle.

End Operation Overflow Check

At the end of a divide operation, the 'overflow' FF is turned on if a quotient other than 0 does not have the correct sign. The 'arithmetic sign' FF and the 'A bit 0' FF are, in effect, fed to an exclusive OR circuit. To prevent activating 'overflow gate' which gates turning on the 'overflow' FF, both the 'arithmetic sign' FF and the 'A bit 0' FF must be in the same state. In other words, if the dividend and divisor have different signs ('arithmetic sign' FF on), the quotient must be minus ('A bit 0' FF on).

AND (AND)




- The word read from the EA is ANDed with the word in the A-register and then written back into EA, unchanged.
- The result is in the A-register.
- The word from core storage is first set into the D-register.
- When an A-register FF is on, and the corresponding D-register FF is also on, the A-register FF is left on. All other A-register FF's are turned off.
- 'Carry' FF and 'overflow' FF are not affected.
- Maintenance diagram AA691.

Execution of this instruction requires an E1 cycle following the I-cycles necessary to develop the EA. At T2 time the word read from core storage at the EA is set into the D-register. During T3, the 'AND SPD gate' is activated by op code decoding. Very early in T4 'phase A SP B' fires the 'AND SP.' Each D-register FF that is off gates the turning off of the corresponding A-register FF. Although a 'reset D register SP' is also activated, any D-register FF that is on does not turn off in time to gate turning off the corresponding A-register FF. Thus, the action which occurs is:

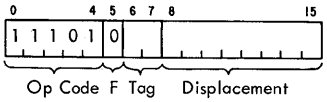
1. D on, corresponding A on: no action.
2. D on, corresponding A off: no action.
3. D off, corresponding A on: A receives sample pulse that turns it off.
4. D off, corresponding A off: A receives sample pulse to turn it off, but pulse has no effect.

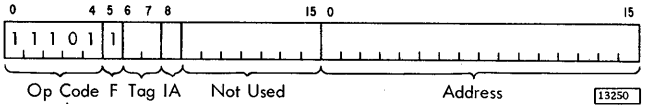
Examples:

| | |
|---|---|
| D-register | 1100 |
| A-register | 1010 |
| Result in A | 1000 |

The operation is complete, and the next T0 causes entry into an I1 cycle.

OR (OR)




- The word read from the EA is ORed with the word in the A-register and written back into EA, unchanged.
- The result is in the A-register.
- The word from core storage is first set into the D-register.
- For each D-register FF that is on, turn on the corresponding A-register FF. Leave on any A-register FF that is already on.
- 'Carry' FF and 'overflow' FF are not affected.
- Maintenance diagram AA961.

Execution of this instruction requires an E1 cycle following the I-cycles necessary to develop the EA.

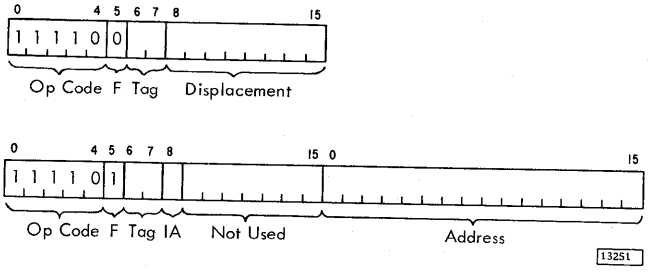
At T2 time the word read from core storage at the EA is set into the D register. During T3, the 'add/subtract/OR/EOR gate' and the 'reset D SPD gate' are activated by op code decoding. At the start of T4, the 'reset D reg SP' is fired, turning off any D-register FF that is on. The resulting shift turns on the corresponding A-register FF, because circuit delays are keeping 'add/subtract/OR/EOR gate' active. No A-register FF's are turned off, and the operation has been completed.

Examples:

| | |
|---|---|
| D-register | 1100 |
| A-register | 1010 |
| Result in A | 1110 |

The next T0 causes entry into the next I1 cycle.

EXCLUSIVE OR (EOR)




- The word read from the EA is exclusive ORed with the word in the A-register and written back into EA, unchanged.
- The result is in the A-register.
- The word from core storage is first set into the D-register.
- Turning off any D-register FF changes the state of the corresponding A-register FF.
- 'Carry' FF and 'overflow' FF are not affected.
- Maintenance Diagram AA691.

Execution of this instruction requires an E1 cycle following the I-cycles necessary to develop the EA. At T2 time, the word read from core storage at the EA is set into the D-register. During T3, three gates are activated by op code decoding -- 'add/subtract/OR/EOR gate,' 'add/subtract/EOR gate,' and 'reset D SPD gate.' When the D-register is reset early in T4, turning off D-register FF results in changing the state of the corresponding A-register FF. Circuit delays keep the A-register control gates active until after the 'reset D reg SP.'

Examples:

| | |
|---|---|
| D-register | 1100 |
| A-register | 1010 |
| Result in A | 0110 |

The operation is complete and the following T0 causes entry into the next I1 cycle.

## SHIFT LEFT INSTRUCTION GROUP

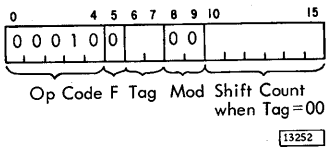
- All 'shift left' instructions are valid in short format only (F = 0).
- The op code is the same for all 'shift left' instructions.
- The operation is further defined by the tag and modifier bits, positions 6-7 and 8-9, respectively.
- Maintenance diagram AA631

'Shift left' instructions cause the binary digits in the A-register (or the A- and Q-registers) to be shifted to the left one position at a time. These instructions are valid in short format only, and no EA is required. All the 'shift left' operations are initiated by the same op code, and bit 5 is always a 0. The tag (positions 6 and 7) and modifiers (positions 8 and 9) further define shift left operation.

The tag determines whether the CCC is set from positions 10-15 of the instruction word or from those positions of an index register. When an index register is used, an IX cycle occurs to set the CCC. Each shift decrements the CCC by 1, and the CCC reaching 0 stops the shift operation. An instruction which initially sets the CCC to 0 causes no shifting and acts as a 'no-op' instruction.

Modifiers 8 and 9 modify the operations as described in the following paragraphs.

### SHIFT LEFT ACCUMULATOR (SLA)




- The binary digits in the A-register are shifted to the left the number of positions specified (shift count).
- When tag = 00, the shift count in positions 10-15 of the instruction is transferred to the CCC.
- When tag ≠ 00, positions 10-15 of the selected index register contain the shift count. An IX cycle transfers the shift count to the CCC.
- The 'A bit 15' cannot be turned on, so vacated positions always contain 0's after execution of this instruction.
- Shifting a 1 out of the high order position (A bit 0) turns on the 'carry' FF. Shifting a 0 out turns off the 'carry' FF.
- The 'overflow' FF is not affected.
- Maintenance diagram AA631.

Execution of this instruction is completed during I1 cycle when the tag = 00, although extended T7 times may be required. When the tag ≠ 00, an IX cycle follows, the I1 cycle. Then the shift operation is completed during the IX cycle, with extended T7 times possibly required.

At T3 of I1 cycle, a U-to-A transfer occurs for shift instructions. Remember that at T1 the contents of A were transferred to the U-register in case the accumulator should be required for EA computation. However, shift instructions require no EA computation, because they are executed on the contents of the A-register.

'Tag 00' and 'I1 cycle' activate 'gate B10-B15 to CCC' at T3 time, 'B bit 10' to 'B bit 15' FF's which are on turn on the 'CCC 32' to 'CCC 1' FF's as follows:

| B bit FF | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|
| CCC FF | 32 | 16 | 8 | 4 | 2 | 1 |

Thus, positions 10 through 15 of the instruction word provide the shift count when tag = 00. Setting the CCC to 0 at this time allows a T3 phase B pulse to turn off the 'shift control trigger' FF. Later, CCC = 0 allows an end operation condition at T7. This SLA instruction, with a shift count of 0, is effectively a 'no op' instruction.

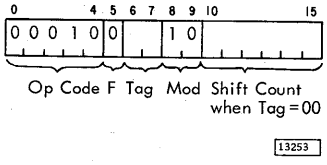
Tag not 00 prevents the transfer of B10-B15 to the CCC. The usual I1 function of loading CCC = 1 occurs at T5, and prevents an end operation at T7. The next T0 turns on the 'IX' FF, and an IX cycle starts. Selection of the index register and reading it out proceeds as in any IX cycle. As usual, the contents are set into the B-register. At T3 of the IX

cycle 'gate B10-B15 to CCC' transfers the shift count to the CCC. This is the same action as occurred during I1 when the tag was 00.

Once the shift count is set into the CCC, shifting continues in the same manner whether the tag is 00 or not. The only difference is the type of cycles, I1 when the tag is 00, and IX when the tag is not 00. The 'shift control trigger' FF, turned on at T3, remains on until the CCC is decremented to 0. Each 'write gate phase A' activates a 'shift left sample' resulting in a 'shift left A SP 0-7' and a 'shift left A SP 8-15.' Each 'shift sample' activates 'CCC decrement sample,' which decreases the shift count in the CCC by one.

The T-clock advances to T7 and stays at T7 as long as the 'shift control trigger' is on. The phase B, following the CCC reaching 0, turns off the 'shift control trigger' FF. The SLA execution is complete, and the next I1 cycle starts at T0.

## SHIFT LEFT ACCUMULATOR AND EXTENSION (SLT)



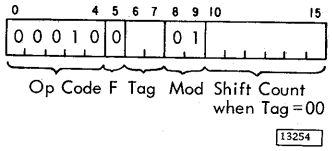

- The 1 in bit position 8 causes the instruction to treat the contents of the A- and Q-registers as one 32-bit double word.
- SLT shifts the bits of the double word to the left the number of positions specified (shift count).
- When tag = 00, positions 10-15 of the instruction contain the shift count, which is transferred to the CCC during I1.
- When tag ≠ 00, positions 10-15 of the selected index register contain the shift count, which is transferred to the CCC during an IX cycle.
- 'Q bit 15' cannot be turned on; so vacated positions always contain 0's after execution of this instruction.
- Bits shift from Q bit 0 position into A bit 15 position.
- Shifting a 1 out of the high order position (A bit 0) turns on the 'carry' FF. Shifting a 0 out turns off the 'carry' FF.
- The 'overflow' FF is not affected.
- Maintenance diagram AA631.

Execution of the SLT instruction is so nearly the same as execution of an 'SLA' instruction that only the differences are described here.

The 1 in bit 8 position turns on the 'modifier 8' FF, actively the 'modifier 8' lines. One of these lines gates the firing of the 'shift left Q SP' lines for 0-7 and 8-15. These are the sample pulses which cause shifting through the Q register. Another 'modifier 8' line allows activating the 'accumulator bit 15 shift left entry' line if the 'Q bit 0' FF is on. Thus, bits shifted out of Q bit 0 position enter the A bit 0 position.

All other functions of the SLT instruction are the same as those of the SLA instruction.

## SHIFT LEFT AND COUNT ACCUMULATOR (SLCA)




- When tag = 00, the operation is the same as SLA.
- When tag ≠ 00, the selected index register contains a shift count for transfer to the CCC during an IX cycle.
- Shifting stops when a 1 is sensed in A bit 0 position or the CCC contains 0.
- Stopping the shift operation by a 1 in A bit 0 position turns on the 'carry' FF. The shift count remainder is set into the selected index register.
- Stopping the shift operation by CCC reaching 0 leaves the index register reset to 0, and leaves the 'carry' FF turned off.

- When a 1 in A bit 0 position and CCC = 0 occur at the same shift, the CCC = 0 controls the operation ('carry' does not turn on).
- 'A bit 15' cannot be turned on, so vacated positions always contain 0's after execution of this instruction.
- The 'overflow' FF is not affected.
- Maintenance diagram AA631.

Execution of this instruction when the tag is 00 can be completed during the I1 cycle, possibly requiring extended T7 times. Operation is the same as that of an SLA instruction with a tag of 00.

When an index register is selected by a tag of 01, 10, or 11, the operation may be completed in the resulting IX cycle. The selected XR contains a shift count which is transferred to the CCC in the same manner as in executing any shift left instruction. By T2 of the IX cycle, the B-register has been set with the contents of the selected index register. During T3, after the shift count has been set into CCC, the 'T3' sample line activates 'SLC reset carry.' This line resets both the 'carry' FF, and B-register 8-15 positions. Thus, when the index register is written back during the IX cycle, positions 0-7 are unchanged, but positions 8-15 have no 1's written into them.

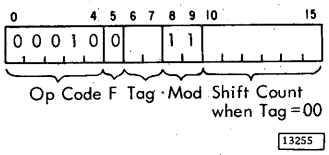
Shifting starts at T4 of the IX cycle, and continues until stopped by a 1 in the A bit 0 position or by the CCC being decremented to 0. If the CCC reaches 0 before or on the same shift that the first 1 is sensed in A bit 0 position, the 'shift control trigger' turns off. T7 then activates 'T7 count 0,' and the next 'clock advance SP A' causes an 'end op T0 SP.' The next I1 cycle starts.

If a 1 is sensed in A bit 0 position before the CCC reaches 0, activating the 'SLC bit' line allows a phase B pulse to turn off 'shift control trigger.' The cycle timer circuits cause entry into an E1 cycle at the next T0.

The purpose of the E1 cycle at this time is to set the remaining shift count from CCC back into the selected XR positions 10-15. The fact that an IX cycle has just been completed in the SLC operation prevents turning off the 'IX inhibit SAR' FF. Therefore, the tag bits address the same index register that was addressed during the preceding IX cycle. 'SLC E' and 'T1' cause an A-to-U transfer to save the contents of the A-register. Then the A-register is reset. 'T2,' 'SLC operation,' and 'E cycles' activate 'gate CCC to A10-A15,' causing the transfer of the remaining shift count to the low order positions of the A register. The same 'gate CCC to A10-A15' turns on the 'carry' FF, indicating that shifting was stopped by a 1 in A bit 0 position. (Remember that this was the cause for entering an E1 cycle.) Transferring A to B8-15 and writing the contents of the B-register into storage sets the shift count remainder into the selected index register.

The objectives of the instruction have been accomplished, so the CCC is reset to allow an end op condition at T7. 'SLC E' and 'T7 end op' activate the 'U to A SPD sample' to cause the return to the A register of its contents which had been saved in the U-register. The next T0 then causes entry into the following I1 cycle.

## SHIFT LEFT AND COUNT ACCUMULATOR AND EXTENSION (SLC)




- The 1 in bit position 8 causes the instruction to treat the contents of the A- and Q-registers as one 32-bit double word.
- When tag = 00, the operation is the same as 'SLT.'
- When tag $\neq$ 00, the selected index register contains a shift count for transfer to the CCC during an IX cycle.
- Q bit 0 position bits shift into A bit 15 position.
- 'Q bit 15' cannot be turned on; so vacated positions always contain 0's after execution of this instruction.
- Shifting stops when a 1 is sensed in A bit 0 position or the CCC contains 0.
- Stopping the shift operation by a 1 in A bit 0 position turns on the 'carry' FF. The shift count remainder is set into the selected index register.
- Stopping the shift operation by CCC reaching 0 leaves the index register reset to 0, and leaves the 'carry' FF turned off.

- When a 1 in A bit 0 position and CCC = 0 occur at the same shift, CCC = 0 controls the operation. ('Carry' does not turn on.)
- The 'overflow' FF is not affected.
- Maintenance diagram AA631.

Execution of this instruction only varies from that of an SLCA in the effects of the 'modifier 8' FF being on. Refer to the description of shift left accumulator and extension (SLT) for the functions of the 'modifier 8' FF.

## SHIFT RIGHT INSTRUCTION GROUP

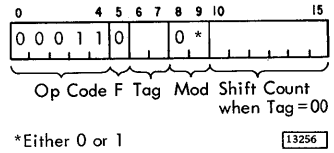
- All 'shift right' instructions are valid in short format only (F = 0).
- The op code is the same for all 'shift right' instructions.
- The operation is further defined by the tag and modifier bits, positions 6-7 and 8-9, respectively.
- Maintenance diagram AA632.

'Shift right' instructions cause the binary digits in the A-register (or the A- and Q-registers) to be shifted to the right one position at a time. These instructions are valid in short format only, and no EA is required. All the shift right operations are initiated by the same op code, and bit 5 is always a 0. The tag (positions 6 and 7) and modifiers (positions 8 and 9) further define shift right operation.

The tag determines whether the CCC is set from positions 10-15 of the instruction word or from those positions of an index register. When an index register is used, an IX cycle occurs to set the CCC. Each shift decrements the CCC by 1, and the CCC reaching 0 stops the shift operation. An instruction which initially sets the CCC to 0 causes no shifting and acts as a no op code instruction.

Modifiers 8 and 9 modify the operations as described in the following paragraphs.

### SHIFT RIGHT ACCUMULATOR (SRA)




- The binary digits in the A-register are shifted to the right the number of positions specified (shift count).
- When tag = 00, the shift count in positions 10-15 of the instruction is transferred to the CCC.
- When tag ≠ 00, positions 10-15 of the selected index register contain the shift count. An IX cycle transfers the shift count to the CCC.
- The 'A bit 0' is never turned on; so vacated positions contain 0's after execution of this instruction.
- Bits shifted out of 'A bit 15' are lost.
- 'Carry' FF and 'overflow' FF are not affected.
- Maintenance diagram AA632.

Execution of this instruction is completed during the I1 cycle when the tag = 00, although extended T7 times can be required. When the tag ≠ 00, an IX cycle follows the I1 cycle. Then the shift operation is completed during the IX cycle, with extended T7 times possibly required.

At T3 of I1 cycles, a U-to-A transfer occurs for shift instructions. Remember that at T1 the contents of the A-register were transferred to the U-register in case the accumulator should be required for EA computation. However, shift instructions require no EA computation, because they are executed on the contents of the A-register.

'Tag 00' and I1 cycle' activate 'gate B10-B15 to CCC' at T3 time. 'B bit 10' to 'B bit 15' FF's which are on turn on the 'CCC 32' to 'CCC1' FF's as follows:

| B bit FF | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|
| CCC FF | 32 | 16 | 8 | 4 | 2 | 1 |

Thus, positions 10 through 15 of the instruction word provide the shift count when tag = 00. Note that setting the CCC to 0 at this time allows a T3 phase B pulse to turn off the 'shift control trigger' FF. Later, CCC = 0 allows an end operation condition at T7. This SRA instruction, with a shift count of 0, is effectively a 'no op' instruction.

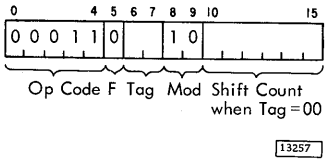
Tag not 00 prevents the transfer of B10-B15 to the CCC. The usual I1 function of loading CCC = 1 occurs at T5 and prevents an end operation at T7. The next T0 turns on the 'IX' FF, and an IX cycle starts. Selection of the index register and reading it out proceeds as in any IX cycle. As usual the contents are set into the B-register. At T3 of the IX cycle 'gate B10-B15 to CCC' transfers the shift count to the CCC. This is the same action as occurred during I1 when the tag was 00.

Once the shift count is set into the CCC, shifting continues in the same manner whether the tag is 00 or not. The only difference is the type of cycle: I1 when the tag is 00, IX when the tag is not 00. The 'shift control trigger' FF, turned on at T3, remains on until the CCC is decremented to 0. Each 'write gate phase A' activates a 'shift right sample'

resulting in a 'shift right A SP 0-7' and a 'shift right A SP 8-15.' These lines cause the shifting of bits within the A-register. Each 'shift sample' activates 'CCC decrement sample,' which decreases the shift count in the CCC by one.

The T-clock advances to T7 and stays as long as 'shift control trigger' is on. The phase B following the CCC reaching 0 turns off the 'shift control trigger' FF. The 'SRA' execution is complete, and the next I1 cycle starts at T0.

SHIFT RIGHT ACCUMULATOR AND EXTENSION (SRT)

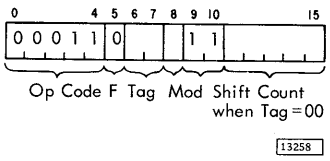



- The 1 in bit position 8 causes the instruction to treat the contents of the A- and Q-registers as one 32-bit double word.
- SRT shifts the bits of the double word to the right the number of positions specified (shift count).
- When tag = 00, load the CCC, during I1, with the shift count from positions 10-15 of the instruction.
- When tag ≠ 00, positions 10-15 of the selected index register contain the shift count. An IX cycle transfers the shift count to the CCC.
- When a 1 is shifted right from A bit 0 position, the 'A bit 0' FF turns back on to maintain the negative sign.
- Bits shift from A bit 15 position into Q bit 0 position.
- Bits shifted out of the Q bit 15 position are lost.
- 'Carry' FF and 'overflow' FF are not affected.
- Maintenance diagram AA632.

Only the differences in operation caused by the 'modifier 8' FF being on need be described. One 'modifier 8' line activates the 'shift right Q SPD gate' which allows shift right sample pulses to fire the 'shift right Q SP 0-7' and 'shift right Q SP 8-15' drivers. These lines cause shifting through the Q-register. Another 'modifier 8' line allows activating the 'accumulator bit 0 shift right entry' if 'modifier 9' is off and 'A reg bit 0' is on. Thus, when A reg 0 contains a 1 at the start of the operation, each shift pulse turns it back on, maintaining a negative sign.

Other functions of the SRT instruction are the same as those of an SRA instruction.

ROTATE RIGHT (RTE)




- The 1 in bit position 8 causes the instruction to treat the contents of the A- and Q-registers as one 32-bit double word.
- RTE shifts the bits of the double word to the right the number of positions specified (shift count).
- When tag = 00, load the CCC, during I1, with the shift count from positions 10-15 of the instruction.
- When tag ≠ 00, positions 10-15 of the selected index register contain the shift count. An IX cycle transfers the shift count to the CCC.
- Bits shift from A bit 15 position into Q bit 0 position.
- Bits shift out of Q bit 15 into A bit 0 position.
- A shift count of 16 or 48 results in an exchange of the contents of the A and Q registers. Bits are not shifted one position at a time.
- 'Carry' FF and 'overflow' FF are not affected.
- Maintenance diagram AA632.

Execution of this instruction is like that of an SRT instruction except for the differences caused by the

'modifier 9' FF being on. In fact the only time 'modifier 9' lines are effective in a shift right operation is when 'modifier 8' lines are also active. (This is not true in shift left operations.) A 'modifier 9' line blocks the circuit to activate 'accumulator bit 0 shift right entry' from 'A register bit 0.' Instead, a 'modifier 9' line allows activating 'accumulator bit 0 shift right entry' when 'Q bit 15' is active. As a result, bits shift from Q bit 15 position into A bit 0 position.

A shift count of 16 or 48 is recognized when 'CCC 16' is on and 'CCC8,' 'CCC4,' 'CCC2,' and 'CCC1,' are all off. If the shift count set into the CCC at T3 is 16 or 48, the 'shift right rotate 16/48' lines become active. 'T3 phase B' turns off the 'shift control trigger.' Then a 'write gate phase A' activates 'exchange A and Q SPD sample,' firing four SPD's. Each SPD causes the exchange of bits in four positions of the A- and Q-registers 'Exchange A and Q SPD sample' also resets the 'CCC 16' and 'CCC 32' FF's.

A shift count between 16 and 48 rotates the bits until the CCC reaches 16. Then the 'shift right rotate 16/48' line becomes active, and the contents of A and Q are exchanged as described in the previous paragraph. A shift count greater than 48 rotates the bits until the CCC reaches 48. Then the exchange occurs.

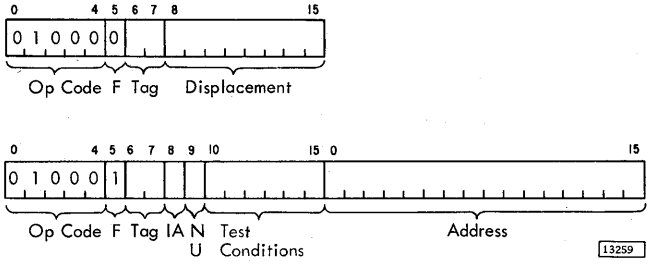
When the CCC = 0, and the clock is at T7, the operation ends. The next T0 causes entry into I1 cycle.

## BRANCH OR SKIP INSTRUCTION GROUP

- Branch or skip instructions may be unconditional or conditional.
- Unconditional branch or skip instructions always cause the next instruction to be obtained from a location other than the next higher-numbered location of storage.
- Conditional branch or skip instructions can alter the program execution sequence, depending on the results of testing specified conditions.

Usually, at the end of execution of an instruction, the IAR contains an address that is one higher than the address of the instruction. In the case of a long-format instruction, IAR contains an address one higher than that of the second word of the instruction. Execution of an unconditional branch or skip instruction sets some other address into IAR. Therefore, the instruction following the branch or skip instruction is obtained from a location other than the next higher location. Execution of a conditional branch or skip instruction changes IAR only when the branch or skip is to take place. Conditions of the accumulator, carry indicator, and overflow indicator are tested when specified by the instruction (Figure 3-14). Depending on the instruction, results of testing these conditions determine whether or not a branch or skip occurs.

## BRANCH AND STORE IAR (BSI)




| Displacement Bit Position | Condition Tested |
|---|---|
| 15 | Overflow Indicator Off |
| 14 | Carry Indicator Off |
| 13 | Accumulator Even |
| 12 | Accumulator Plus (>0) |
| 11 | Accumulator Negative (<0) |
| 10 | Accumulator = Zero |

24039

Figure 3-14. Test Conditions

- The effective address is obtained or computed during I-cycles. Both indexing and indirect addressing may be used.
- BSI stores the contents of IAR at the core storage location selected by the EA.
- The short format (F = 0) instruction causes an unconditional branch and store.
- The long format (F = 1) instruction causes a branch and store operation, only if none of the conditions tested is true.
- The 'carry' FF and accumulator are not affected by testing.
- The 'overflow' FF is turned off by testing.
- Maintenance diagram AA651.

### Operation When F = 0

Execution of the short format BSI instruction requires an E1 cycle following the I-cycles which compute the EA. This E1 cycle:

1. Sets present value from IAR into core storage at EA.
2. Sets IAR to EA + 1.

During T2 time, the BSI activates 'I to B SPD gate,' so that the end of T2 fires 'I to B SP 0-7' and 'I to B SP 8-15.' These sample pulses set the contents of IAR into the B-register. Remember that IAR contains the address of the BSI instruction incremented +1 during the I1 cycle. Writing the

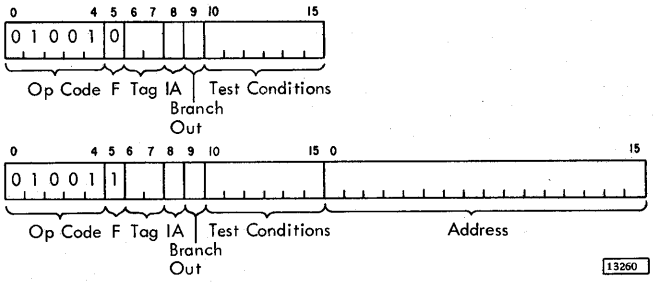
B-register contents into storage completes the first objective of the BSI instruction. During T3 time, the BSI activates 'M to I SPD gate,' which remains active long enough to gate the next 'clock advance SP B.' Then 'M to I SP 0-7' and 'M to I SP 8-15' set the EA into IAR early in T4 time. The end of the 'M to I SPD gate' reactivates 'IAR 0-7 increment gate' and 'IAR 8-15 increment gate.' Then T5 activates 'increment IAR sample' and fires the 'start IAR increment.' This incrementing sets the IAR to EA + 1. The actual branch occurs when the next I1 cycle starts and the new value in IAR reads out the next instruction.

Operation When F = 1

A long format BSI instruction causes certain changes in the functions of the I1 cycle. The I1 cycle tests the conditions of the accumulator, carry indicator, and overflow indicator that are specified in the instruction word. At T2 of the I1 cycle, any 1 in positions 10-15 of the instruction word turns on the corresponding D-register FF. Then any tested condition which is true activates the 'skip condition' line. For example, 'D bit 15' active and the 'overflow' FF off activate 'skip condition.' With 'skip condition' active, an I1T4 pulse turns on the 'skip condition' FF, indicating that no branch and store functions are to occur. The same pulse resets the 'overflow' FF to the off side if its condition is being tested. T5 then activates the 'skip sample' line, when 'skip condition' is on. 'Skip sample' results in incrementing IAR plus 1 for the second time in the I1 cycle. This sets IAR to the address of the instruction following the long format BSI. The CCC has not been changed from 0; so T7 activates end operation, and the next T0 causes entry into the next I1 cycle.

Assuming that no tested condition is true, the 'skip condition' FF is not turned on during I1. Branch and store functions are to be accomplished. I1 cycle is completed in the usual manner, and the CCC is set to 1. A standard I2 cycle and any IX and/or IA cycles, necessary to develop the EA, follow in that order. The remainder of execution (the E1 cycle) is identical to that of a short format BSI.

BRANCH OR SKIP ON CONDITION (BSC AND BOSC)




- During I1 cycle, test the conditions of the accumulator, carry indicator, and overflow indicator, as specified by 1's in positions 10-15 of the instruction word.
- Execution, thereafter, depends on the format.
- Short-format (F = 0) instruction causes skipping past the instruction in the next higher core location when any tested condition is true. That next instruction must be short format also. If modifier 9 is on and a skip occurs, turn off the highest level interrupt FF that is on. The tag bits have no effect on operation.
- Long-format (F = 1) instruction causes branching to the EA when no tested condition is true. If modifier 9 is on and a branch occurs, turn off the highest level interrupt FF that is on.
- 'Overflow' FF is turned off if it is tested.
- 'Carry' FF and accumulator are not affected by testing.
- Maintenance diagram AA652.

The I1 cycle is much the same for either a short-format or a long-format BSC instruction. No EA computation is required for a short-format BSC instruction, and I1 never enters into EA generation for long-format instructions. The first function

| Instruction Word | | | | | | | | Skip Conditions | | | | | | Result |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| F | | Displacement | | | | | | Accumulator Status | | | | Indicators | | |
| 5 | - | 10 | 11 | 12 | 13 | 14 | 15 | Zero | Minus | Plus | Even | Carry | O'flow | |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | | - | No | - | [Yes] | - | - | Skip |
| 0 | 1 | 0 | 1 | 0 | 0 | 1 | | No | - | No | - | - | On | No Skip |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | | - | No | - | - | [Off] | - | Skip |
| 0 | 1 | 0 | 0 | 1 | 0 | 1 | | No | - | - | [Yes] | - | [Off] | Skip |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | | [Yes] | No | No | No | On | On | Skip |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | | - | - | - | - | - | - | No Skip |
| 1 | 1 | 0 | 1 | 1 | 0 | 1 | | No | - | No | [Yes] | - | On | No Branch |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | | - | - | - | - | - | - | Branch |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | | No | [Yes] | No | [Yes] | On | On | No Branch |
| 1 | 1 | 1 | 0 | 1 | 0 | 1 | | No | No | - | No | - | [Off] | No Branch |
| 1 | 0 | 1 | 0 | 1 | 0 | 0 | | - | No | - | No | - | - | Branch |

[ ] Indicates condition causing skip or preventing branch. 24040B

Figure 3-15. BSC Examples

of the I1 cycle for either BSC instruction is to determine if any of the conditions tested are true. Examples of BSC operation appear in Figure 3-15. Testing of the specified conditions is described in the paragraphs concerning the BSI instruction. By T4 of the I1 cycle, the state of the 'skip condition' FF is determined, establishing the operation thereafter.

'Skip Condition' FF On

The operation is completed during the I1 cycle. The address presently in IAR is incremented by 1. For a short format BSC, this results in a skip past the following short format instruction. For a long format BSC, no branch occurs. The address word is skipped, and IAR contains the address of the next sequential instruction. If the 'modifier 9' FF is on and the BSC is short format, the 'branch out' line is activated. Then a 'T7 SP' resets the highest (interrupt) level FF which is on. Contents of the U-register are returned to the A-register, and the next T0 causes entry into an I1 cycle.

'Skip Condition' FF Off

When the 'skip condition' FF is not turned on and the BSC is short format, no skip takes place. In effect, the instruction is a 'no op.' IAR is not incremented the second time during I1, and the program obtains its next instruction from the next higher core storage location.

When 'skip condition' is off and the BSC is long format, a branch takes place. The I1 cycle is followed by the cycles required to compute or obtain the EA. The last I-cycle may be an I2, an IX, or an IA cycle. The end of T5 of that cycle turns on the 'branch 1' FF. If 'modifier 9' is on, the 'branch 1' line completes the conditions to activate 'branch out.' Then a 'T7 SP' turns off the highest level (interrupt) FF. The EA may not yet be completely computed. When arithmetic action is complete and 'branch 1' is on, T7 and phase A turn on 'branch 2.' Maintenance diagram AA652 shows the further functions of T7's or extended T7's. The T-clock cannot advance to T0 until 'arithmetic control trigger,' 'branch 1,'

and 'branch 2' have all been turned off. By that time the EA ("branch to" address) is in IAR, and the original contents of the A-register have been returned by a U-to-A transfer. The CCC is decremented to 0, so that 'end op T0 SP' can cause entry into the next I1 cycle.

Branch Out or Skip on Condition (BOSC)

The operation of a BSC instruction, in which bit position 9 contains a 1, has been described. This instruction is commonly used at the end of an interrupt servicing subroutine to cause a "branch out" from that subroutine. When the branch out occurs, the (interrupt) level FF for that level is turned off. Remember that the CPU always executes the highest level that is active. Thus the highest level FF that is on is turned off by the instruction.

Usually, the instruction is in long format, using indirect addressing. The EA is set equal to the EA of the BSI instruction which caused entry into the subroutine. Execution of the BOSC instruction may result in:

1. Reentry into same level servicing subroutine.
2. Entry into lower level servicing subroutine.
3. Return to mainline program.

Additional description of BOSC operation is given in "Interrupt Circuits" in Chapter 2.

MODIFY INDEX AND SKIP (MDX)

Execution of a short format MDX instruction adds the displacement portion of the instruction to the contents of IAR or the contents of an index register. The tag bits select the index register. The operations and results are different when IAR is used from those when an index register is used.

Execution of a long format MDX provides one of these operations:

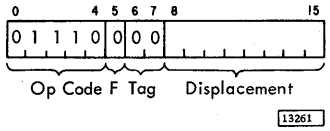
1. Add bits of address word itself to contents of selected XR and store sum in the XR.
2. Add displacement to contents of core storage location specified by address.
3. Add contents of core storage location specified by address to contents of selected XR and store sum in the XR.

The tag bits select the XR which is used in an operation.

Any 'MDX,' except the short-format tag = 00 instruction, may cause a skip past the next one-word instruction. The skip occurs only when the modified XR or core storage location changes sign or is zero after modification.

Operation of each MDX instruction is described in detail in the following paragraphs.

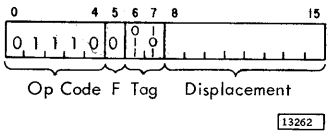
Short Format MDX, Tag = 00




- The displacement is added to the contents of IAR. At the time of addition, IAR contains the address of the instruction incremented by +1.
- The sum is set into IAR, causing an unconditional branch at the next I1 cycle.
- Bit 8 = 1 indicates a negative displacement.
- Original contents of the accumulator are returned at the end of the operation.
- 'Carry' FF and 'overflow' FF are not affected.
- Maintenance diagram AA663, Sheet 1.

Execution of this instruction requires only an I1 cycle. Operation during that I1 cycle proceeds as in the usual I1 cycle through T5 time, when the CCC is set to 1. The end of T5 then turns on the 'branch 1' FF. Completion of the addition of displacement and contents of IAR can occur at any phase A time, and the following phase B turns off the 'arithmetic control trigger.' If this occurs before T7, a 'T7 phase A' pulse turns on the 'branch 2' FF, and an extended T7 time occurs. Otherwise, the next extended T7 phase A, after the 'arithmetic control trigger' FF turns off, turns on 'branch 2.' Maintenance diagram AA663, sheet 1 shows how extended T7 times complete the functions of the MDX. The T-clock cannot advance to T0 until after 'branch 2' turns off. Then the clock advance causes entry into the next I1 cycle.

Short Format MDX, Tag ≠ 00




- The displacement is added to the contents of the index register specified by the tag bits.
- The sum is set back into the specified index register.
- A skip occurs if the sum is zero or the sign in the index register changes.
- Bit 8 = 1 indicates a negative displacement.
- Original contents of the accumulator are returned at the end of the operation.
- 'Carry' FF and 'overflow' FF are not affected.
- Maintenance diagram AA663, Sheet 2.

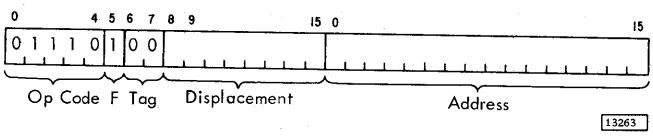
Execution of this instruction requires three cycles: I1, E1 and E2. The I1 cycle places the displacement in the A-register; E1 adds the displacement to the contents of the selected XR; and E2 tests the sum for all 0's or a sign change and sets the sum into the selected XR.

Functions of the I1 cycle are standard. After transferring the contents of the A-register to the U-register to save them, reset A to all 0's. Block the I-to-A transfer, because the tag ≠ 00, and the A-register remains clear. Set the displacement into D-register 8-15 positions, and, if D bit 8 is a 1, set D-register 0-7 positions to 1's. Doing so maintains the sign of a negative displacement. Then add the displacement to the cleared A-register.

The E1 cycle functions almost identically to an IX cycle. The tag register, instead of the M-register, addresses core storage and the contents of the selected index register set the D-register. Also, a 1 in the bit 0 position of the index register turns on the 'B bit 0 latch' FF if the index register contents are negative. Addition of the contents of the A- and D-registers results in the sum being in the A-register.

In the E2 cycle, the tag register continues to address core storage, selecting the XR. T1 decrements CCC to 0, so that the end of the cycle can activate the end operation conditions. At T2, 'A-register equal zero' or '(not) like signs' can activate the 'increment IAR sample.' These are the conditions which cause a skip. T3 transfers the sum from the A-register to the B-register. Write time then sets the sum back into the index register (still selected). The original contents of the A-register are transferred back from the U-register, and T0 causes entry into the next I1 cycle.

Long Format 'MDX', Tag = 00




- The displacement is added to the contents of the core storage location specified by the address (add to storage operation).
- 'Modifier 8' is turned on regardless of bit position 8 of the instruction word. This forces an IA cycle.
- Bit 8 = 1 indicates a negative displacement.
- The sum is set back into the core storage location specified by the address.
- A skip occurs if the sum is zero or if the operation changes the sign in the affected core storage location.
- The original contents of the A-register are returned at the end of the operation.
- 'Carry' FF and 'overflow' FF are not affected.
- Maintenance diagram AA663, sheets 3 and 4.

Execution of this instruction requires I1, I2, IA, E1 and E2 cycles. Only the unusual functions of these cycles, necessary to complete the operation,

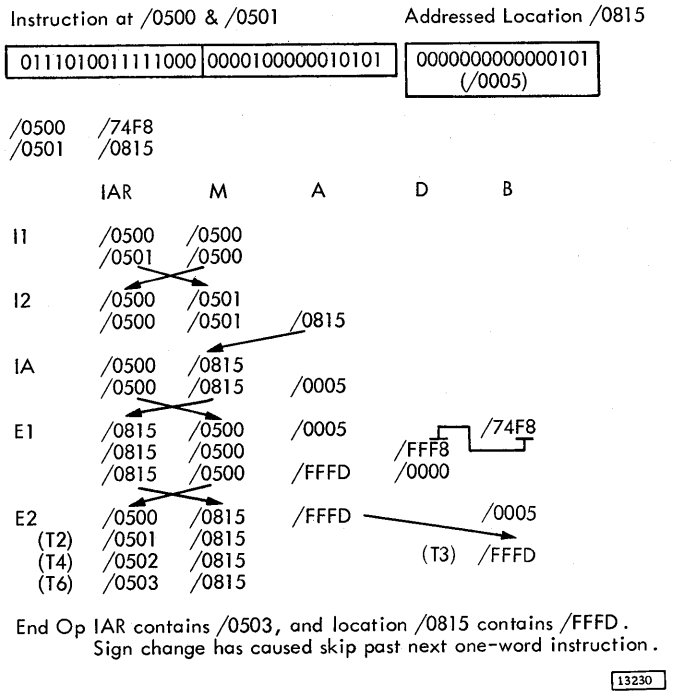



Figure 3-16. Add-to-Storage Operation

are described in the following paragraphs. Figure 3-16 shows an example of the operation.

I1 Cycle

At T6, the 'add to storage set interlock sample' turns on the 'add to storage interlock' FF. 'Add to storage interlock' turns on 'modifier 8' to force an IA cycle after the I2 cycle. During T7, both 'I to M SPD gate' and 'M to I SPD gate' are activated.

I2 Cycle

As I2 is entered, 'clock advance SP B' causes the exchange of the contents of the IAR and M-register. Before the exchange, the M-register contained the address of the MDX instruction word and IAR contained the address of the MDX address word. The exchange occurs very early in T0. As usual in an I2 cycle, the M-register addresses the second word of the MDX instruction.

'Add to storage op' blocks the usual 'increment IAR sample;' so IAR retains the address of the MDX instruction. After the increment has been blocked, the 'add to storage interlock' FF turns off at T1. Then T2 continues with the usual functions of setting the address word in the A-register.

IA Cycle

Functions of the forced IA cycle are nearly the same as any IA cycle. The contents of the core storage location specified by the address are set into the A-register in preparation for adding the displacement. If the contents are negative the 'B bit 0 latch' is turned on when the 1 is set in D-register 0 on its way to the A-register. T6 turns on 'add to storage interlock,' and T7 activates the gates for exchanging the contents of IAR and the M-register.

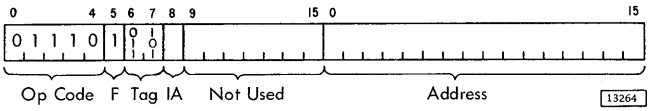
E1 Cycle

Exchanging the contents of IAR and the M-register saves the address word of the instruction for the E2 cycle. The address of the MDX instruction is set into the M-register to address core storage on this cycle. Incrementing of IAR is again blocked because 'add to storage interlock' is on. Then 'add to storage interlock' is turned off again. Reading storage on this cycle again sets the MDX instruction word into the B-register. At T2, B bits 8-15 (the displacement) set D-register positions 8-15. If B bit 8 is a 1, D-register 0-7 positions are set to 1 in order to maintain the negative sign. 'Add to storage op' and 'E1 cycle' block the 'gate B to D 0-7.' Add-subtract circuits then add the contents of the D- and A-registers in the usual manner. Once again, T6 turns on the 'add to storage interlock,' and T7 activates the gates for exchanging the contents of IAR and the M-register.

E2 Cycle

The exchange of the contents of IAR and the M-register at the start of the E2 cycle causes the address word of the MDX to address core storage again. T1 resets the 'add to storage interlock' after the exchange. T1 also decrements the CCC, because this is the last cycle of the operation. Remember that IAR still contains the address of the MDX instruction word. In order to obtain the next instruction, IAR must be incremented twice in this cycle. If the next one-word instruction is to be skipped, IAR must be incremented an extra time. Therefore, the A-register is tested for all 0's or a change of signs. Either condition causes incrementing of IAR contents by one at T2. 'Add to storage op' and 'E2' allow additional incrementing of IAR at T4 and T6. At T3, the sum is transferred from A to B, and at write time is set into the addressed core storage location. The previous contents of the A-register are returned from the U-register, and the operation ends at T7.

Long Format 'MDX', Tag ≠ 00




- When IA bit 8 = 0, the address word itself is added to the contents of the index register specified by the tag.
- When IA bit 8 = 1, the contents of the core storage location specified by the address are added to the contents of the selected index register.
- The sum is stored in the selected index register.
- A skip occurs if the sum is zero or if the sign in the index register is changed.
- The original contents of the A-register are returned at the end of the operation.
- 'Carry' FF and 'overflow' FF are not affected.
- Maintenance diagram AA663, Sheet 2.

The MDX instruction, being an index instruction, does not cause an IX cycle, even when the tag is not equal to 00. As usual, at the end of the I2 cycle, the A-register contains the address word. The rest of the operation depends on the state of the IA bit 8 in the instruction. Where bit 8 = 1, an IA cycle sets the contents of the location specified by the address into the A-register. Otherwise the address word itself is used as data.

The functions of the E1 cycle are to set the contents of the XR specified by the tag bits into the D-register and to add. When the 'E gate turn on' line becomes active, the 'IX inhibit SAR' FF turns on. M-register outputs are inhibited, and the tag register outputs activate the lines to address the specified XR. The bits from the XR, via the B-register, set the D-register at T2. Transferring a 'B bit 0 to D' turns on the 'B bit 0 latch' to save the negative sign of the index register. Adding action starts at T4 and may be complete by T7 or require extended T7 times.

The purpose of the E2 cycle is to store the sum in the selected XR. 'E1,' 'MDX,' and 'not tag 00' block the turn-off gate; so the 'IX inhibit SAR' FF does not turn off at the start of the E2 cycle. The tag register addresses core storage again as it did in E1. A condition of all 0's in the A-register or 'A bit 0' not the same as the 'B bit 0 latch' causes incrementing of IAR at T2. (IAR already contains the address of this instruction +2.) Transferring A to B and writing the B-register into the addressed core location stores the sum in the selected XR. T7 causes the return of the original contents of the A-register and an end operation condition. CCC was decremented to 0 at T1 of the E2 cycle, so the next T0 causes entry into an I1 cycle.

WAIT

- Any unused op code decodes as a 'wait' instruction.
- End operation occurs at end of I1 cycle because CCC is not set to 1.
- Contents of temporary accumulator (U-register) are returned to the A-register at 'T7 end op.'
- No EA can be computed during I1 cycle. None is needed.
- 'Wait' prevents T-clock advancing beyond T7 until start key is pressed or an interrupt occurs.
- X-clock can run and cycle stealing can occur while the CPU is in a 'wait' condition.
- Maintenance diagram AA641.

All the unused op codes (listed in maintenance diagram AA641) fit into one of these categories:

| | | | | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| x | x | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | x |
| 1 | 0 | 1 | 1 | x |

Any of the codes activate the 'wait op' line. The 'wait op' line then blocks 'set CCC1' at T5 and allows 'U to A SPD sample' at T7 of the I1 cycle. 'Wait op' also blocks the 'set arith control' line during I1; so no arithmetic action can occur. If neither 'single step mode' nor 'storage load/display' is active, 'wait op' gates the turn-off of

the 'run' FF at T7. Turning off the 'run' FF then prevents the 'T clock advance sample,' and the clock remains at T7.

The 'wait' instruction is not truly a branch or skip instruction. However, note that turning off the 'run' FF does not turn on the 'CPU stop latch.'

Therefore, an 'interrupt request' can turn back on the 'run' FF. The interrupt-forced BSI takes place, and the servicing subroutine is executed. Further operation is determined by the mainline program.

The X-clock advance is not dependent on the 'run' FF. Therefore, cycle steals can occur while the CPU is in a 'wait' condition.

## INPUT/OUTPUT INSTRUCTION

All input/output operations are initiated by a single CPU instruction, 'execute I/O.' 'Execute I/O' obtains a two-word input/output control command (IOCC) from core storage. The IOCC completely describes the operation to be performed.

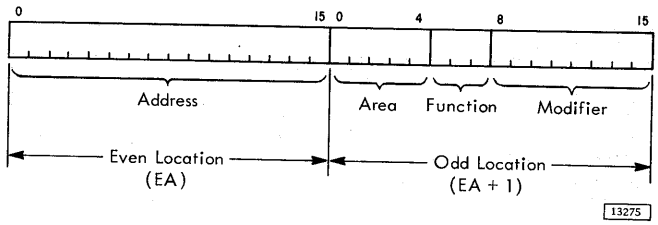
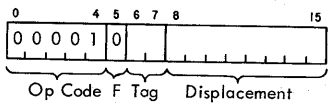
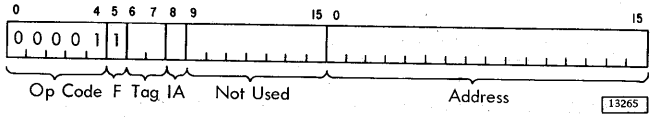



Figure 3-17. Input/Output Control Command (IOCC)

### EXECUTE I/O (XIO)




- Short format can be indexed. EA is computed during I1 cycle and IX cycle when called for.
- Long format can be indexed, indirectly addressed, or both. The address word provides the EA or is used in EA computation.
- XIO reads the IOCC from core storage locations EA and EA + 1.
- The control word of the IOCC (from EA + 1), set into the U-register during E1, defines the area, function, and modifiers.
- The area (device) and modified function determine the operation to be performed.
- The operation determines the use made of the IOCC address word.
- EA of the instruction must be even. Otherwise, the IOCC address word is never read.
- Device adapter circuits provide communication with the I/O devices.
- Maintenance diagram AA621.

The first objective of the XIO instruction is to obtain the control word of the IOCC (Figure 3-17) from EA + 1 and set it into the U-register. All further action depends on the decoding of area, function, and modifiers. Decoding the area determines which I/O device is affected and activates lines to the adapter circuits for that device. Function and modifier decodes also activate lines to the adapter circuits to define the action to be taken by the selected device.

Decoding an XIO op turns on the 'DPW odd address' FF when the E1 cycle is entered at T0. 'DPW odd address SAR 15' then simulates a 1 in M-register bit 15 to select and read the word at EA + 1. The word (control word of the IOCC) is transferred from storage via the B and D-registers into the A-register. 'T6' and 'XIO E1' activate the 'A to U SPD sample;' so the control word enters the U-register at T6 and starts to control the operation.

The U-register positions 0-4, through a decoding network, activate one of the 'area' lines to select an I/O device. Figure 3-18 shows the area code assignments. U-register positions 5-7 provide one of seven function codes (000 is not used) activating lines to all I/O devices which can perform that function. The usual operation resulting from each code and active line is:

001 -- XIO Write

Transfers a single word from core storage to an I/O device. The word is transferred from the location specified by the address word of the IOCC.

010 -- XIO Read

Transfers a single word from an I/O device to core storage via the B-register. The word is stored at the location specified by the address word of the IOCC.

| Binary | Decimal | |
|---|---|---|
| 00000 | 0 | Not used in the 1130 system |
| 00001 | 1 | Console Keyboard - Printer |
| 00010 | 2 | 1442 Card Read Punch |
| 00011 | 3 | 1134 Paper Tape Reader and 1055 Paper Tape Punch |
| 00100 | 4 | Disk Storage |
| 00101 | 5 | 1627 Plotter |
| 00110 | 6 | 1132 Printer |
| 00111 | 7 | Console Entry Switches |
| 01000 | 8 | 1231 Optical Mark Page Reader |
| 01001 | 9 | 2501 Card Reader |
| 01010 | 10 | Synchronous Communications Adapter |
| 01011 | 11 | Reserved for expansion within the basic 1131 I/O system. These codes, when unused by the 1130 system, can be assigned to any of the customers' I/O devices on SAC I or SAC II. |
| 01100 | 12 | |
| 01101 | 13 | |
| 01110 | 14 | |
| 01111 | 15 | |
| 10000 | 16 | |
| 10001 | 17 | 2310 B Disk Storage Drive 1 |
| 10010 | 18 | 2310 B Disk Storage Drive 2 |
| 10011 | 19 | 2310 B Disk Storage Drive 3 |
| 10100 | 20 | 2310 B Disk Storage Drive 4 |
| 10101 | 21 | 1403 Printer |
| 10110 | 22 | |
| 10111 | 23 | Reserved for system expansion . These codes when unused by the 1130 system can be assigned to any of the customer's I/O devices on SAC I or SAC II. |
| 11000 | 24 | |
| 11001 | 25 | |
| 11010 | 26 | |
| 11011 | 27 | |
| 11100 | 28 | |
| 11101 | 29 | |
| 11110 | 30 | |
| 11111 | 31 | |

Note: Any unused area code can be assigned to a customer's I/O device attached to the SAC I or SAC II.

16218A

Figure 3-18. XIO Area Code Assignments

011 -- XIO Sense ILSW

Sets the interrupt level status word (ILSW), for the interrupt level being serviced, into the A-register. No area code is used because the device is not known.

100 -- XIO Control

Causes various control actions in the I/O adapters or mechanical units. The action may be specified by modifier or address bits in the IOCC.

101 -- XIO Start Write (Initiate Write)

Enables the adapter circuits to request cycle steal cycles and accept write data from the CPU when the I/O device requires write data.

110 -- XIO Start Read (Initiate Read)

Enables the adapter circuits to request cycle steal cycles and transfer read data to the CPU when available.

111 -- XIO Sense Device

Sets the device status word (DSW) for the device specified by the area code into the A-register. If the device is assigned to a single interrupt level, a 1 in modifier bit 15 position of the IOCC resets off any status indicator. If the device is assigned to more than one level, bit 15 resets indicators on the highest level, bit 14 resets indicators on the next highest level, etc.

Any operation resulting from the function codes just described, that is peculiar to a particular I/O device, is noted in the description of the adapter for that device.

Once the IOCC control word is set into the U-register, CPU operation depends on the function code as shown in maintenance diagram AA621.

XIO Sense DSW or XIO Sense ILSW

As soon as the control word reaches the B-register during E1, 'gate sense inhibit storage use' becomes active. 'T7 SP' turns on the 'storage use' FF, which stays on until T7 of the following cycle. The 'storage use' FF being on blocks the 'storage use' line. Core storage is neither read nor written during an 'XIO sense' E2 cycle. 'Storage use' is blocked for either 'sense DSW' or 'sense ILSW.'

The adapter circuits place either the DSW or the ILSW on the I/O bus, depending on the function code. Activating 'I/O entry sample' fires 'I/O to B SP 0-7' and 'I/O to B SP 8-15.' The DSW or ILSW is set into the B-register and transferred via the D-register to the A-register at T1 to T3. The XIO sense operation is complete, and the next T0 causes entry into an I1 cycle.

XIO Control

The 'not end op T0 SP,' which causes entry into E2, also turns off 'DPW odd address.' Thus, the EA addresses core storage during the E2 cycle. By T3, the address word of the IOCC has been read and set into the B-, D-, and A-registers. Certain I/O adapter circuits use the contents of the B-register to define the control action. In a single disk storage operation, for example, the B-register at this time contains the number of cylinders the read/write heads are to be moved. However, not all devices use the address word of the IOCC for 'XIO control' operations.

The CCC is decremented to 0 at T1; so the operation is recognized as complete at T7. The next T0 causes entry into another I1 cycle.

XIO Initiate Read or XIO Initiate Write

Like all function codes other than XIO sense, E2 cycle reads the address word of the IOCC and sets it into the B-register. From there, the address word is available to I/O adapter circuits. The operation thereafter depends on the I/O device specified by the area code. CPU circuits usually gate the I/O device adapter circuits to set the IOCC address word into the CS address register of the selected device. Thus, the IOCC address word provides the address to be used for the first cycle steal cycle.

These function codes also gate the I/O adapter circuits so that the adapter can subsequently activate cycle steal requests when data transfer is required.

Decrementing the CCC to 0 allows end operation at T7, and the following T0 causes entry into an I1 cycle.

XIO Read or XIO Write

These XIO's usually appear in interrupt servicing subroutines to transfer a word of data. The E2 cycle for these function codes is like the others which are not 'XIO sense,' except that the CCC is not decremented to 0. The address word of the IOCC is set into the B- , D- , and A-registers. End operation cannot be activated at T7 because the CCC still contains a 1. 'Not end op T0 SP' turns on the 'E3' FF because 'XIO E2 read/write' is active.

Early in the E3 cycle, the contents of the A-register set the M-register. Consequently, the IOCC address word addresses core storage during the E3 cycle. For 'XIO write' operation, the word read from the addressed location into the B-register is available there for transfer to the I/O device. The adapter circuits usually set the output word into a buffer register. The CCC was decremented to 0 at T1, and the following T0 starts an I1 cycle.

Addressing core storage is the same for an 'XIO read' operation. In this case the word read from storage to the B-register is replaced at T3 by a word of data from the device. 'I/O to B SP' sets the B-register with the I/O bits from the device. Then, at write time of the storage cycle, the input word is stored. The CCC was decremented to 0 at T1, and the following T0 starts an I1 cycle.

## CONSOLE BIT SWITCHES

- These switches can provide data for input to core storage.
- For use with other switches -- for example, load IAR -- see Chapter 6.
- Program controlled operation only is described here.
- Area code assigned is area 7.
- IOCC functions of read and sense device are the only applicable functions.
- The instruction to read the bit switches may be given at any time. Usually, however, the instruction is in a level 4 interrupt servicing subroutine, so that data input results from pressing the interrupt request key.

### XIO Read, Area 7

An XIO instruction with an IOCC containing area code 7 and function code of read (Figure 3-19) causes the word set up in the bit switches to be stored in core storage. Each bit switch that is on causes storing of a 1, and each switch that is off stores a 0. During the E3 cycle caused by the XIO read, the 'console DSW/data bit' lines, corresponding to the bit switches that are on, are active. These, in turn, activate 'I/O bit' lines in the I/O bus. At T3, the B-register is set to match the bit switches. Then at write time, the word is written into core storage, and the operation is complete.

Programs may make data entry from the bit switches possible only after pressing the interrupt request key. Pressing the interrupt request key causes an interrupt level 4 request and a branch to the level 4 servicing subroutine. At this time, the setting of the console/keyboard switch must be determined. The switch being set to console activates

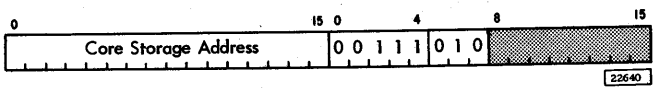



Figure 3-19. 'XIO Read' IOCC for Bit Switch Input

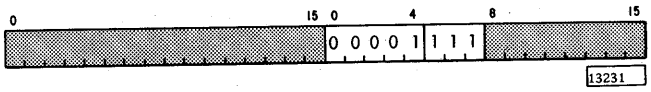



Figure 3-20. IOCC for Sensing Console Mode

a DSW bit that the program can use to branch to the proper portion of the subroutine. An 'XIO sense device' with an area code of 1 (Figure 3-20) is required to activate DSW bit 3, indicating that 'console mode' is active. When the program recognizes that the switch is set to console, a branch occurs to the portion of the level 4 subroutine which reads the bit switches.

The complete DSW for area 1, which includes both the console keyboard and the console printer, is shown in Figure 3-21.

### XIO Sense DSW, Area 7

'Level 5 interrupt request' may be activated either by pressing the program stop key or by operating in the interrupt run mode. In the latter case, the 'program trace trigger' FF is turned on for each mainline program instruction. Either of these conditions activates the 'console ILSW bit 0' when the ILSW is sensed while executing the level 5 subroutine. A means must be provided for determining which of these conditions caused the 'level 5 interrupt request.' An 'XIO sense device' with an area code of 7 (Figure 3-22) sets a DSW into the A-register. Figure 3-23 shows that this DSW provides the required status indicators.

## CONSOLE KEYBOARD

- Data transfer to the CPU from the console keyboard is under direct program control.

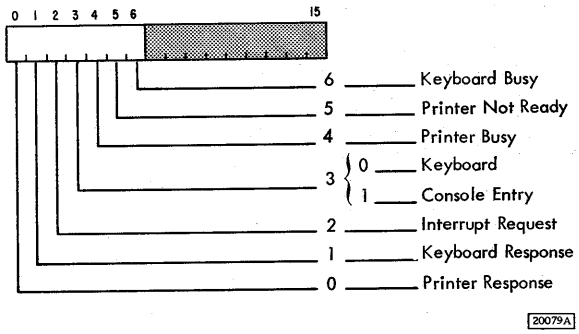



Figure 3-21. Device Status Word (Area 1)

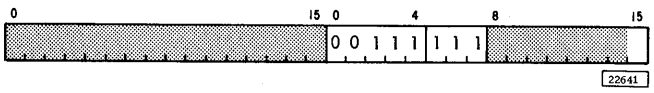



Figure 3-22. Sense DSW (Area 7)

- Area code assigned is 1.
- Keyboard response, resulting from keyboard mechanical action, causes a level 4 interrupt request.
- IOCC function codes applicable to the console keyboard are: sense interrupt (ILSW), sense device (DSW), control, and read.
- Maintenance diagrams XK501 and XK701.

Data input from the keyboard can be accomplished only when the keyboard is selected. As each input character is processed, the 'keyboard select' is turned off; so the keyboard must be selected again for each character. The character which is transferred to the CPU may be a data character or a control character (Figures 3-24 and 3-25).

Description of a typical keyboard input operation follows.

1. Set console/keyboard switch to keyboard.
2. Press interrupt request key, resulting in level 4 interrupt request. See Chapter 6.
3. When priority allows, branch to level 4 subroutine.
4. Program senses and analyzes level 4 ILSW. Bit 1 = 1 indicates console keyboard or printer (area 1).
5. Program senses and analyzes DSW. DSW bit 2 = 1 indicates manual interrupt and DSW bit 3 = 0 indicates keyboard is to be used for input. 'XIO sense reset 15' turns off manual interrupt.
6. Branch to instruction to select keyboard (KB select indicator lights). Then return to mainline program.
7. Press character key, resulting in level 4 interrupt request caused by keyboard response.
8. Program senses and analyzes ILSW and DSW to find interrupt was caused by keyboard response. Program resets keyboard response.
9. Branch to XIO instruction with area 1 and read function, storing one input character.
10. Program analyzes input character.
11. If character was data, program modifies address for storing the next character. Then operation skips to step 15.
12. If character was end-of-field control character (only bit 12 = 1), program skips to step 16.
13. If character was backspace control character (only bit 13 = 1), program modifies address so that next data character will replace last previously stored character. Then program skips to step 15.
14. If character was erase-field control character (only bit 14 = 1), program sets address back to location where first data character of this message was stored. Then proceeds to step 15.
15. Reselect the keyboard by instruction.
16. Complete the subroutine and return to the mainline program. Another character can then be entered if step 15 was taken. Note that pressing the end-of-field key caused the program to skip step 15. Then entry of another field requires that the interrupt request key be pressed again.

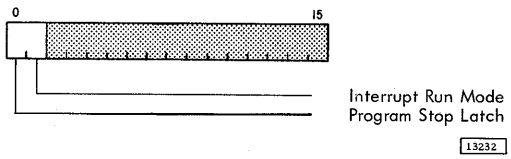



Figure 3-23. Device Status Word (Area 7)

The following paragraphs describe the operation of the function codes that are applicable to the console keyboard.

## Sense Interrupt

- Control word of the IOCC sets the U-register during E1.
- Address word of the IOCC is not used.
- ILSW sets the A-register via the I/O bus and the B- and D-registers during E2.

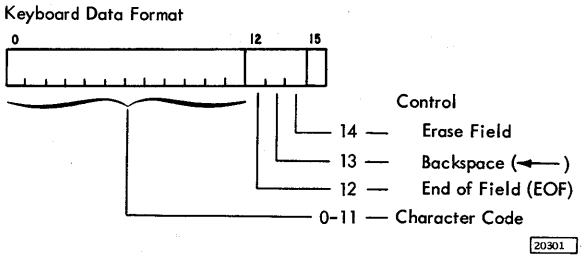



Figure 3-24. Keyboard Data Format

| Key | IBM Card Code | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| * | 11,8,4 | | 1 | | | | | 1 | | | | 1 | | | | | |
| / | 0,1 | | | 1 | 1 | | | | | | | | | | | | |
| Ø | 0 | | | 1 | | | | | | | | | | | | | |
| 1 | 1 | | | | 1 | | | | | | | | | | | | |
| 2 | 2 | | | | | 1 | | | | | | | | | | | |
| 3 | 3 | | | | | | 1 | | | | | | | | | | |
| 4 | 4 | | | | | | | 1 | | | | | | | | | |
| 5 | 5 | | | | | | | | 1 | | | | | | | | |
| 6 | 6 | | | | | | | | | 1 | | | | | | | |
| 7 | 7 | | | | | | | | | | 1 | | | | | | |
| 8 | 8 | | | | | | | | | | | 1 | | | | | |
| 9 | 9 | | | | | | | | | | | | 1 | | | | |
| $ | 11,8,3 | | 1 | | | | 1 | | | | | 1 | | | | | |
| . | 12,8,3 | 1 | | | | | 1 | | | | | 1 | | | | | |
| | 0,8,3 | | | 1 | | | 1 | | | | | 1 | | | | | |
| EOF | None | | | | | | | | | | | | | 1 | | | |
| Back Space | None | | | | | | | | | | | | | | 1 | | |
| ER FLD | None | | | | | | | | | | | | | | | 1 | |
| = | 6,8 | | | | | | | | | 1 | | 1 | | | | | |
| ' | 5,8 | | | | | | | | 1 | | | 1 | | | | | |
| ( | 12,5,8 | 1 | | | | | | 1 | | | | 1 | | | | | |
| ) | 12,5,8 | | 1 | | | | | | 1 | | | 1 | | | | | |
| + | 12,8,6 | 1 | | | | | | | | 1 | | 1 | | | | | |
| - | 11 | | 1 | | | | | | | | | | | | | | |
| A | 12,1 | 1 | | | 1 | | | | | | | | | | | | |
| B | 12,2 | 1 | | | | 1 | | | | | | | | | | | |
| C | 12,3 | 1 | | | | | 1 | | | | | | | | | | |
| D | 12,4 | 1 | | | | | | 1 | | | | | | | | | |
| E | 12,5 | 1 | | | | | | | 1 | | | | | | | | |
| F | 12,6 | 1 | | | | | | | | 1 | | | | | | | |
| G | 12,7 | 1 | | | | | | | | | 1 | | | | | | |
| H | 12,8 | 1 | | | | | | | | | | 1 | | | | | |
| I | 12,9 | 1 | | | | | | | | | | | 1 | | | | |
| J | 11,1 | | 1 | | 1 | | | | | | | | | | | | |
| K | 11,2 | | 1 | | | 1 | | | | | | | | | | | |
| L | 11,3 | | 1 | | | | 1 | | | | | | | | | | |
| M | 11,4 | | 1 | | | | | 1 | | | | | | | | | |
| N | 11,5 | | 1 | | | | | | 1 | | | | | | | | |
| O | 11,6 | | 1 | | | | | | | 1 | | | | | | | |
| P | 11,7 | | 1 | | | | | | | | 1 | | | | | | |
| Q | 11,8 | | 1 | | | | | | | | | 1 | | | | | |
| R | 11,9 | | 1 | | | | | | | | | | 1 | | | | |
| S | 0,2 | | | 1 | | 1 | | | | | | | | | | | |
| T | 0,3 | | | 1 | | | 1 | | | | | | | | | | |
| U | 0,4 | | | 1 | | | | 1 | | | | | | | | | |
| V | 0,5 | | | 1 | | | | | 1 | | | | | | | | |
| W | 0,6 | | | 1 | | | | | | 1 | | | | | | | |
| X | 0,7 | | | 1 | | | | | | | 1 | | | | | | |
| Y | 0,8 | | | 1 | | | | | | | | 1 | | | | | |
| Z | 0,9 | | | 1 | | | | | | | | | 1 | | | | |
| Space | Blank | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| ¢ | 12,8,2 | 1 | | | | 1 | | | | | | 1 | | | | | |
| < | 12,8,4 | 1 | | | | | | 1 | | | | 1 | | | | | |
| \| | 12,8,7 | 1 | | | | | | | | | 1 | 1 | | | | | |
| & | 12 | 1 | | | | | | | | | | | | | | | |
| ! | 11,8,2 | | 1 | | | 1 | | | | | | 1 | | | | | |
| ; | 11,8,6 | | 1 | | | | | | | 1 | | 1 | | | | | |
| ¬ | 11,8,7 | | 1 | | | | | | | | 1 | 1 | | | | | |
| % | 0,8,4 | | | 1 | | | | 1 | | | | 1 | | | | | |
| _ | 0,8,5 | | | 1 | | | | | 1 | | | 1 | | | | | |
| > | 0,8,6 | | | 1 | | | | | | 1 | | 1 | | | | | |
| ? | 0,8,7 | | | 1 | | | | | | | 1 | 1 | | | | | |
| : | 8,2 | | | | | 1 | | | | | | 1 | | | | | |
| # | 8,3 | | | | | | 1 | | | | | 1 | | | | | |
| @ | 8,4 | | | | | | | 1 | | | | 1 | | | | | |
| " | 8,7 | | | | | | | | | | 1 | 1 | | | | | |

24062B

Figure 3-25. Keyboard Character Code

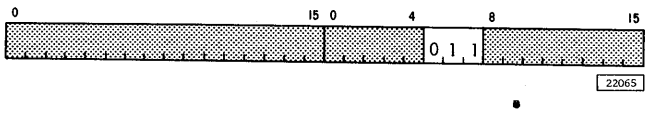



Figure 3-26. IOCC for Sense Interrupt

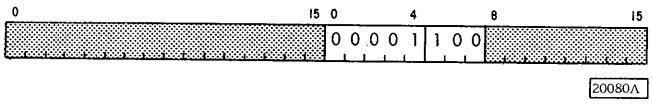



Figure 3-28. Control Function IOCC

The 'XIO sense ILSW' transfers to the A-register the ILSW for the level that is being serviced when the instruction is given. The IOCC is shown in Figure 3-26.

Figure 3-27 shows the level 4 ILSW set into the A-register by sense interrupt when level 4 is active.

The console keyboard (keyboard response), console printer (typewriter response), or interrupt request key (manual interrupt) set a 1 in bit position 1.

Sense Device

The 'XIO sense device' for area 1 is described in the paragraphs concerning console bit switches because it is used in the program for bit switch input to check for console mode. Figure 3-20 shows the IOCC, and Figure 3-21 shows the DSW for area 1. IOCC bit 15 = 1 resets the response which caused the interrupt.

Control

- Only control function is keyboard selection.
- Address word is not used for keyboard operation.

With an area code 1, when the 'XIO control' line is deactivated, the 'keyboard select' FF turns on. The FF lights the KB select indicator, gates the firing of keyboard singleshot 1, and gates the keyboard busy bit in the DSW. 'XIO control' is active during the time the 'XIO time gate' FF is on in an E2 cycle, or from T1 through T6. Thus, the 'keyboard select' FF is turned on at T7 time.

The IOCC for this operation is shown in Figure 3-28.

Read

- 'XIO read' is performed in the level 4 interrupt servicing subroutine.
- Data is provided by the keyboard latch and bail contacts.
- 'XIO read' sets the data on the I/O bus.
- I/O-to-B sample pulses set the word in the B-register during an E3 cycle. At write time in the cycle, the data word is written into storage.

Pressing a key activates 'keyboard data' lines according to the latch and bail contacts that are operated by the key. Any 'keyboard data' line activates either 'keyboard data A' or 'keyboard data B.' Either of these lines fires keyboard SS-1 if 'keyboard select' FF is on, and activates 'keyboard response gate.' Singleshot 1 timing out turns on the 'keyboard response' FF. 'Keyboard response' results in a level 4 interrupt request and a 1 in DSW bit position 1. When the program has determined that a keyboard response caused the interrupt, and the program has turned off the 'keyboard response' FF, the 'XIO read' instruction is given. The IOCC is shown in Figure 3-29.

As shown in maintenance diagram AA621, 'XIO read' requires an E3 cycle. The address word of the IOCC is in the A-register as the E3 cycle starts.

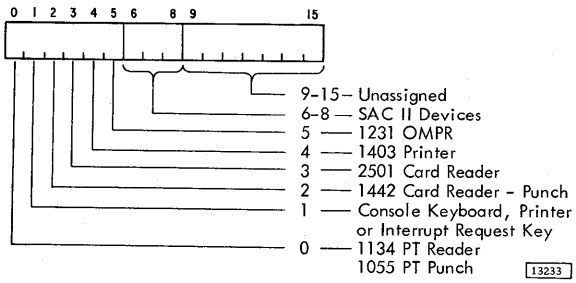



Figure 3-27. ILSW for Interrupt Level 4

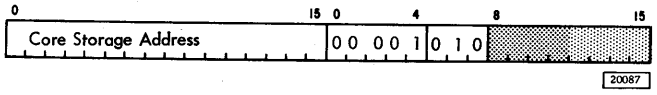



Figure 3-29. IOCC for Read (Area 1)

Transferring it to the M-register allows the address word of the IOCC to address storage for the E3 cycle. 'XIO read' and 'area 1' turn off the 'keyboard select' FF, and doing so fires keyboard SS-2. While the singleshot is timing out, it gates '+48 volts' to energize the keyboard restore magnet. Before the restore action, the keyboard latch and bail contacts are activating the 'keyboard data' lines. 'XIO read' activates 'entry gate' and, at T3 of the E3 cycle, 'I/O entry sample' fires the I/O-to-B sample pulses. The keyboard data bits on the I/O bus set the B-register. When the character is written into core storage, 'XIO read' is complete. The remainder of the level 4 subroutine determines further action.

## CONSOLE PRINTER

- Data transfer from the CPU to the console printer is under direct program control.
- Area code assigned is 1.
- Printer (typewriter) response, resulting from mechanical action, causes a level 4 interrupt request.
- IOCC function codes, applicable to the console printer, are: sense interrupt (ILSW), sense device (DSW), and write.
- Maintenance diagrams XW401, XW501, and XW701.

Any console printer operation is initiated by transferring an eight-bit character to the console printer adapter from CPU core storage. The character that is transferred may be either a data character or a control character. All encoding of output data must be done by the CPU program, and each character occupies one-half a core storage location (Figure 3-30). The complete data character coding is shown (Figure 3-31), and control character codes are shown (Figure 3-32).

Description of a typical console printer output operation follows.

1. Program senses and analyzes DSW to ensure printer is ready and not busy before giving 'XIO write.'

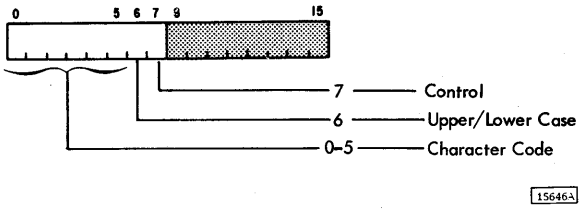



Figure 3-30. Console Printer Character Format

| Character Code Bits | | | | | | U/L Case | | Ctrl |
|---|---|---|---|---|---|---|---|---|
| B0 $T_2$ | B1 $T_1$ | B2 $R_1$ | B3 $R_{2A}$ | B4 $R_2$ | B5 $R_5$ | B6=0 LC | B6=1 UC | B7 |
| 0 | 0 | 1 | 1 | 1 | 1 | A | A | 0 |
| 0 | 0 | 0 | 1 | 1 | 0 | B | B | 0 |
| 0 | 0 | 0 | 1 | 1 | 1 | C | C | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 | D | D | 0 |
| 0 | 0 | 1 | 1 | 0 | 1 | E | E | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | F | F | 0 |
| 0 | 0 | 0 | 1 | 0 | 1 | G | G | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 | H | H | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | I | I | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | J | J | 0 |
| 0 | 1 | 0 | 1 | 1 | 0 | K | K | 0 |
| 0 | 1 | 0 | 1 | 1 | 1 | L | L | 0 |
| 0 | 1 | 1 | 1 | 0 | 0 | M | M | 0 |
| 0 | 1 | 1 | 1 | 0 | 1 | N | N | 0 |
| 0 | 1 | 0 | 1 | 0 | 0 | O | O | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 | P | P | 0 |
| 0 | 1 | 1 | 0 | 0 | 1 | Q | Q | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | R | R | 0 |
| 1 | 0 | 0 | 1 | 1 | 0 | S | S | 0 |
| 1 | 0 | 0 | 1 | 1 | 1 | T | T | 0 |
| 1 | 0 | 1 | 1 | 0 | 0 | U | U | 0 |
| 1 | 0 | 1 | 1 | 0 | 1 | V | V | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 | W | W | 0 |
| 1 | 0 | 0 | 1 | 0 | 1 | X | X | 0 |
| 1 | 0 | 1 | 0 | 0 | 1 | Y | Y | 0 |
| 1 | 0 | 1 | 0 | 0 | 0 | Z | Z | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | ( | 0 |
| 1 | 1 | 0 | 1 | 1 | 0 | 2 | + | 0 |
| 1 | 1 | 0 | 1 | 1 | 1 | 3 | < | 0 |
| 1 | 1 | 1 | 1 | 0 | 0 | 4 | ¬ | 0 |
| 1 | 1 | 1 | 1 | 0 | 1 | 5 | ) | 0 |
| 1 | 1 | 0 | 1 | 0 | 0 | 6 | ; | 0 |
| 1 | 1 | 0 | 1 | 0 | 1 | 7 | * | 0 |
| 1 | 1 | 1 | 0 | 0 | 1 | 8 | ' | 0 |
| 1 | 1 | 1 | 0 | 0 | 0 | 9 | " | 0 |
| 1 | 1 | 0 | 0 | 0 | 1 | 0 | \| | 0 |
| 1 | 1 | 0 | 0 | 0 | 0 | # | = | 0 |
| 1 | 0 | 1 | 1 | 1 | 1 | / | ¬ | 0 |
| 1 | 0 | 0 | 0 | 0 | 1 | - | ? | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | , | : | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 | & | > | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | $ | . | 0 |
| 0 | 0 | 0 | 0 | 0 | 1 | @ | % | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | . | ¢ | 0 |

22265A

Figure 3-31. Data Character Coding, Console Printer

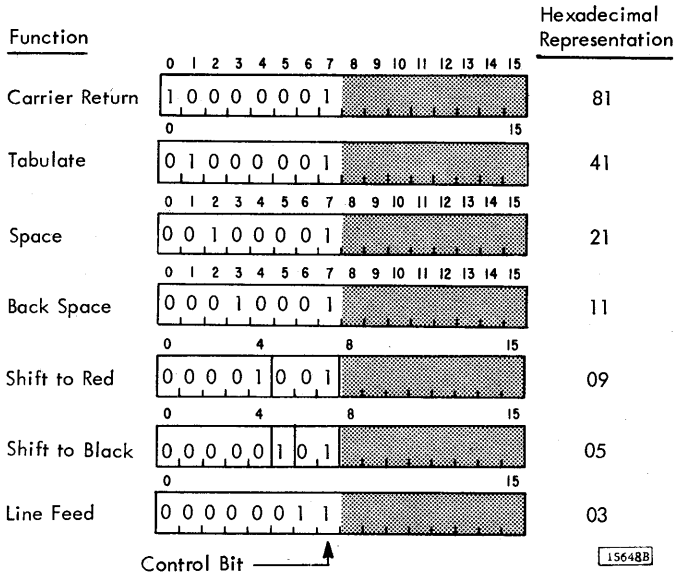



Figure 3-32. Control Character Coding, Console Printer

2. 'XIO write' transfers character to buffer in console printer adapter. Buffer bit configuration determines mechanical action printer is to take.
3. Program modifies IOCC address word to location that is to provide next character. Then execution of mainline program continues.
4. At end of typewriter cycle, typewriter (printer) response causes level 4 interrupt request.
5. Program senses and analyzes ILSW and DSW to find interrupt was caused by printer response. Program resets printer response.
6. Program determines whether or not there is more output data. If there is not, skips to step 9.
7. Program gives another 'XIO write' to start another typewriter cycle.
8. Program modifies IOCC address word to obtain next output character.
9. Program completes the subroutine and returns to the mainline program.

## Sense Interrupt

The 'XIO sense ILSW' operation is the same as that for the console keyboard. The IOCC is shown in Figure 3-26, and the ILSW which is set into the A-register by the operation is shown in Figure 3-27. 'Printer (typewriter) response' is one of the means of activating 'ILSW 4 bit 1.'

## Sense Device

The 'XIO sense device' operation is the same as that described for the other console devices (area 1). Figure 3-20 shows the IOCC, and Figure 3-21 shows the resulting DSW. Three positions in the DSW are set to 1 by conditions in the console printer and its adapter.

Bit 0: A 1 in this position of the DSW indicates that the 'response' FF in the printer adapter is on. 'Typewriter (console printer) response' also causes a 'keyboard/typewriter interrupt level 4.'

Bit 4: A 1 in this position of the DSW indicates that the typewriter (printer) is busy. The typewriter becomes busy when either the 'typewriter cycle' FF or the 'end of line' FF turns on. Then the typewriter remains busy as long as either of those FF's or the 'interlock latch' stays on.

Bit 5: A 1 in this position of the DSW indicates that the typewriter is busy or that the end of forms contact is open. The contact is open with no forms in the typewriter.

## Write

The IOCC for an 'XIO write' using the console printer is shown in Figure 3-33. During the E3 cycle resulting from the CPU 'XIO' instruction, a word is read from the core storage location specified in the IOCC. This word may be either a data character to be printed or a control character to cause a printer control operation (Figure 3-32). Figure 3-34 shows the operation from the setting of the buffer at T6 of the E3 cycle.

In transferring the B-register to the console printer buffer, the absence of a B bit 7 gates the turn-on of the 'check bit' FF. 'Check bit' is on for a character to be printed, and is off to cause some other mechanical action of the printer. Bit positions 0-6 of the control character determine which control magnet is energized (Figure 3-32).

When a data character is to be printed, the status of buffer 'bit 6' FF determines whether the character is upper- or lowercase. Comparing the

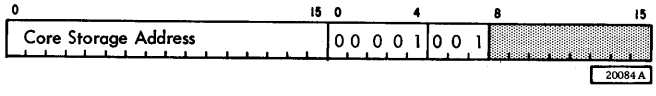



Figure 3-33. IOCC for Console Printer (Write)

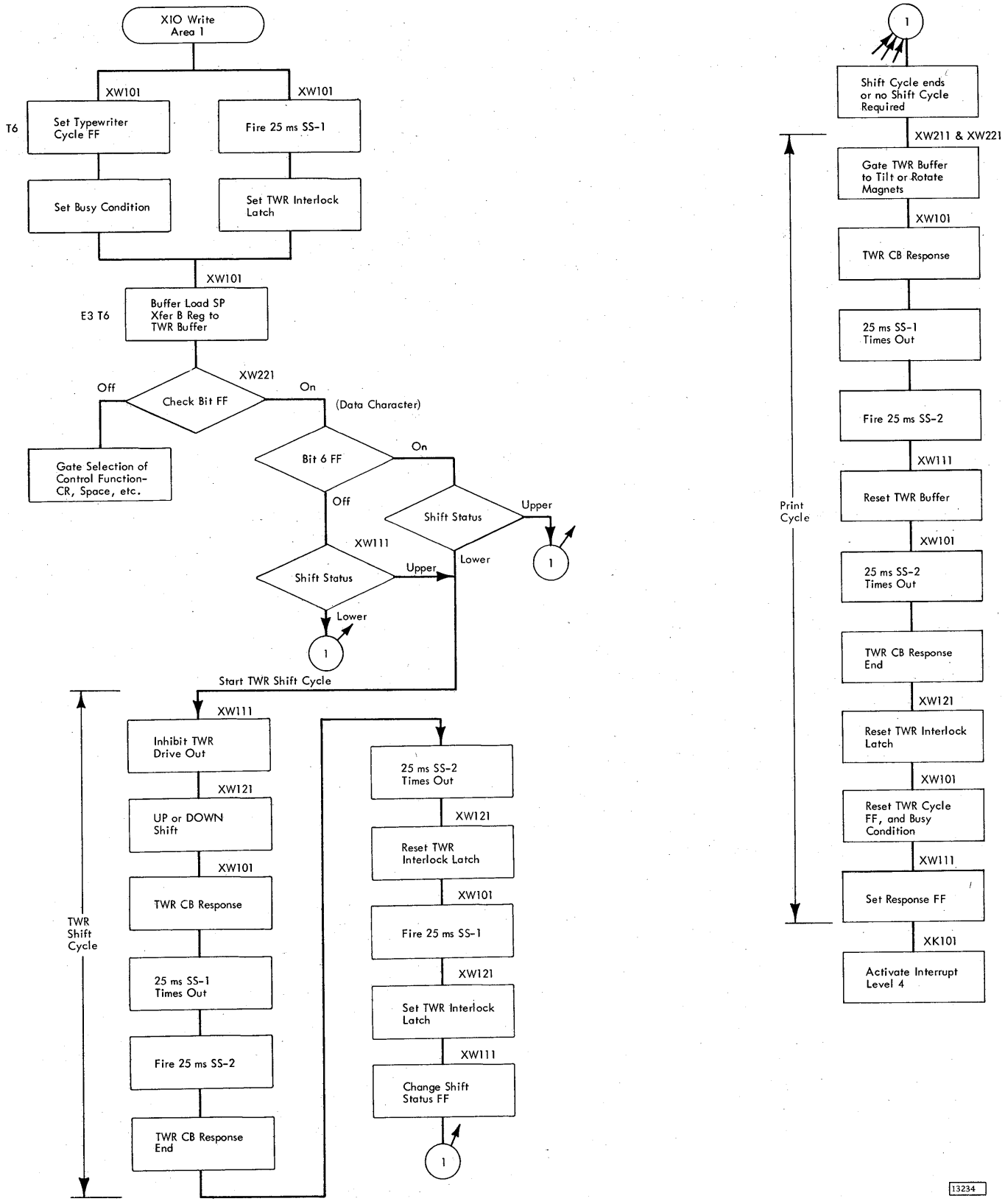



**Figure 3-34. Print Cycles**

shift called for and the present shift status determines whether a typewriter shift cycle is required. During the typewriter shift cycle, activating 'inhibit typewriter drive out' blocks the buffer output. Also the 'typewriter cycle' FF is prevented from turning off, even though the 'typewriter interlock latch is turned off and back on again. The required up- or down-shift is completed, and the 'shift status' FF is changed to reflect the new shift status.

When the shift cycle ends or no shift cycle is required, buffer outputs are gated to the tilt and rotate select magnets. The mechanical action of printing the data character starts. 'Typewriter CB response,' 'typewriter cycle' and typewriter SS-1 not active (timed out) activate the 'typewriter drive interlock' line. This line, with 'typewriter cycle' and not 'typewriter shift cycle' causes a buffer reset. Then, when both singleshots time out and 'CB response' ends, deactivating 'typewriter interlock' turns off the 'interlock latch.' The 'typewriter cycle' FF turns off, and 'typewriter busy' is no longer active. Deactivating busy turns on the 'response' FF, which is one means of causing an interrupt level 4. The 'XIO write' operation is finished, and further console printer operation depends on the program.

Only the single disk storage adapter is described in this manual because this feature is basic to certain models of the IBM 1130 Computing System. Other feature adapters are described in the IBM 1130 Computing System Features FETO.

## SINGLE DISK STORAGE

### DESCRIPTION

- Single disk storage drive 0 and its adapter are basic to the IBM 1130 Computing System, Models 1 and 2. Both the drive and the adapter are within the 1131 unit.
- The 2315 disk cartridge is the storage medium, providing on-line capacity of 512,000 sixteen-bit data words.
- Interchangeability of cartridges, in effect, provides unlimited storage capacity.
- Two read/write heads on one access mechanism provide for the use of both sides of the disk.
- Disk information can be stored in 203 tracks by moving the access mechanism to any of 203 positions.
- A track on the upper surface and a track on the lower surface comprise a cylinder.
- Each surface has four sectors; so a cylinder contains eight sectors numbered 0 through 7.
- Normal programming practice calls for writing an identification word preceding the 320 data words in each sector.
- Each disk storage word contains four check bits in addition to the 16-bit data word.
- The disk speed is 1,500 rpm or one revolution in 40 ms.
- One word is written or read in 27.8 microseconds, and one bit time is 1.39 microseconds.
- Additional information concerning the single disk storage drive is included in the IBM Single Disk Storage (Incremental Access) FETO, Form Y26-3669.
- Unit data and control maintenance diagram: XF401.

The single disk storage device provides the IBM 1130 Computing System with low-cost random or sequential access data storage. On-line data capacity is 512,000 words. Off-line capacity is virtually unlimited because the interchangeable disk cartridge is easily removed and replaced with another. Thus, the large storage capacity, comparable to that of magnetic tape, coupled with the advantage of random access, affords the IBM 1130 Computing System great flexibility in handling engineering, scientific, industrial and commercial programs.

#### Disk Storage Unit

Disk storage drive 0 for the 1130 system is contained in the CPU cabinet and is connected to the CPU circuits by the disk storage adapter. It has two components: the disk drive assembly and the access mechanism.

#### Disk Cartridge Assembly

The assembly is a single disk, completely enclosed in a protective housing or cartridge. The recording medium is an oxide-coated disk that provides two surfaces for the magnetic recording of data. The disk rotates at the rate of 1,500 revolutions per minute or one revolution in 40 ms.

#### Access Mechanism

The disk storage access mechanism has two arms, one above the disk and one below the disk. Each arm has a magnetic read/write head to read or write on the corresponding disk surface. The entire head assembly moves horizontally forward and backward, so that the heads have access to the entire recording area.

The access mechanism is positioned automatically at the home position when the disk cartridge is inserted.

When the disk drive switch is turned off, the access mechanism retracts the heads from the cartridge.

## Disk Organization and Capacity

The access mechanism is moved as a result of program instructions and can be placed in any one of 203 positions. At each position, either head can read or write in a circular pattern on a surface of the revolving disk. These circular patterns of data are called tracks. The track on the upper surface of the disk and the corresponding track on the lower surface, which can be read or written while the access mechanism is in the same position, comprise a cylinder. Figure 4-1 shows the innermost and outermost cylinders of two tracks each. To complete the picture, the 201 intermediate cylinders, or pairs of tracks, should be visualized. They were omitted for the sake of clarity in the diagram. Usually programs and programming systems use only 200 tracks, setting aside 3 for emergency use.

Each track is divided into four equal segments called sectors. Sectors are numbered from 0 through 7 as shown in Figure 4-2. Sectors 0-3 are on the upper surface and sectors 4-7, the lower. A sector can contain 321 words and is the largest segment of data that can be read or written with a single instruction. Usually programs and programming systems use the first word of the sector as a sector identification word.

In addition to 320 data words, each sector has an additional word that can be used as needed by a particular program. In the programs and programming systems provided by IBM (e.g., the monitor system and its programs) the first word of a 321-word sector is used for sector number. Therefore, the first word of the sector must be used with caution by the programmer if the assembler program or other components of the monitor system are used. Otherwise, this word may be used for a sector address, data, or other purposes.

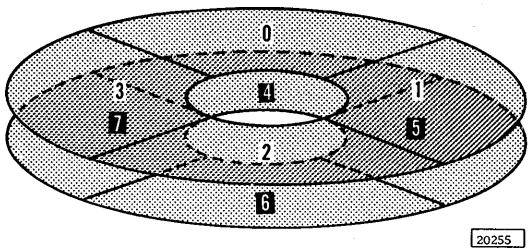



Figure 4-2. Sector Numbers of Upper and Lower Disk Surfaces

The format of a sector is shown in Figure 4-3. Following a sector pulse, 0's are written for approximately 250 microseconds. This 0-bit field is followed by the synchronization (sync) word, consisting of fifteen 0's, four 1's, and a 0, in that order. The 0-bit field and the sync word are written by the adapter controls and are used by the adapter circuits to synchronize with the data when reading back. The sync word is not included in the 321 words that can be written in a sector.

A disk storage word comprises 16 data bits and 4 check bits. Figure 4-4 shows the layout of a word. Clock bits are recorded every 1.39 microseconds and if a 1 is to be written, it is placed between the two clock bits. The check bits in the last 4 bit positions of each 20-bit word depend on the number of data 1's written. A two-position 'file check counter' in the adapter is used to write the check bits. The

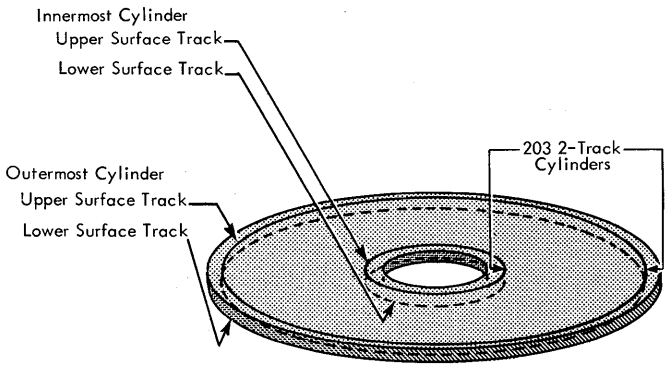



Figure 4-1. Disk Cylinder Concept

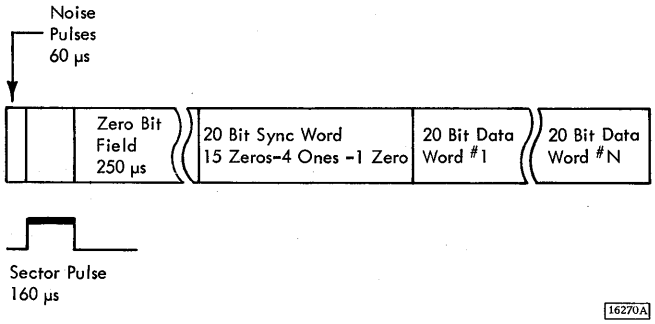



Figure 4-3. Sector Format

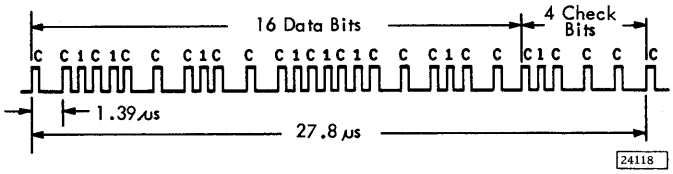



Figure 4-4. Disk Storage Data Word

| No. of Per | Disks | Cylinders | Tracks | Sectors | Words |
|---|---|---|---|---|---|
| Cylinders | 200 | | | | |
| Tracks | 400 | 2 | | | |
| Sectors | 1,600 | 8 | 4 | | |
| Data Words | 512,000 | 2,560 | 1,280 | 320 | |
| Bits | | | | | |
| Data | 8,192,000 | 40,960 | 20,480 | 5,120 | 16 |
| Check | 2,048,000 | 10,240 | 5,120 | 1,280 | 4 |
| Total | 10,240,000 | 51,200 | 25,600 | 6,400 | 20 |

20261

Figure 4-6. Disk Storage Organization

relationship of the number of data bits to the file check counter and the check bits written is shown in Figure 4-5.

Figure 4-6 shows the organization of disk storage data. Capacities are based on the 320-word sector and 200-track disk area.

## Disk Storage Timing

Timing considerations of disk storage operations involve three elements: access time, read/write time, and the CPU time used.

Access: The access mechanism moves in increments of one or two cylinders at the rate of 15 ms per increment. All movements of the mechanism are in increments of two, except the first movement if the number of cylinders to be moved is odd. During the stabilization period that follows the last incremental movement, a read or write instruction can be given, but execution does not start until the end of the stabilization period. The stabilization time is approximately 22.5 ms.

| Number of 1's in Data Word Positions 0-15 | 0<br>4<br>8<br>12 | 1<br>5<br>9<br>13 | 2<br>6<br>10<br>14 | 3<br>7<br>11<br>15 |
|---|---|---|---|---|
| File Check Counter | 00 | 01 | 10 | 11 |
| Check Bits | | | | |
| 16 | 0 | 1 | 1 | 1 |
| 17 | 0 | 1 | 1 | 0 |
| 18 | 0 | 1 | 0 | 0 |
| 19 | 0 | 0 | 0 | 0 |

22218A

Figure 4-5. Check Bit Chart

Thus, access time (ms) = 7.5 (N) +22.5 when the number of tracks (N) is even. When N is odd, access time = 7.5 (N) +30.0 ms.

Read/Write: Disk storage words are written or read at an approximate rate of 36,000 per second, resulting in a word time of 27.8 microseconds and a bit time of 1.39 microseconds. Average rotational delay time is 20 ms, based on 1,500 rpm or 40 ms per revolution. Thus, a sector can be read or written in an average of 30 ms.

A full cylinder of eight 321-word sectors can be read or written in 100 ms because the rotational delay is required for only the first sector. There are no timing considerations for switching heads because there is an interval of 450 µs between sectors; the interval is increased by 27.8 µs for each word less than 321 read or written.

CPU Time: An interrupt in a disk storage operation occurs only at the end of the seek, read, or write operation. This means that once the instruction is initiated, disk storage operation is virtually independent of the CPU. As data is being read or written, a cycle steal occurs every 27.8 microseconds for the transfer of the next word. Approximately 9.25 ms are required to transfer the data for a full cylinder, when the CPU cycle time is 3.6 µs. The same data transfer requires approximately 5.65 ms when the cycle time is 2.2 µs.

## Disk Storage Data Checking

Data is checked on each transfer between disk storage and core storage. When writing on disk storage, the number of 1's in each word is effectively divided by 4, using a two-position file check counter. The counter causes enough 1's to be written following the 16 data bit positions to make the total number of 1's

in the word evenly divisible by 4 (modulo 4). By the end of the word, the counter contains 0. Figure 4-5 shows the number and location of 1's written under control of the file check counter.

The same file check counter checks each word read from the disk. Reading a number of 1's not evenly divisible by 4 leaves the counter containing a value other than 0 at the end of the word. This is a data error and results in a 1 in the data error position of the disk storage DSW.

## ADAPTER FUNCTIONAL UNITS

### Word Count Register

- Also called "word counter," the word count register is used as a nine-position binary counter.
- Register is loaded, in a 1's complement form, from the B-register when starting a seek, read, or write operation.
- In read or write operations, the counter is incremented +1 for each word transferred.
- In a seek operation, the counter is incremented +1 when the access mechanism moves one track and is incremented +2 when the access mechanism moves two tracks.
- All counter FF's on indicates a full word count (end of operation).
- Maintenance diagrams XF511 and XF521.

The word count register is set to the number of words to be read or written, or it is set to the number of tracks the access mechanism is to travel (Figure 4-7). The number is set in 1's complement binary form. Then incrementing +1 for each word read or written, or for each track the access mechanism travels, results in a full word count when the operation is complete. When the access mechanism is moving in 2-track increments, the word count register is also being incremented +2.

### Loading Word Count Register

During a read or write operation, the 'bit counter gate' is active, and X6 time of a level 0 CS cycle fires the 'load word counter SP.' During a seek operation, T4 of the E2 cycle resulting from 'XIO control' and area 4 fires the 'load word counter SP.'

| B Register | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 |
|---|---|---|---|---|---|---|---|---|---|
| Word Count Three | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Word Counter Positions | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 |
| Set Complement | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Move One Cylinder | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Move Two Cylinders (Word Counter Full) | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Operation Complete | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

22225 A

Figure 4-7. Word Counter Operation in Accessing

The 1's complement of the B-register 7-15 positions is set into the word count register. The complement is loaded because each word count register FF turns on only if the corresponding B-register FF is off.

### Incrementing Word Count Register

Read or Write: 'Bit counter gate' is active; so each 'increment word count' pulse changes the state of the 'bit 15' FF. The 'bit 14' FF turns on as a result of 'bit 15' turning off. When both 'bit 15' and 'bit 14' are on, the 'increment word count' pulse turns both off. The result is that the counter is incremented +1 for each word read or written.

Accessing: When the word count is odd, 'bit 15' FF is off. 'Bit counter gate' is not active during accessing; so 'increment word count' has no effect on the 'bit 15' FF. The end of the first 'access drive' turns on 'bit 15' which remains on throughout the accessing operation. Thus, the counter is incremented +1 by the first (single-track) access motion. If the word count is even, this increment +1 does not occur.

While 'bit 15' is on, each 'increment word count' changes the state of the 'bit 14' FF, resulting in an increment +2 to the counter. Turning on the 'access drive latch' activates ' increment word count'. Thus, two-track access motion steps the counter by 2.

### File Data Register

- Register is set with a data word from the B-register during each CS cycle after the first in a write operation.
- Serial data to the storage drive unit is provided by shifting bits in the register when writing.

- Bit 15 position FF on determines that a binary 1 is written.
- In a read operation, bit 0 position is turned on when a 1 is read from the disk and turned off when no 1 is read.
- Serial data from the storage drive unit is shifted into the register when reading.
- A 16-bit data word is transferred to the B-register via the I/O bus during each CS cycle after the first in a read operation.
- Maintenance diagram: XF511.

The file data register serializes data in a write operation and deserializes data in a read operation.

Write

The sync word (fifteen 0's, four 1's, and another 0) is written at the end of the field of 0's started at sector pulse time. In order to write the sync word, a 1 is set into the file data register bit 0 position. 'Load sync word,' when active, turns on the 'bit 0' FF. The 1 shifts through the file data register. When the 1 shifts into bit 15 position, it writes a 1 on the disk and steps the check counter to 1. This completes the function of the file data register in writing the sync word.

In order to write a data word, a 'register load SP' (pulse) sets the file data register to the same bit configuration as the B-register. X4 time of a CS cycle causes the parallel data transfer. After the bit in bit 15 position is written, a 'shift SP' shifts all bits in the register one position toward the bit 15 end. The process is repeated until all bits of the data word have been written.

Read

In a read operation, the state of the 'input trigger' FF determines whether the 'bit 0' FF turns on or off at shift time. The state of the 'input trigger' is controlled by 'read data' from the disk. Thus, data from the disk enters the bit 0 end of the file data register and is shifted toward the bit 15 end. 'CS level 0' activates 'level 0 data gate,' gating the file data register word to the I/O bus. The word is transferred in parallel to the B-register for writing into storage.

File Core Address Register

- Address register is set from positions 1-15 of the address word of the IOCC.
- Setting of register occurs during E2 cycle of XIO instruction when the function is initiate read or initiate write.
- Register addresses CPU core storage during CS cycles.
- Value in the register is incremented +1 during each CS cycle at X7 time.
- Maintenance diagram: XF511.

During E2 cycle of an XIO instruction, the address word of the IOCC is in the B-register of the CPU. T6 activates the 'load file address' line, because 'file load gate' is active. Either function code, initiate read or initiate write, can activate 'file load gate.'

After a word has been transferred from core storage to the single disk storage adapter, the file core address register is incremented. The end of the X6 pulse during CS level 0 changes the state of the 'file address bit 15' FF. The other FF's in the register are binarily connected, so that each pulse to the bit 15 position increments the core storage address used during CS cycles.

File Check Counter

- This two-position binary counter ensures that the number of 1's written in a disk storage word is evenly divisible by 4.
- The counter indicates an error if the number of 1's read from a disk storage word is not divisible by 4.
- Contents are incremented for each 1 that is written or read.
- Maintenance diagram: XF511.

The file check counter is stepped (the units position changes state), when 'check counter gate' is active and a 'counter SP' (pulse) occurs. 'Check counter gate' is active when a 1 is written or read. 'Counter SP' is activated at each read or write clock time.

'Counter SP' is blocked when the 'read/write condition' FF is off.

In a write operation, during bit counter E-time, the counter causes 1's to be written until the counter reaches 0. The counter is stepped each time a 1 is written.

In a read operation, the counter is stepped each time a 1 is read. A data error is indicated if the counter is not at 0 when reading of a word stops (end of bit counter E-time).

Bit Counter

- The bit counter consists of five FF's, 'A' through 'E.'
- Counter steps each bit time.
- Sixteen data positions are read or written before 'E' turns on.
- Check bits are written while 'E' is on.
- Maintenance diagram: XF511.

The bit counter is stepped by each 'counter SP' pulse when the 'bit counter gate' is active during read and write operations (Figure 4-8). The counter operates in a standard binary manner until 'A' through 'D' turn off and 'E' turns on. The next four sample pulses:

1. Turn on 'A.'
2. Turn off 'A' and turn on 'B.'
3. Turn on 'A,' with 'B' remaining on.
4. Turn off 'A,' 'B,' and 'E,' so that 'A' through 'E' are all off.

Turning off 'B' when 'E' is on does not change the state of (complement) 'C'.

| Bit Positions | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit Ctr. A | | x | | x | | x | | x | | x | | x | | x | | x | | x | | x |
| Bit Ctr. B | | | x | x | | | x | x | | | x | x | | | x | x | | | x | x |
| Bit Ctr. C | | | | | x | x | x | x | | | | | x | x | x | x | | | | |
| Bit Ctr. D | | | | | | | | | x | x | x | x | x | x | x | x | | | | |
| Bit Ctr. E | | | | | | | | | | | | | | | | | x | x | x | x |

22227

Figure 4-8. Bit Counter Sequence

While the bit counter steps from an "all off" condition to the next "all off" condition, one word is read or written. The 16-position data word is read or written before the 'E' FF turns on, and check bits while 'E' is on.

Sector Register

- The three register flip-flops, 'head request,' 'sector high,' and 'sector low' can be turned on during an E2 cycle when starting read and write operations.
- 'Head request' is set off by a 1 in the U-register position 13.
- 'Sector high' and 'sector low' are set on by 1's in U-register positions 14 and 15, respectively.
- 'Head request' off results in 'head select' FF being off, and the lower head is used.
- 'Sector high' and 'sector low' determine which of the four sectors are read or written by the selected head.
- Maintenance diagrams: XF501 and XF511.

Assume that sector 7 (binary 111 in positions 13, 14, and 15 of the IOCC control word) is to be read or written. During E2 when the function code is start (initiate) read, the sector register is loaded. Bit 13 = 1 causes the turning off of the 'head request' FF by the 'file load SP' pulse. Then 'read/write select' turns off the 'head select' FF, resulting in the selection of head 1 (lower).

Both 'sector high' and 'sector low' in the sector register turn on when sector 7 is to be read or written. Then comparison of these two flip-flops with the sector counter determines when 'sector equal' becomes active.

The sector pulse that occurs at the end of the 'sector equal' signal indicates the start of the selected sector.

Sector Counter

- This two-position binary counter indicates the sector that is available to the read/write heads.
- 'Index' resets both 'sector high' and 'sector low' off.
- Four sector pulses step the sector counter.

- Sector counter is compared with the sector register. 'Sector equal' then allows read or write operation.

- Maintenance diagram: XF501.

The disk activates eight sector pulses and one index pulse during each revolution. The adapter circuits include a FF circuit to block every other one of the eight pulses. Consequently, the disk surfaces are effectively divided into four sectors. The four sector pulses that are not blocked step the sector counter. Thus, the counter steps in time with the sector available to the read/write heads.

The end of each of the four sector pulses fires the read/write singleshot and changes the state of the counter 'sector low' FF. Because 'index pulse' resets both FF's off, 'sector low' turns on at the end of the next sector pulse that is not blocked. 'Sector high' turns on when 'sector low' turns off at the next sector pulse, in the standard binary counter manner.

Figure 4-9 shows the relationship between the sector counter and the sector passing the read/write heads. While sector 3 passes the read/write heads, the sector counter is at 00. Assuming an instruction to read or write sector 0, 'sector equal' is active until the end of the next sector pulse. The combination of 'sector pulse' and the last portion of 'sector equal' starts the read or write operation. Sector 0 is just starting to pass the read/write heads.

## ADAPTER OPERATIONS

- The IOCC function codes that are applicable to the single disk storage are control, sense, initiate write, and initiate read.

The description of an XIO instruction is in Chapter 3. The area code assigned to the single disk storage drive 0 is decimal 4. Adapter circuit operations for the applicable IOCC function codes follow.

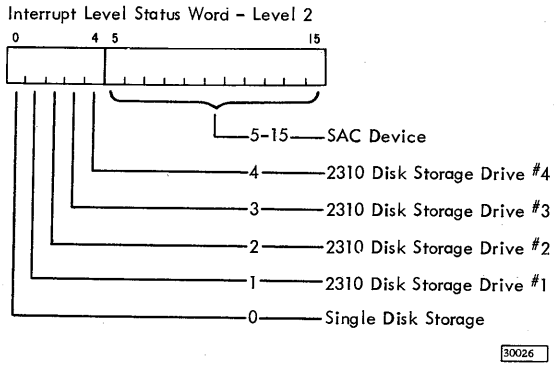
### Sense Interrupt

- Control word of the IOCC contains only the function code.

- Address word of the IOCC is used.

- Single disk storage interrupts on level 2.

The sense interrupt function code causes the ILSW for the active interrupt level to be set in the CPU accumulator. In the case of the single disk storage, the level 2 ILSW is transferred.



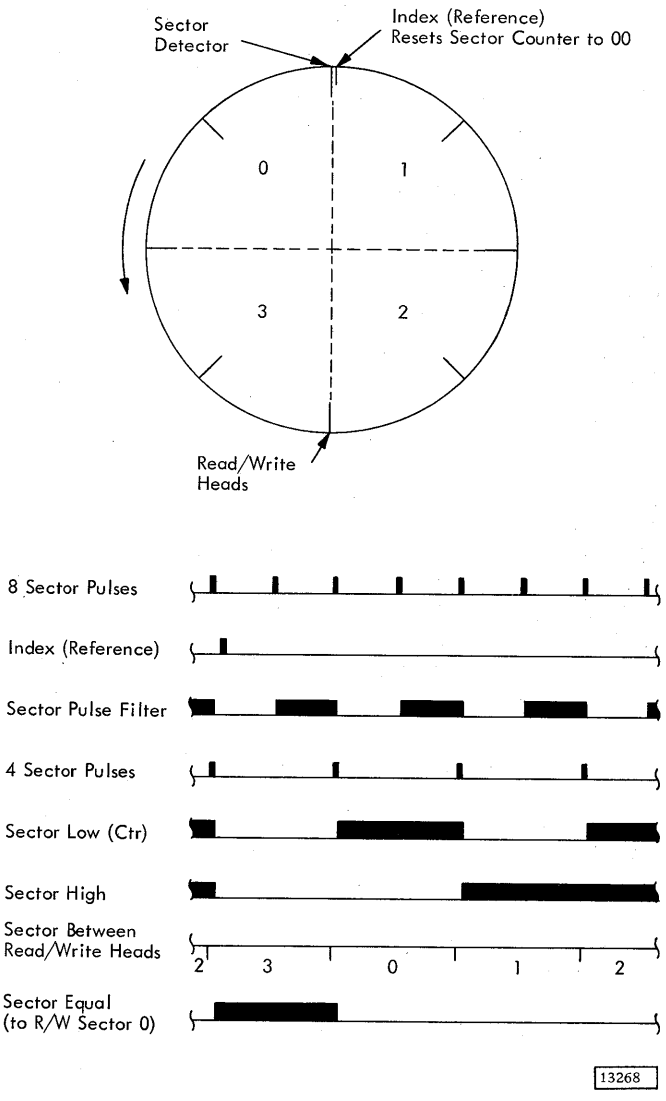
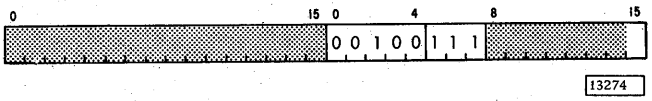




Figure 4-9. Sector Counter Operation

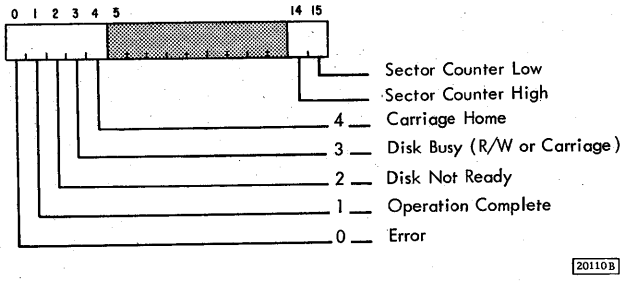
Sense Device

- Area code 4 and sense device function code set the DSW in the CPU accumulator.
- Modifier bit 15 = 1 turns off certain status conditions.

The IOCC for area 4 and the sense device function code is:




Execution of the instruction sets the DSW for the single disk storage in the CPU accumulator:




When bit 15 of the IOCC control word is a 1 and 'op complete' is on, the 'error' FF receives a reset pulse to turn it off. Also, 'op complete' turns off. The status conditions which may set 1's in the DSW are described in the following paragraphs.

Bit 0 -- Error: Several conditions can set this position to 1:

1. 'Error' FF is on. This FF is turned on when modulo 4 (data) error is detected during a read or read-check instruction. 'Error' is set also if sector pulse comes while reading or writing. This indicates that word count was too large.
2. 'Power unsafe' is active. This is active if +3V is not available after 90-second time delay. Also sets DSW bit 2.
3. 'Write select error' is active. This is a line from single disk storage drive 0, indicating 'write select error' condition in device.

Bit 1 -- Operation Complete: This position is set to 1 when the 'op complete' FF is on, having been turned on by:

1. Reading or writing number of words specified by word count.
2. Sector pulse while 'read/write condition' is active.
3. End of 'access busy' in seek operation.

Bit 2 -- Disk Not Ready: This position is set to 1 when the 'file ready' line from drive 0 is inactive. Bit 2 is set along with bit 3 when the adapter 'file busy' line is active. Bit 2 is also set with bit 0 when 'power unsafe' is active.

Bit 3 -- Disk Busy: This position is set to 1 along with bit 2 when 'file busy' is activated by 'read/write request,' 'read/write select,' or 'access busy.'

Bit 4 -- Carriage Home: This position is set to 1 when the access mechanism is in the home position. At this time, the 'access home latch' is off.

Bit 14 -- Sector Counter High: This position is set to 1 when the counter 'sector high' FF is on.

Bit 15 -- Sector Counter Low: This position is set to 1 when the counter 'sector low' FF is on. Bits 14 and 15 can be analyzed by the program to determine which sector is between the read/write heads when the disk is operating.

Control

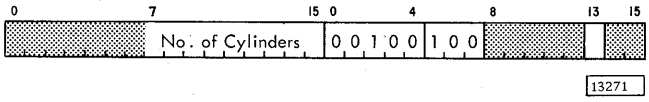
- Area code 4 and control function code cause access mechanism movement.
- Modifier bit 13, in the control word of the IOCC, specifies the direction of movement: 0 towards center of disk, 1 towards outer edge.
- Address word specifies the number of cylinders of movement, and is set into the word count register in 1's complement form.
- If the number is odd, the first access movement is a single track. Increment word count register +1.

- If the number is even, each movement is two tracks. Increment word count register +2.

- When word count register is all 1's, accessing stops. Turn on 'op complete' to request level 2 interrupt.

- Maintenance diagrams: XF501 and XF521.

At T4 time of the E2 cycle, 'XIO control' and 'area 4' activate the 'load word counter sample' line, loading the word count register with the 1's complement of positions 7 through 15 in the IOCC address word. The IOCC is:




Then, if 'file ready' is active, and the count that was just set into the word count register is not zero (full word count), a 'file load SP' (pulse) turns on 'access control' and sets access direction. Either control or initiate write function code can activate 'file load SP.' 'Access control' activates 'access busy' and turns on the 'access drive latch,' if 'access ready' is active. Turning on the 'access drive latch':

1. Increments word count register (+1 or +2 for odd or even word count, respectively).
2. Starts usage meter.
3. Activates 'access drive' line to single disk storage drive, starting motion of access mechanism. Direction of motion depends on state of 'head request' FF, which can be called access direction FF in this operation. When FF is on, motion is towards center of disk. If word count register bit 15 is off before incrementing, 'step mode' causes single-track access movement.

The single disk storage drive circuits deactivate 'access ready' some time during access motion, turning off the 'access drive latch.' When 'access ready' again becomes active, the 'access drive latch' turns on again, unless 'full word count' has been activated. Access movements of two tracks each continue until 'full word count' becomes active. Then, instead of turning on 'access drive latch,' 'access ready' turns off 'access control,' starting the 22.5-ms 'settle' singleshot.

Turning off 'access control' deactivates 'access busy,' thus turning on 'op complete' if 'full word count' is active. 'Op complete' then activates 'interrupt request level 2.'

Even though 'XIO start read' or 'XIO start write' may have turned on 'read/write op,' 'read/write select' cannot turn on until the 'settle' SS has timed out. Therefore, the read/write heads have time to settle after a seek, before they are used to read or write.

The adapter circuits prevent the access mechanism from moving backwards from the home position as a result of an instruction. Remember that the 'access home latch' is off when the mechanism is in home position or is turned off when the mechanism reaches home during access movement. When 'access home latch' is off, activating 'access ready' turns off 'access control,' and no motion can occur.

Initiate Write

- XIO instruction IOCC selects the disk storage adapter and addresses the word count word in core storage.

- Start write operation when sector called for in the IOCC reaches the read/write heads.

- Write field of 0's and sync word.

- Request level 0 cycle steal cycle for each data transfer.

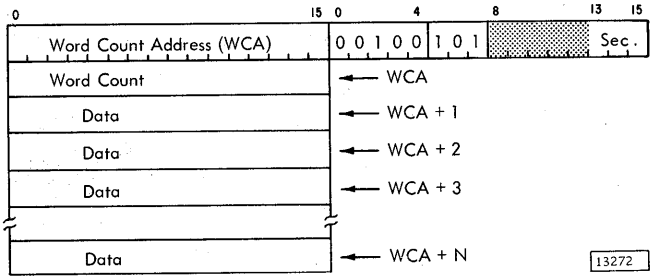
- Write one sector or less of data from core storage onto the disk. When less than a sector of data is written, fill remainder with 0's (clock bits only).

- Write check bits with each word.

- Maintenance diagrams: XF401, XF501, XF511, and XF701.

Area 4 and a function of initiate write cause the number of words specified by the word count to be written in disk storage, beginning at the first word of the sector indicated by modifier bits 13-15. The address word of the IOCC contains the word count address (WCA). The data is transmitted from core storage location WCA + 1 and ascending addresses. A full sector can contain 321 words. Succeeding sectors, or parts of sectors, require an 'XIO initiate write' for each one.

The IOCC and the data table used in the operation are:




Data is transferred during CS cycles, and an operation-complete interrupt, level 2, occurs when the number of words in the word count has been transferred.

The operation resulting from 'XIO initiate write' consists of the following steps.

1. During E2, load word count address into file address register and load sector register.
2. Cycle steal and set word count register.
3. Compare sector register with sector counter until equal condition occurs.
4. At end of sector equal, write field of 0's and sync word.
5. Cycle steal and load file data register with data word to be written.
6. Write data word, shifting bits towards bit 15 end of file data register.
7. Repeat steps 5 and 6 until 'full word count' or sector pulse signals 'op complete.'
8. Interrupt on level 2 and reset adapter circuits.

E2 Cycle

'Area 4,' 'U register 5,' and 'XIO time gate and not U6' activate 'file load gate.' Then a T6 pulse activates both the 'load file address' line and the 'file load SP.' 'Load file address' line sets the address word from the B-register into the file address register. The address word specifies the location of the first word in the data table (contains the word count). The 'file load SP' sets the sector register from U-register positions 13, 14, and 15, and turns on the 'CS request' FF. The 'file load SP' also turns on the 'read/write request' and 'read/write op' flip-flops.

First CS Cycle

'CS levels' prevents the M-register from addressing core storage, during CS cycles. Instead, 'CS level 0' causes the file address register to address core storage and set the word count from the data table into the B-register. The 'bit counter gate' is not yet active, so X6 time activates the 'load word counter SP.' Contents of the B-register set the word count register. The end of X6 time steps the file address register.

Sector Comparison

When the sector register and sector counter are equal (Figure 4-9), the leading edge of 'sector pulse' turns on 'read/write select.' Turning 'read/write select' on turns off 'read/write request' and turns on the 'write select latch.' 'Write select latch' and 'sector pulse' activate 'load sync word,' which turns on the file data register 'bit 0' FF (sets a 1 in that position). The end of sector pulse starts the 250-ms single shot to step the sector counter and cause writing of 0's before the sync word.

Writing Sync Word

When the read/write SS times out, a 'counter SP' pulse turns on the 'write sync' FF. 'Counter SP' is a pulse from the disk storage drive, gated, in this case, by 'write select.' 'Write sync' activates 'bit counter gate,' and succeeding 'counter SP's step the bit counter. Also, the 1 that was set in 'bit 0' shifts toward the bit 15 end of the file data register. During the last bit time before bit counter E, the 1 is in bit 15 position of the file data register. 'File bit 15' activates 'write data gate' to cause the writing of the 1 on the disk. Also, 'check counter gate' is activated; so the next 'counter SP' turns on the low-order file check counter FF. In other words, a 1 is set in the file check counter.

Turning off bit counter 'D' FF turns on the 'CS request' FF, requesting a level 0 cycle steal. Then during the four bit-times of bit counter E, the file check counter provides three 1's and a 0 for the sync word. The end of 'bit counter E' turns off the 'write sync' FF and turns on 'read/write condition.'

Data Transfer CS Cycles

The second cycle steal cycle of the operation transfers the first data word from the data table. During level 0 CS cycles, the file address register, instead of the M-register, addresses core storage. The data word that is read out sets the B-register. X4 time activates a 'register load SP,' setting the same word into the file data register (all bits in parallel). The end of X6 steps the file address register in preparation for the next data word transfer.

The CS cycle must have progressed far enough to load the file data register by the end of bit counter E-time.

Writing Data

After the sync word has been written (bit counter E-time), writing of a data word starts. File data register 'bit 15' FF determines whether a 0 or a 1 is written in each bit cell until bit counter 'E' FF is on.

'Read/write condition' was turned on at the end of writing the sync word, and it stays on as long as writing of data continues. At the start of bit counter D-time, 'increment word count' is activated, stepping the word count register.

Operation Complete

Writing of data, with interspersed data transfer CS cycles, continues until a full word count is sensed or 'op complete' is turned on by a sector pulse. The latter case occurs when an attempt is made to write too many data words in a sector.

While the last word is being written, the word count register steps to 'full word count' at the start of bit counter D-time. 'Full word count' then prevents turning on 'CS request' at the end of bit counter D. No more data transfer is required.

At the end of writing check bits for the last data word (end of bit counter E), a full word count turns on the 'op complete' FF. 'Op complete' activates 'interrupt request level 2' and 'file reset.' 'Op complete' stays on until 'XIO sense reset 15.' A sense DSW, area 4, with 'op complete' on sets a DSW bit 1. The 'file reset' lines turn off most of the same flip-flops in the adapter as a power on 'DC reset.'

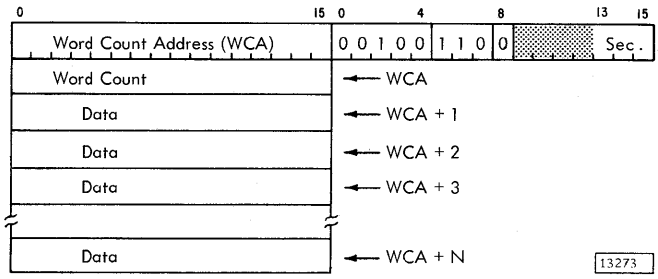
'Write select,' however, does not turn off until the next sector pulse. Clock bits only are written in the rest of the sector after 'op complete.'

Initiate Read

- XIO instruction IOCC selects the disk storage adapter and addresses the word count word in core storage.
- Start the read operation when sector called for in the IOCC reaches the read/write heads.
- Read 0's until the sync word is read. Sync the bit counter with the sync word.
- Request a level 0 cycle steal cycle for each data transfer.
- Read one sector or less of data from the disk and transfer each data word to core storage.
- Check each data word that is read.
- Read and read check operations are the same except that no cycle steal requests are made in the read check operation (no data transfer).
- Maintenance diagrams: XF401, XF501, XF511, and XF711.

Area 4 and a function of initiate read cause the number of words specified by the word count to be read from the disk storage sector (0-7) identified by modifier bits 13-15. The address word of the IOCC contains the word count address (WCA), and modifier bit 8 determines whether the function is read (0) or read check (1).

A full sector can be read by one XIO instruction. Succeeding sectors, or parts of sectors, require an 'XIO initiate read' for each one. The IOCC and the data table used in the operation are:




Data is transferred during CS cycles, and an operation-complete interrupt level 2 occurs when the number of words in the word count has been transferred.

The operation resulting from an 'XIO initiate read' consists of the following steps. Through the finding of the sector to be read, the operation is nearly the same as 'XIO initiate write.' The major difference is that 'read/write op' is turned off by the 'file load SP.'

1. During E2, load word count address into file address register and load sector register.
2. Cycle steal and load word count register.
3. Compare sector register with sector counter until equal condition occurs.

4. At end of sector equal, read field of 0's until sync word is reached. Then read sync word to signal that reading of data words is to start.
5. Read data word, bit by bit, and shift bits towards bit 15 end of file data register.
6. When full 16-bit data word is in file data register, stop shifting. Cycle steal and transfer data word to CPU core storage.
7. Repeat steps 5 and 6 until 'full word count' or sector pulse turns on 'op complete.'
8. Interrupt on level 2 and reset adapter circuits.

E2 Cycle

Activating 'XIO start read and area 4' is another means of activating the 'file load gate' line. A T6 pulse activates both the 'load file address' line and the 'file load SP.' The former line sets the address word from the B-register into the file address register. The address word specifies the location of the first word in the data table (contains the word count). The 'file load SP' pulse sets the sector register from U-register positions 13, 14, and 15, and turns on the 'CS request' FF. The 'file load SP' also turns on the 'read/write request' FF. In a read operation, the 'read/write op' FF is not turned on.

First CS Cycle

'CS levels' prevents the M register from addressing core storage, during CS cycles. Instead, 'CS level 0' causes the file address register to address core storage and set the word count from the data table into the B register. The 'bit counter gate' is not yet active, so X6 time activates the 'load word counter SP.' Contents of the B-register set the word count register. The end of X6 time steps the file address register.

Sector Comparison

When the sector register and sector counter are equal, the leading edge of 'sector pulse' turns on the 'read/write select' FF. 'Read/write select' turns off the 'read/write request' FF, and with the 'read/write op' FF off, 'read/write select' activates 'file read select.' When 'four sector pulses' is deactivated, the 'read select' line to disk storage drive 0 becomes active. In the adapter, 'read select' gates the turn-on of the 'input trigger' and allows each read clock pulse to fire 'counter SP.'

Reading the Sync Word

While the field of 0's is being read (approximately 250 microseconds), 'counter SP' pulses shift the 0's through the file data register. However, the bit counter is not yet advancing. Figure 4-10 shows the sensing of the four 1's that are contained in the sync word. Each 1 that is read turns on the 'input trigger' FF. Also, each 'counter SP' activates a 'shift SP A' and a 'shift SP B.' The four 1's are set into file data register (FDR) position 0, and they are shifted toward the FDR 15 position.

The 0 following the four 1's does not turn on the 'input trigger.' At the time of the next 'shift SP,' all conditions are active to turn on 'read/write condition.' Turning on 'read/write condition' activates the 'bit counter gate'; so following 'counter SP' pulses can step the bit counter.

Reading Data and Data Transfer

Reading of data words starts when 'read/write condition' turns on and the bit counter starts stepping. The state of the 'input trigger' determines whether a 0 or a 1 is set into FDR 0 by the 'shift SP.' Bits shift towards the bit 15 position of FDR as long as 'bit counter E' is not active.

Turning off bit counter 'D' FF turns on the 'CS request' FF, when the operation is read and not read-check. The data word in the file data register activates the lines of the I/O bus when the 'level 0 data gate' is active. 'X3' of the CS level 0 cycle, with 'file data entry gate' active, sets the data word into the B register. The data transfer must be complete by the end of bit counter E time. X6 of the CS cycle increments the file address register in preparation for the next data transfer.

Even though shifting in FDR stops during bit counter E time, 1's that are read continue stepping the check counter. The number of 1's in a disk storage word must be divisible by 4. When this is the case, both file check counter flip-flops are off at the end of bit counter E-time. Either check counter FF being on blocks the 'zero check count' line. Then the end of bit counter E, with 'read/write condition' active, turns on the 'error' FF.

'Bit counter E,' in a read operation, increments the word count register towards an "all 1's" condition.

Operation Complete

Reading of data, with interspersed data transfer CS cycles, continues until a full word count is sensed or

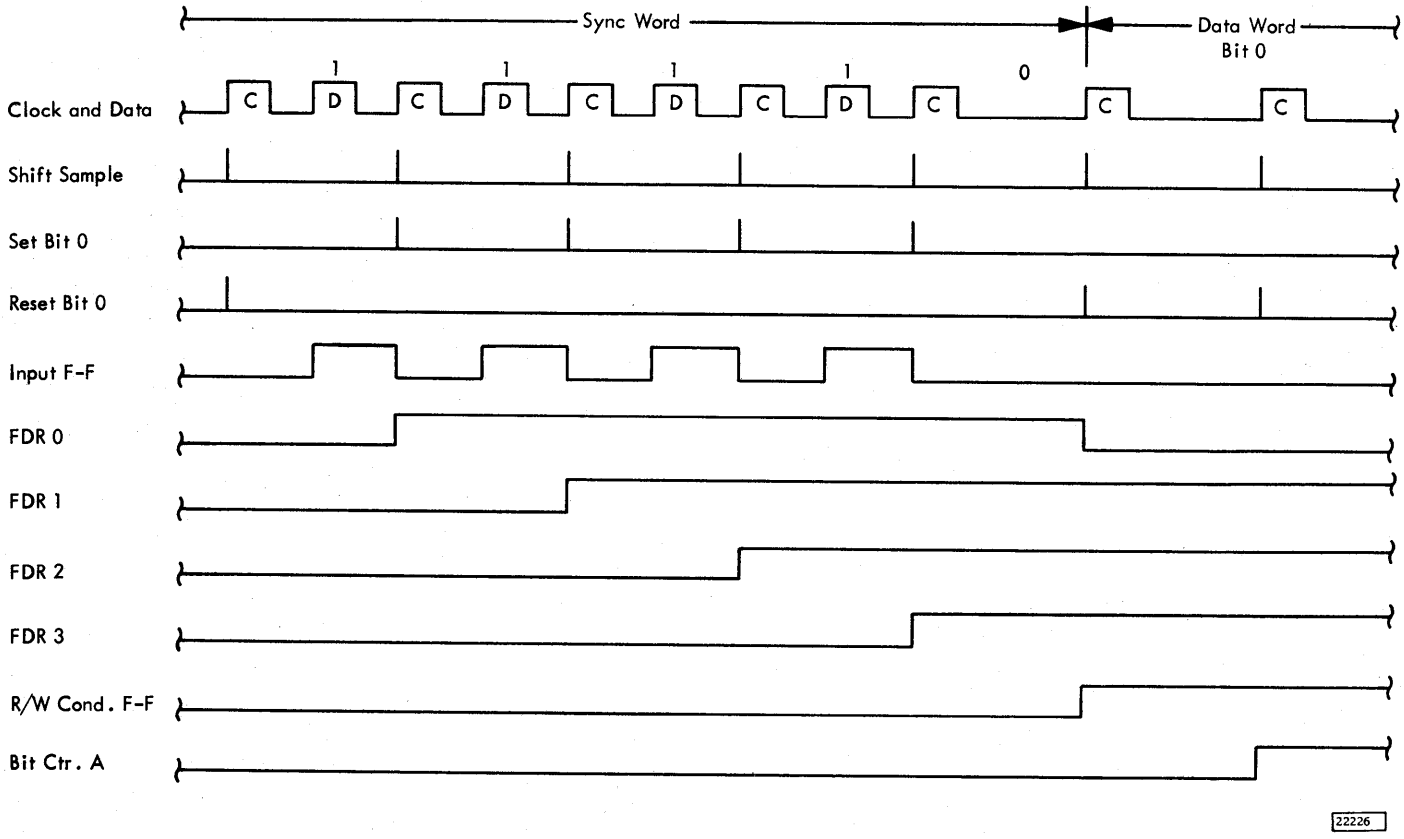



Figure 4-10. Reading the Sync Word

'op complete' is turned on by a sector pulse. The latter case occurs when an attempt is made to read more data words than can be contained in one sector.

The word count register steps to full word count after the last of the specified number of words has been read and transferred to the CPU. After reading the check bits for the last data word, the end of 'bit counter E' turns on the 'op complete' FF.

'Op complete' activates 'interrupt request level 2' and 'file reset.' 'Op complete' stays on until 'XIO sense reset 15.' A sense DSW, area 4, with 'op complete' on sets a 1 in DSW bit 1 position. The 'file reset' lines turn off most of the same flip-flops in the adapter as a power on 'DC reset.'

Read-Check Operation

The IOCC used to initiate a read-check operation contains a 1 in bit 8 position of the control word. The 'file load SP' turns on the 'read-check' FF when 'U bit 8' is active. When 'read-check' is on, only the 'file load SP' can turn on 'CS request.' Loading of the word count register during the resulting CS cycle is standard.

All further requests for CS cycles are blocked because 'read-check' is on. Data within the sector is read in the regular manner, and each data word is checked for a modulo 4 condition. However, no data is transferred to the CPU because no further CS cycles occur.

## PRIMARY POWER INPUT

- AC primary power requirement depends on the system model and configuration.
- Models 1A, 1B, 2A, and 2B can operate on 115 Vac, one phase, three wire, 60 Hz. However, the 208V or 230V feature is optional on these models.
- Models 2C, 2D, 3B, 3C, and 3D require 208/230 Vac primary power.
- When a 1231, 1442 model 5, or a 2501 is to be attached, primary power must be 208/230 Vac.
- When an 1133 is included in the system, power for the 1131 is provided via the 1133. The 1131 uses one phase of the 208/230 Vac, three-phase, four-wire, 60 Hz primary power required by the 1133.
- Primary power for World Trade Corporation (WTC) machines is 195/220/235 Vac, one-phase, 50 Hz.

Figure 5-1 shows the manner in which various features and model numbers affect the primary power requirements of the 1131. For example, 208 Vac (or 230 Vac) power is shown as an optional feature for models 1A, 1B, 2A, and 2B. However, the same primary power is shown as a basic feature (required) for models 2C, 2D, 3B, 3C, and 3D.

For WTC 1130 systems which include an 1133, the primary power can be 380/405 Vac, three phase, 50 Hz. However, the power supplied to the 1131 must be 195/220/235 Vac.

## POWER SUPPLY OUTPUTS

- DC logic voltages of +3 Vdc, -3 Vdc +6 Vdc, +48 Vdc, and +12 Vdc are required for all systems.
- Certain additional dc voltages are required for certain features.
- DC power supplies are either MPS (medium power standard) or mid-pac (middle power package). Power sequencing is different for the two types.
- Internal operation of power supplies is described in the SLT Power Supplies manual, 223-2799.
- AC voltages other than line voltage are: 24 Vac for power sequencing, 41 Vac for use meters, and 7.25 Vac for console indicator lamps.
- Convenience outlets, blowers, and fans use 115 Vac, regardless of line voltage.

In addition to the direct current voltages required for logic circuits of the basic 1131, some features require additional power supplies. For example, when expanded core storage (greater than 8k) is included in the system, additional power supplies can be required in the "blister." The voltages required depend on the amount of additional core storage, the core storage cycle time, and the type of power system. See system diagram page YP009 for a chart of additional power supplies required.

Both the MPS and mid-pac (called "midpack" in system diagrams) power systems have been used. The type of power system used in the 1131 also affects the additional power supplies required by features. The power sequencing circuits are different for the two types of power systems.

Regardless of input line voltage, various machine components require definite ac voltages. Examples are: 24 Vac for power sequencing circuits, 41 Vac for use meters, and 7.25 Vac for console indicator lamps. Step-down transformers with adjustment taps and jumpers can accommodate the various line voltages and provide the required operating voltages.

When the input line voltage is 115 Vac, line voltage is used directly for power supply and gate fans and blowers. Machines with 208/230 Vac line voltage have step-down transformers to provide the required 115 Vac.

## POWER SEQUENCING

- Circuits activate +48 Vdc and +12 Vdc last and deactivate them first in normal power sequencing.

| Basic Features | 1131 Model 1* | 1131 Model 2* | 1131 Model 3 |
|---|---|---|---|
| Console Keyboard | A and B | A, B, C, and D | B, C, and D |
| Console Printer | A and B | A, B, C, and D | B, C, and D |
| 3.6 μs Core Storage | A and B | A, B, C, and D | |
| 115 vac Power | A and B | A and B | |
| Single Disk Storage | | A, B, C, and D | B, C, and D |
| 2.2 μs Core Storage | | | B, C, and D |
| 208 vac Power (60 Hz) | | C and D | B, C, and D |
| **Optional Features** | | | |
| 208 vac Power (60 Hz) | A and B | A and B | |
| 1055 Paper Tape Punch | A and B | A, B, C, and D | B, C, and D |
| 1132 Printer | A and B | A, B, C, and D | B, C, and D |
| 1134 Paper Tape Reader | A and B | A, B, C, and D | B, C, and D |
| 1442 Models 6 and 7 Card Read Punch | A and B | A, B, C, and D | B, C, and D |
| 1627 Plotter | A and B | A, B, C, and D | B, C, and D |
| Synchronous Communications Adapter | A and B | A, B, C, and D | B, C, and D |
| Storage Access Channel I (SAC) | A and B | A, B, C, and D | B, C, and D |
| 1133 Multiplexer Control | | A and B ‡, C and D † | B, C, and D† |
| 2310 B Disk Storage | | A, B, C, and D** | B, C, and D** |
| 1403 Printer | | A, B, C, and D** | B, C, and D** |
| Storage Access Channel II (SAC II) ¤ | | A, B, C, and D** | B, C, and D** |
| 2501 Card Reader *** | | A and B ‡ , C and D | B, C, and D |
| 1442 Model 5 Card Punch | A and B | A and B ‡ , C and D | B, C, and D |
| 1231 Optical Mark Page Reader *** | | A and B ‡ , C and D | B, C, and D |

* Model 1 can be field changed to Model 2 or 3.
Model 2 can be field changed to Model 3.
(Only systems with midpack power supplies can be upgraded, in the field, to a Model 3D.)

† Storage Access Channel 1 required on all models.

‡ 208 vac or 230 vac 60 Hz power feature required in U.S.A. on Models A and B.

** 1133 Multiplexer Control required on all models.

¤ Required to connect any device or devices previously connected to SAC when an 1133 Multiplexer Control is connected to SAC.

*** A system can not have both a 2501 and a 1231

16213 A

Figure 5-1. Basic and Optional Features for the 1130 System

- Loss of a single dc voltage deactivates all dc power supplies.
- Circuits disable the power on/off switch to prevent power on attempts with a logic voltage missing.

Although the objectives of power sequencing for both MPS and mid-pac power systems are the same, the circuits differ somewhat.

MPS

Power On Sequence

Closing the line circuit breaker (Figure 5-2) provides line voltage to transformer T1 and 115 Vac to the convenience outlets. Then, turning on the power on/off switch, with the emergency power off (EPO) and CE switches normal, picks contactor K1. The K1 points apply line voltage to all power supply inputs except the +48 Vdc and +12 Vdc supplies. When the line voltage is 115 volts, it picks RLY 2 and drives the gate and power supply blowers and fans. When the line voltage is 208 or 230 volts, transformer T2 provides the 115 volts to pick RLY 2 and drive the blowers.

The outputs of the +6V, +3V, and -3V supplies pick RLY B, RLY C, and RLY D, respectively. Points of these relays provide a hold circuit for K1. Also when the +6V and -3V supply outputs reach operating levels, RLY 1 is picked. RLY 1 points complete the circuits to the inputs of the +48V and +12V power supplies, and to pick RLY A.

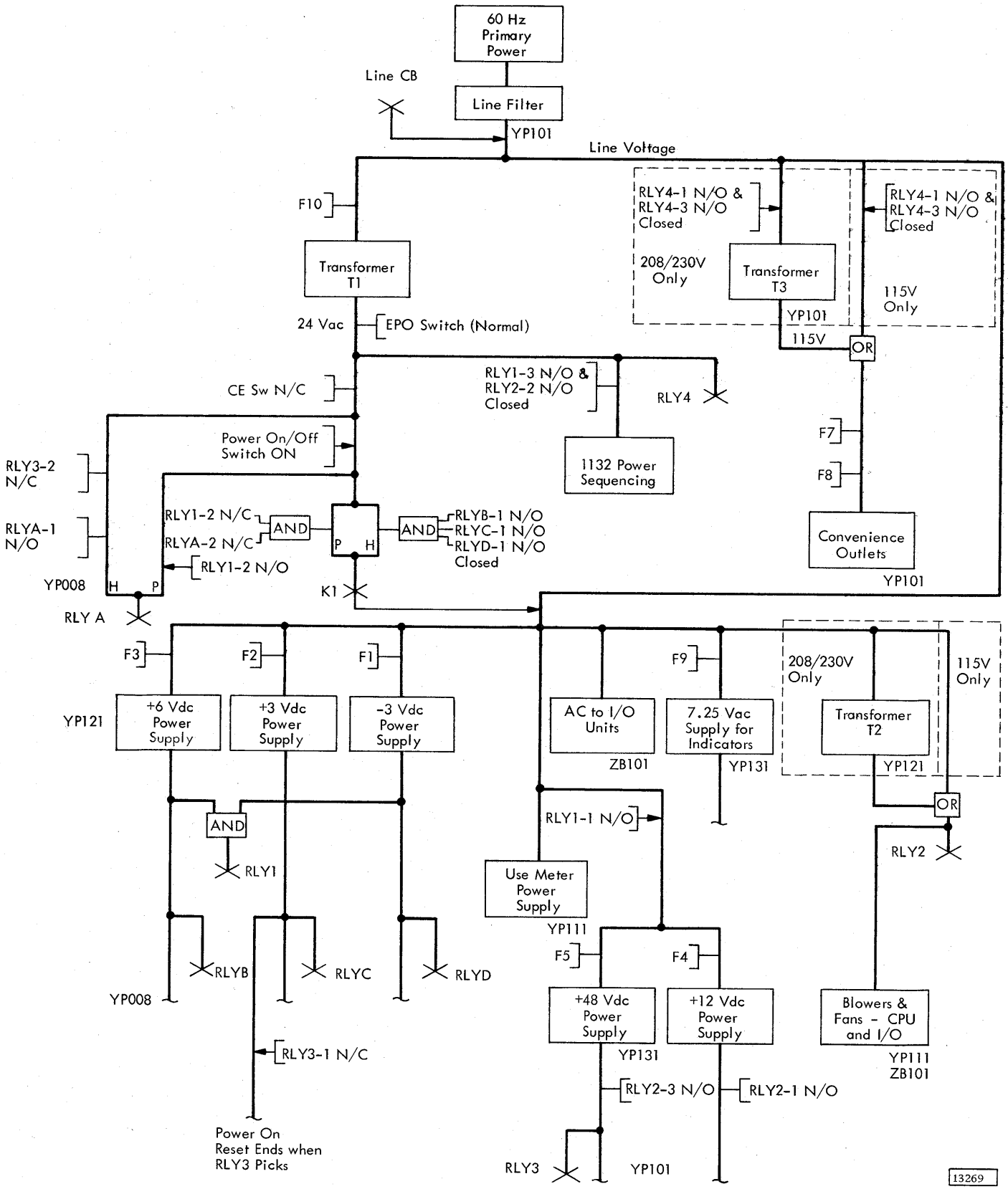



Figure 5-2. Power Sequencing (MPS)

Picking RLY 1 also opens the pick circuit to K1.

When the 48V power supply output reaches operating level, RLY 3 picks. RLY 3-1 n/c point then opens the 'power on reset' line. Machine operation can start. In the normal power on sequence, RLY 2 picks first, then RLY 1, and RLY 3 picks last.

Power Off Sequence

The usual power off sequence (Figure 5-2) results from setting the power on/off switch to the off position. The hold circuit to K1 and pick circuit to RLY A are opened by this action. RLY A drops before RLY 3 so that the hold circuit to RLY A is not effective. Dropping K1 opens the input circuits to all power supplies and opens the pick circuit to RLY 2. Dropping RLY 2 opens the +48V and +12V power supply output circuits (before the +6V, +3V and -3V supply outputs deteriorate). The sequence of relays dropping is:

1. K1 and RLY A, practically together.
2. RLY 2.
3. RLY 3.
4. RLY 1.

A power off sequence also results from a loss of logic voltage (+6V, +3V, or -3V). In this case, RLY A does not drop at the same time as contactor K1. Then when RLY 3 drops, the hold circuit for RLY A is closed. RLY A remaining picked keeps the pick circuit for K1 open, and K1 cannot be repicked until RLY A is dropped. RLY A can be dropped by turning off the CE switch or the mainline CB.

Mid-Pac

Power On Sequence

Closing the line circuit breaker (CB) provides the line voltage (Figure 5-3) to transformer T1. The sequencing 24 Vac picks R3, if the EPO switch is normal. R3 points activate the convenience outlets, either directly (115V line) or by closing the input circuit to transformer T3. Turning on the power on/off switch, with the CPO and CE switches normal, picks K1 through the normally closed RR1-2 points.

The K1 points apply line voltage to all power supply inputs and, when 115V, directly to the gate and power supply blowers and fans. When line voltage is 208V or 230V, transformer T2 output drives the blowers and fans. The two inputs to the mid-pac (midpack) bulk supply are separately fused. The +12V and 48V are regulated within the bulk supply, and the +6V, -3V and +3V outputs use regulator cards.

The regulated +6V, -3V, and +3V outputs pick RR2, RR3, and RR4, respectively. When all three outputs are at operating levels, the drop between +3V and -3V picks R1. Picking R1 picks R2, but prevents the pick of RR1. R2 points gate the +12V and +48V outputs to machine circuits. Thus, they are activated after +6V, -3V, and +3V.

When K1 picked, the delayed pick to TD1 relay started. After approximately 5 seconds, TD1 picks. TD1-2 opens the +3V 'power on reset' line. Machine operation can start. In the normal power on sequence, K1, R1, R2, and TD1 pick in that order.

Power Off Sequence

The usual power off sequence (Figure 5-3) results from setting the power on/off switch to the off position. R2 and TD1 drop before R1, because the +6V, -3V, and +3V outputs deteriorate slowly. Therefore, +12V and +48V are removed from machine circuits first. Also RR1 cannot pick because TD1-1 opens before R1-2 closes.

A power off sequence also results from a loss of logic voltage (+6V, -3V, and +3V). In this case, R1 drops while TD1 remains energized. RR1 picks and holds through its own points. RR1 remaining picked keeps the pick circuit to K1 open. No further power on sequence can occur until RR1 is dropped. RR1 can be dropped by turning off the CE switch or the mainline CB.

POWER ON RESET

- 'Power on reset' line is activated when the +3V power supply output reaches operating level during a power on sequence.

- 'Power on reset' activates 'dc reset 1,' 'dc reset 2,' and 'dc reset 3' lines until a sequence relay opens the circuit to the 'dc reset' lines.

- 'DC reset' lines set machine circuits to normal starting states (T-clock T7, X-clock at X7, registers reset, etc.)

- Picking RLY 3 (MPS) or TD1 (mid-pac) deactivates the 'dc reset' caused by 'power on reset.'

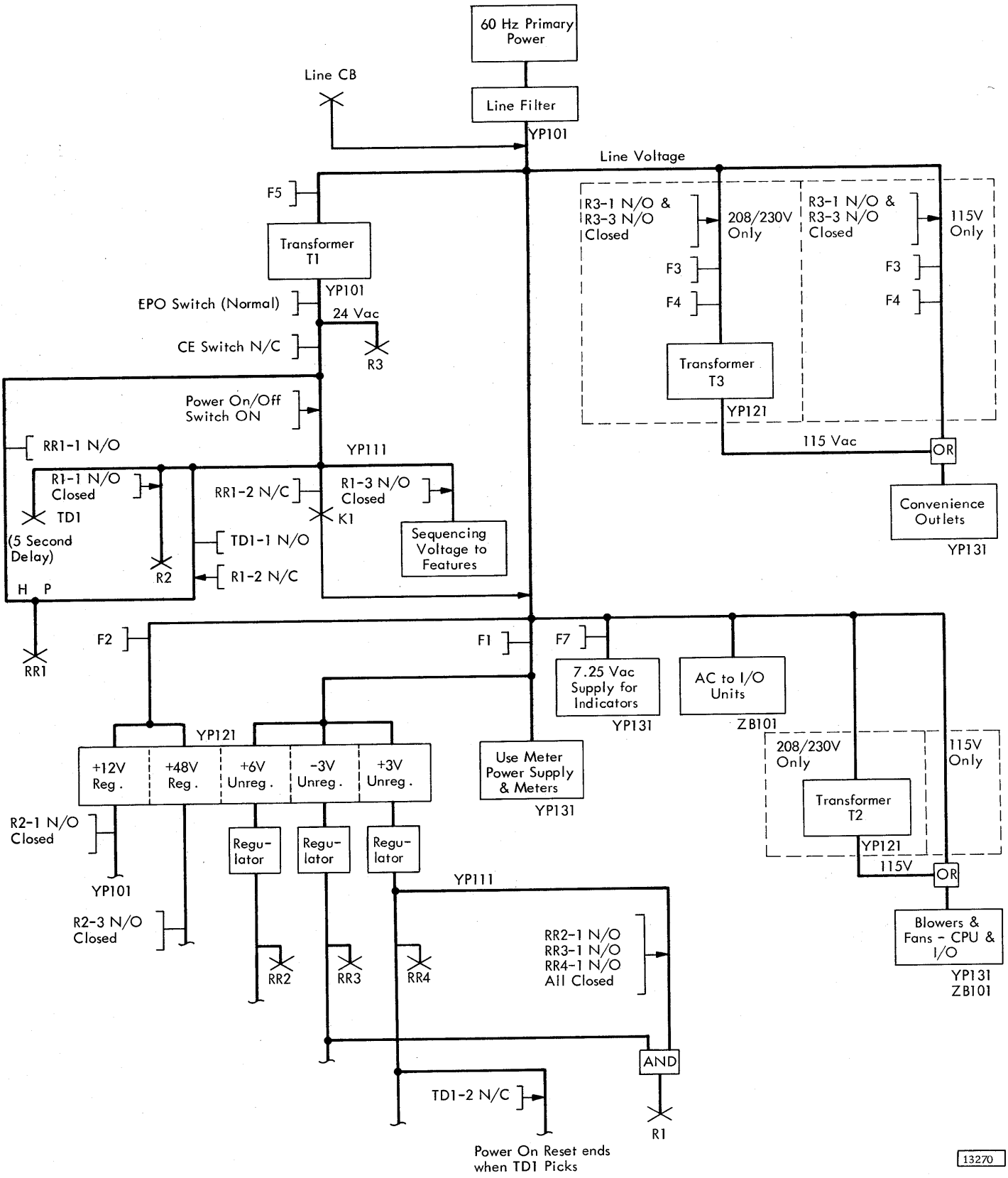



Figure 5-3. Power Sequencing (Mid-Pac)

## SECTION 1. CONSOLE

- The console is an integral part of the IBM 1311 Central Processing Unit.
- The console consists of the console keyboard, the console printer, and the console display panel.

The console keyboard mechanism is described in Chapter 2, and the data transfer from the console keyboard into core storage is described in Chapter 3. The keyboard function keys are described in this chapter. Also, the indicator and switch panels are described in this section.

CPU control of the console printer is described in Chapter 3. Data input from the bit switches that are mounted on the console printer cover is also described in Chapter 3. The other control keys mounted on the same cover are described in this section.

The indicators and rotary mode switch on the console display panel are described in this section.

### CONSOLE KEYBOARD (Figure 6-1)

#### Function Keys

Interrupt Request: Pressing this key fires the 25-ms keyboard SS-1 and gates the 'manual interrupt' FF. When the singleshot times out, the FF turns on, gating the console DSW bit 2 and activating 'level 4 interrupt request.' 'Manual interrupt' also fires keyboard SS-2, which activates 'keyboard restore.'

Restore Keyboard: Pressing this key energizes the two keyboard restore magnets.

Backspace: Pressing this key activates 'keyboard data B.' With the keyboard selected 'keyboard data B' fires the keyboard SS-1 and activates 'keyboard response gate.' SS-1 timing out turns on the 'keyboard response' FF, causing a level 4 interrupt. 'XIO read' then transfers a word with a 1 in bit 13 position only to core storage. Program analysis determines that the previous data word is to be replaced by the following data word.

Erase Field: Pressing this key activates 'keyboard data A.' The resulting operation is similar to backspace, except that the word transferred contains a 1 in bit 14 position only. Program analysis determines that the entire field entered since the manual interrupt request is to be deleted. A corrected message can then be entered into the same core storage positions.

End of Field: Pressing this key causes a level 4 interrupt. 'XIO read' transfers a word with only bit 12 = 1. (See "Backspace.") Program analysis determines that no further keyboard entry is to take place.

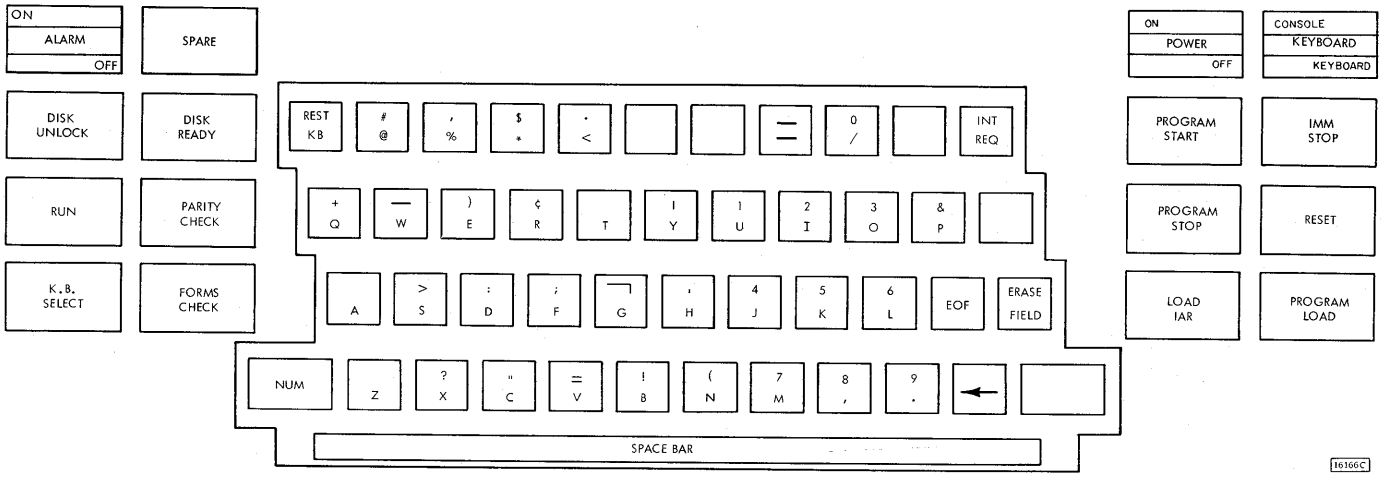



Figure 6-1. Console Keyboard

Numeric: Pressing this key causes the keyboard logic to activate 'keyboard data' lines as required for numeric shift characters. The key must be held down to continue activating numeric codes.

Indicator and Switch Panels

Indicators

Run: This indicator lights when the usage meter runs unless the 'CE key switch' line is active.

Parity Check: This indicator lights when the 'parity check' FF is on. The FF is turned on when either half of the B-register with its associated 'check bit' FF contains an even number of 1's.

Keyboard Select: This indicator lights when the 'keyboard select' FF is on. The FF is turned on by 'XIO control' and 'area 1' and is turned off by 'XIO read' and 'area 1.'

Forms Check: This indicator lights when the typewriter is not busy and has no paper closing the end-of-forms contact.

Disk Unlock: (Only on models having single disk storage drive 0). This indicator lights when the disk unlock solenoid is energized.

File Ready: (Only on models having single disk storage drive 0.) This indicator lights when contactor K1 in the disk storage logic picks.

Switches

Power: This switch controls the logic voltage sensing circuit and thus controls system power. See Chapter 5.

Keyboard: When set to CONSOLE, this switch activates 'console DSW bit 3' if 'XIO sense device' and 'area 1' are activated. Program analysis can then cause a branch to a subroutine to read the bit switches. When set to KEYBOARD, the 'console DSW bit 3' is not set to 1, and the program can branch to a subroutine to select the keyboard. Then the keyboard provides input data.

Alarm: (Only on machines with synchronous communications adapters.) This switch is a manual means of turning off the audible alarm.

Program Start Key: Pressing this pushbutton switch causes the machine to take one clock step or one machine cycle, or to continue to run, depending on the setting of the rotary mode switch.

Immediate Stop: Pressing this pushbutton switch turns on the '(CPU) stop latch,' the same latch that is turned on by a parity stop (parity error) condition. The 'stop latch,' when it is on:

1. Turns off the 'run' FF and prevents 'interrupt request' from turning it back on.
2. Prevents cycle stealing by any I/O device.
3. Signals I/O devices that the CPU is stopped.
4. Blocks the 'T7-X7 not stop latch' line.

Thus, cycle steal cycles and interrupts are blocked, and data transfers between I/O devices and CPU cannot take place. Data may be lost, and a program restart is normally required.

Program Stop: Pressing this pushbutton switch turns on the 'program stop latch.' This results in setting console DSW bit 0 to 1 and causing a level 5 interrupt request. The ILSW for level 5 also has a 1 in the bit 0 position. With usual programming, the CPU branches to a wait loop when the level 5 interrupt request has been serviced.

Reset: Pressing this pushbutton switch activates the 'dc reset (1, 2, and 3)' lines if the 'run program load dot on run' is not active at the time. To activate the reset lines it is necessary to block both AND's which are OR'ed.

Load IAR: This pushbutton switch is effective only when the rotary mode switch activates the 'load/display mode' line. If the rotary switch is set to LOAD, the bit switches are gated to set the B-register FF's. Then pressing the load IAR key causes 'B to I SP 0-7' and 'B to I SP 8-15,' which set the contents of the B-register into IAR.

Program Load: Pressing this pushbutton switch causes loading of data from a card or from paper tape into core storage, starting at position /0000. Jumpers in the CPU circuits determine which device is used, a 1442 card reader, a 2501 card reader, or an 1134 paper tape reader. In either case, pressing the key turns on the 'program load' FF, provided the 'reset condition' FF is not on. 'Program load' then:

1. Blocks 'gate interrupt' line.
2. Activates 'program load to SRP,' 'program load to 2501,' or 'program load to PT.'

3. Activates 'entry gate,' simulating 'XIO read' function code in an IOCC.

If a card reader is being used, a single card is read and stored in core storage locations /0000 through /004F. IAR is then set to /0000 to start execution of the program which was punched in that card.

If a paper tape reader is used, tape is read and stored in location /0000 and progressively higher locations. Only tape channels 4, 3, 2, and 1 are used for data. Four tape columns provide the 16 bits to load one core location. Reading continues until the paper tape reader senses a hole punched in channel 5. IAR is then reset to /0000 to start execution of the program which was punched in the tape.

CONSOLE PRINTER (Figure 6-2)

Mounted on the console printer cover are these controls for the printer mechanisms:

1. Tab set and clear button operates link to set or clear tabs.
2. Tab, space, and return keys operate linkages which cause their respective printer functions.

The bit switches, mounted on the same cover may be used for entering data into the CPU under program control or under control of other switches.

CONSOLE DISPLAY PANEL (Figure 6-3)

Indicators

- Except where otherwise noted, the indicators light when the corresponding flip-flop is on.
- In registers, a lighted indicator represents a binary 1.

Instruction Address (I) Register (IAR): This row of indicators displays the address in IAR (I-register). 'I bit 0' has no indicator.

Storage Address (M) Register: This row of indicators displays the address in the M-register. 'M bit 0' has no indicator.

Storage Buffer (B) Register: This row of indicators displays the word in the B-register. At the end of a T-clock cycle, the word displayed is the word that was written in core storage during that clock cycle.

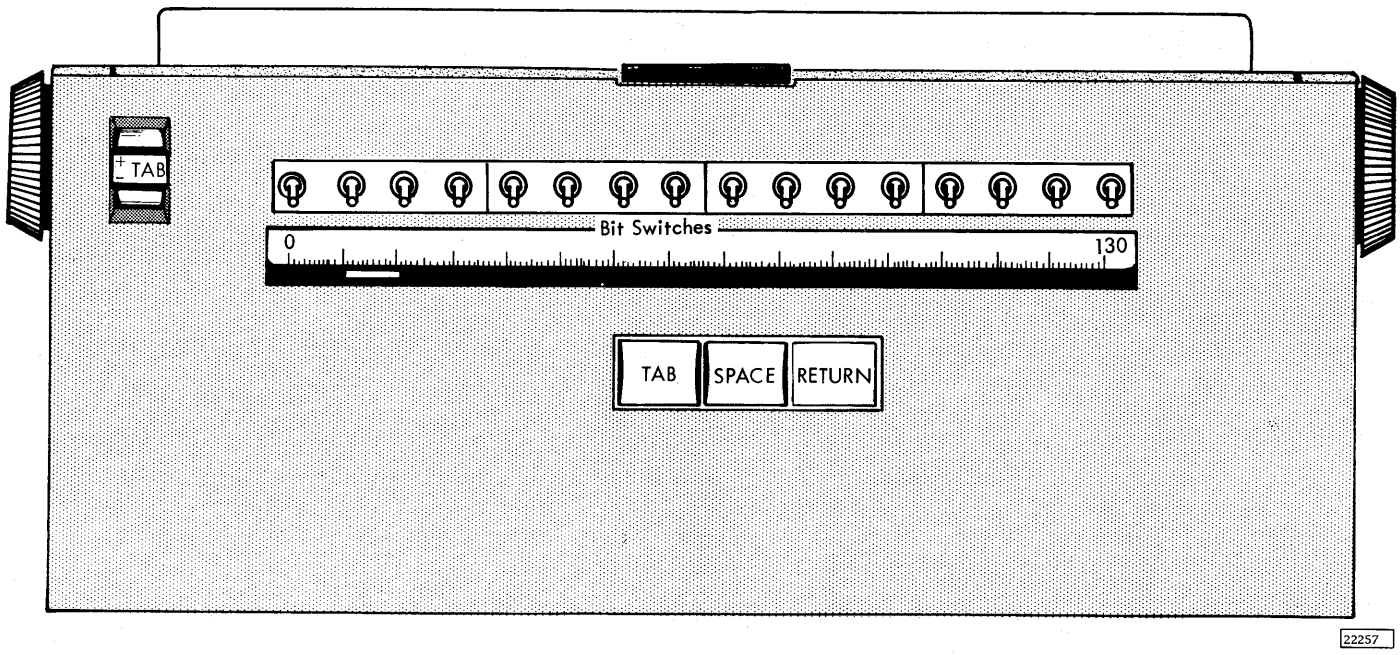



Figure 6-2. Console Printer

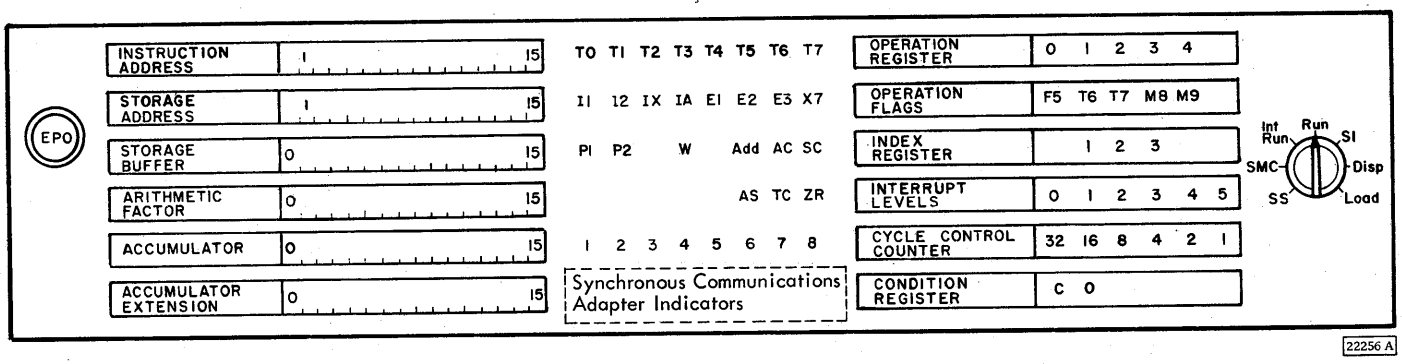



Figure 6-3. Console Display Panel

Arithmetic Factor (D) Register: This row of indicators displays the word in the D-register.

Accumulator (A) Register: This row of indicators displays the word in the A-register.

Accumulator Extension (Q) Register: This row of indicators displays the word in the Q-register.

Operation (Op) Register: These indicators display positions 0-4 of an instruction word. At the end of a T-clock cycle, the lights indicate the operation in process or completed during that cycle.

Operation Flags: These indicators, from left to right, light when the 'format 5,' 'tag 6,' 'tag 7,' 'modifier 8,' and 'modifier 9' FF's are on. Any of them may be turned on during an I1 cycle. 'Modifier 8' is also turned on if 'add to storage interlock' is activated.

Index Register: These indicators light according to the output of the 'tag 6' and 'tag 7' flip-flops, when they are addressing core storage instead of the M-register. 'IX inhibit SAR' must be active at the same time.

Interrupt Levels: These indicators display interrupt levels that are active. (The corresponding 'level' FF's are on.)

Cycle Control Counter: These indicators display the decimal value represented in the flip-flops of the CCC.

Condition Register: One indicator lights when the 'carry' FF is on. The second lights when the 'overflow' FF is on.

T0 through T7: Each indicator displays the last clock step (T-clock time) that was taken. They are useful primarily when the CPU is operating in single step mode. In other modes, T7 is normally the only indicator to glow brightly, doing so when the T-clock is stopped.

Cycle Timer (I and E): Each indicator displays the type of cycle in process when operating in single step mode. Otherwise, they indicate the type of the last cycle taken. All seven indicators except E2 light when the respective flip-flops are on. E2 lights when the 'E2 cycle' line is active.

X7: This indicator lights when the cycle steal X-clock is stopped. (The 'X7' FF is on continuously.)

Parity (P1, P2): Each indicator lights when the corresponding FF is on. There are two ways to turn on each FF. One is by sense amp pulses when reading from core storage. The other is by having even parity in the corresponding half of the B-register when 'storage write cycle' is active and the 'check trigger control' FF is on.

> Note: When the rotary mode switch is set to load and the bit switches are used to set the B register, 'P1' and 'P2' may not be affected to indicate correct parity. However, when a machine cycle writes the contents of the B register into core storage, 'P1' and 'P2' are set for correct parity. Thus, the word is written into storage with correct parity.

Wait (W): This indicator lights when the 'wait op' line is activated by any unassigned op code.

Add: This indicator lights when the 'add' FF is on.

Arithmetic Control (AC): This indicator lights when the 'arithmetic control trigger' FF is on to allow arithmetic action in that cycle.

Shift Control (SC): This indicator lights when the 'shift control trigger' FF is on to allow shifting in that cycle.

Arithmetic Sign (AS): This indicator lights when the 'arithmetic sign' FF is on. The state of this FF depends on the type of cycle, operation being performed, and the signs of the factors or partial results.

Temporary Carry (TC): This indicator lights when the 'temporary carry' FF is on. The FF is turned on by a carry out of A bit 0 position during add or a borrow by A bit 0 position during subtract.

Zero Remainder (ZR): This indicator lights when the 'zero remainder' FF is on. This FF is used in quotient correction during a divide operation.

CE Indicators: These numbered, but otherwise unlabeled, indicators can be wired by a CE to light when selected conditions exist within CPU circuits.

Switches

Emergency Power Off (EPO)

This pull switch drops power to the system without the normal power off sequence. It should only be used, as the name implies, in an emergency.

Mode (Rotary) Switch

This switch selects one of the seven modes in which the CPU may run.

Single Step (SS) Mode: Pressing the program start key causes the T-clock to advance one step (for example from T7 to T0), if there is no parity stop condition. When the key is pressed, 'oscillator trigger' turns on the 'advance' FF, gating the turn-on of 'delay.' The next oscillator trigger pulse turns on 'delay,' which turns on the 'run' FF. 'Single step' prevents 'oscillator trigger' from activating 'phase A.' Instead, 'phase A' is activated by 'delay,' so that only one 'T clock advance SP' occurs for each time the program start key is pressed. When the key is released, 'advance' turns off, turning off 'delay.' Turning off 'advance' also turns off 'run' in single step mode.

Single Machine Cycle (SMC) Mode: Pressing the program start key causes the T-clock to run one cycle. Turning on the 'run' FF causes an advance to T0. Then phase A pulses step the clock until T7 of the clock cycle turns 'run' off, preventing a second advance to T0.

Interrupt Run (INT RUN) Mode: In this mode, a level 5 interrupt occurs after each mainline program instruction is completed, providing a convenient method of "program tracing." When no 'interrupt level' FF is on (CPU is executing a mainline program instruction), T5 turns on the 'program trace' FF. A console ILSW bit 0 is set, as well as a console bit 0 is set, as well as a console DSW bit 1, and 'level 5 interrupt request' is activated. When the 'branch out' from the level 5 servicing subroutine turns off the 'level 5' FF, 'program trace' also turns off. Then as the next mainline program instruction is started, T5 turns on 'program trace' again.

Program Run (RUN) Mode: This is the usual mode for executing programs. When program start is pressed, program execution starts. Operation continues until a 'wait' instruction is given, the immediate stop key is pressed, or the program stop key is pressed.

Single Instruction (SI) Mode: Pressing the program start key causes the reading and execution of one instruction. The 'run' FF is turned on and stays on until a 'T7 count 0' turns it off.

Display Core Storage (DISP) Mode: Pressing the program start key causes a machine cycle. The core storage word at the address in IAR is read, set into the B-register for display, and written back into core storage, unchanged. IAR is incremented +1. The cycle taken is an I1 cycle with many of the normal I1 functions blocked by the switch setting. 'I1' turns on at T0, but the operation register is not set, nor is the D-register set from the B-register. IAR is set into the M-register, as usual in an I1 cycle, and the M-register addresses core storage. Then IAR is incremented as usual. By T2, the operation register has been reset to all 0's, simulating a 'wait.' The 'run' FF turns off at T7, and the display operation is complete.

Load Core Storage (LOAD) Mode: In this mode, turning on a bit switch immediately turns on the corresponding FF in the B-register. Then pressing the program start key causes the word in the B-register to be written into the core storage location

specified by IAR. Pressing the load IAR key sets the word into IAR as previously described.

With the clock at T7, 'load sample gate' is continuously active, and each 'phase A SP B' activates I/O to B sample pulses. 'Load gate' is also active, so the bit switches activate console data bit lines when the switches are on. Thus, the B-register matches the setting of the bit switches.

Pressing the program start key starts a T-clock cycle which is similar to the cycle taken in display mode. (The 'load/display dot' is active for both.) However, in load mode at T3, 'I/O entry sample' is activated. Therefore, the word read from storage is replaced by the word from the bit switches. At write time, the latter word is set into core storage.

## SECTION 2. MAINTENANCE FEATURES

### CE SWITCHES

The CE switches (Figure 6-4) are mounted on a panel below and to the right of the console panel and to the rear of the usage meter.

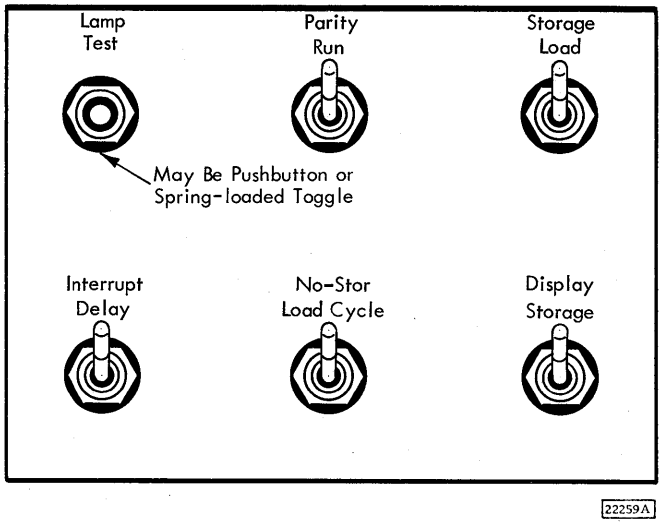



Figure 6-4. CE Switches

Lamp Test: Setting this switch on lights all the display panel indicators to test their operation.

Parity Run: Setting this switch on prevents a parity check from stopping CPU operations. The 'parity check' FF cannot activate 'parity stop,' and 'start advance' turns off the 'parity check' FF. The switch should always be off for customer operations.

Storage Load: Setting this switch on allows the bit switches to control what is written into core. The mode switch determines whether one or all core positions are written. For example, to set all core storage positions to all 0's:

1. Set all bit switches off.
2. Set mode switch to 'run.'
3. Set storage load switch on.
4. Press program start key.

The operation is similar to what occurs when the mode switch is set to 'load' except that multiple machine cycles occur from pressing the program start key once. The 'wait not storage load/display' line is blocked, so the 'run' FF does not turn off at T7.

Interrupt Delay: Setting this switch on blocks the 'gate interrupt' line, preventing any interrupt request from turning on an interrupt level FF.

Non-Storage Load and Cycle: Setting this switch on blocks 'storage use' on any cycle except a CS cycle. Also, phase A pulses set the B-register FF's in agreement with the bit switches when the T-clock is at T7. Depending on the mode switch, cycles or cycle steps can be taken without changing storage. Many CPU functions may be tested with various settings of the mode switch without disturbing the contents of core storage.

Display Storage: When this switch is on, each cycle reads and writes core storage and increments IAR + 1. Thus the contents of increasingly higher addresses in core storage are displayed as the clock runs. The manner of cycling is determined by the mode switch. For example, setting the mode switch to SCM and pressing the program start key, operates the same as having the display storage switch off, and the mode switch set to DISP.

Machine characteristics are given here for the basic 1131 central processing unit and features which are basic to certain models.

## FUNCTIONAL CHARACTERISTICS

Core storage word length: 16 data bits, 2 parity bits
Processing word length: 16 bits
Instruction length: one or two words
Data length: one word (single precision) or two words (double precision)
Core storage capacity: 4,096, 8,192, 16,384, or 32,768 words
Reserved core storage locations:

| | |
|---|---|
| /0001 | Index register 1 |
| /0002 | Index register 2 |
| /0003 | Index register 3 |
| /0008 to /000D | Interrupt address table |
| /0020 to /0027 | Printer scan field (1132 printer only) |

Storage cycle time: 3.6 to 2.2 microseconds
Console printer output: 15.5 characters per second
Single disk storage capacity: 512,000 words, nominal per disk cartridge. Cartridges are interchangeable.
Single disk storage read/write rate: 36,000 words per second.

## PHYSICAL CHARACTERISTICS

| Dimensions | Width | Depth | Height |
|---|---|---|---|
| Inches: | 58.25 | 29 | 46.5 |
| Centimeters: | 148 | 74 | 118 |

| Service Clearance | Front | Rear | Left | Right |
|---|---|---|---|---|
| Inches: | 42 | 36 | - | 30 |
| Centimeters: | 107 | 91 | - | 76 |

Maximum Weight: 760 lb (34,5 kg)
Heat output/hr: 3,100 BTU (781 kcal)
Air flow: 720 $ft^3$/min (21 $m^3$/min) maximum

## ELECTRICAL REQUIREMENTS

Voltage*

| | |
|---|---|
| at 60 Hz: | 115, 208, or 230 |
| at 50 Hz: | 195, 220, or 235 |

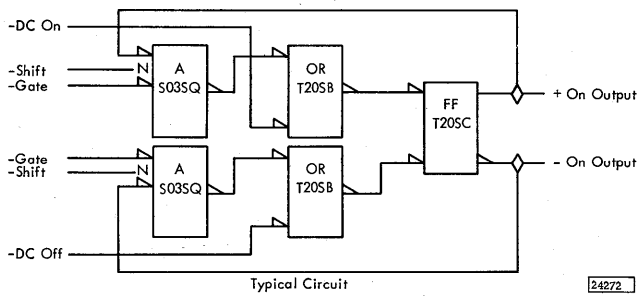
Phase: 1
Power: 1.1 kva

## OPERATING ENVIRONMENT

Temperature: 60° to 90° F (15,6° to 32,2° C)
Relative humidity: 10% to 80%

*Affected by features installed. See Figure 1-3.

## MULTI-INPUT (MI) FLIP-FLOP




The MI trigger is used in clock, counter, and register circuits as a standard flip-flop. Direct-coupled emitter followers are used for the triggering action, and inverters are used for the output functions. The trigger may be connected for binary operation, single-gated ac input, dual-gated ac input, or dc set input. Both in-phase and out-of-phase outputs are provided.

AC Set Input: External RC packs are used for gated input pulse triggering. The set pulse is a 3V negative shift (+3V to 0.3V) with a minimum duration of 30 ns. The gate must be conditioned (at 0.3V) for at least 170 ns before arrival of the set pulse. The gate input remains at an operating level, due to a capacitor charge, for about 90 nanoseconds after the fall of the signal.

DC Set Input: For dc triggering, a down level of 0.3V is applied to the dc set input. The down input signal must be at least 30 ns in duration. For ac set operation, the dc input must be at up (+3V) level.

Binary Operation: The trigger can be adapted for binary operation. The binary input pulse is applied to the two ac inputs. The ac gates may be connected to an external gate or tied to 0V.

It is important to note that should another gated ac set pulse be applied to the on input while the trigger is on, a narrow positive pulse is generated at the off output. Likewise, when the trigger is off, a gated ac set pulse applied to the off input causes a positive pulse at the on output.

For applications where the "reflected" pulses must be suppressed, the circuit is provided with internal gating that requires the trigger to be off before it can be turned on, and vice versa. Circuits with this internal gating can be identified in the ALD's by the output-to-input connections.

A-Register See: Accumulator
AC See: Arithmetic Control Trigger
Accessing, Single Disk Storage
  Access Drive Latch 4-9
  Description 4-8
  IOCC 4-9
  Settle Single-Shot 4-9
Accumulator 2-5
Accumulator Extension 2-6
Add
  Circuit Description 3-14
  Circuits, Functional Unit 2-27
  Data Flow 1-13
  General Description 1-13
  Indicator 6-4
  Operation 2-28
Add Double
  Circuit Description 3-15
  Data Flow 1-14
  General Description 1-13
Add Gate 2-27
Add-to-Storage Operation
  Description 3-35
  Example 3-36
  E1 Cycle 3-36
  E2 Cycle 3-36
  IA Cycle 3-36
  I1 Cycle 3-36
  I2 Cycle 3-36
Add-Subtract Circuits 2-27
Addressing See: Core Storage Addressing
Advance, T-Clock 2-2
AND 3-22
AND, OR, and Exclusive OR
  General Description 1-15
Arithmetic Control Trigger 2-27, 6-5
Arithmetic Factor Register See: D-Register
Arithmetic Sign Indicator 6-5
AS See: Arithmetic Sign Indicator

B-Register 2-5, 6-3
Backspace 3-43
Backspace Key 6-1
Bail Contacts 2-38
Binary Operation of Flip-Flops B-1
Bit Counter 4-6
Bit Switches, Console 3-42
BOSC 3-34
Branch and Store IAR
  Circuit Description 3-31
  Example 1-16
  General Description 1-16
  Long Format 1-16
  Operation When F = 0 3-31
  Operation When F = 1 3-32
  Short Format 1-16
  Test Conditions 1-16, 3-31
Branch Conditions 1-16, 3-31
Branch or Skip on Condition
  Circuit Description 3-32
  Examples 1-17
  General Description 1-16
  'Skip Condition' FF Off 3-33
  'Skip Condition' FF On 3-33
Branch Out or Skip on Condition
  Branch Out or Skip on Condition (BOSC) 3-34
  General Description 1-18
Buffer Register 2-5

CCC
  Decrement Sample 2-5
  Decrementing from 18 2-6
  Functions 2-4
  Indicators 6-4
Central Processing Unit
  Adapter Circuits 1-1
  Console 1-3, 6-1
  Console Bit Switches 1-3, 3-42
  Control Logic 1-1
  Core Storage 1-1
Character Code
  Keyboard 3-44
  Printer, Console 3-46
Clearing Core Storage 6-7
Condition Register 6-4
Connections 1-1
Console
  Components 6-1
  Display Panel 6-3
  Indicators 6-4
  Mode Switch 6-5
  Switches 6-5
Console Bit Switches
  Console Mode 3-42
  Input 3-42
  Read 3-42
Console Keyboard
  Data Transfer 3-42
  Function Keys 6-1
  Layout 6-1
  Mechanism 2-37
Console Printer
  Character Format 3-46
  Control Characters 3-47
  Data Character Coding 3-46
  Mechanical Controls 6-3
  Other Manuals 2-40
  Output Operation 3-46
  Print Cycles 3-48
  Response 3-46
  Write 3-47
Contact Bails 2-38
Core Storage
  Arrays 2-12

Core Storage (Continued)
Current Flow 2-20, 2-21
Data Flow and Control 2-11
Module Selection 2-12, 2-16
Reading Out 2-10
Writing Into 2-9
Core Storage Addressing
Decode 2-13
Schematic, Simplified 2-14
4k Array 2-10
8k Array 2-10
Core Storage Arrays
Cycle Times 2-12
Differences Between 2.2 and 3.6 Arrays 2-12
Physical Layouts 2-12
Core Storage Word
Bit Positions 1-10
Capacity 1-11
Decimal Values 1-11
Parity 1-11
CPU Cycles 1-4
CPU Registers 1-4
Current Flow
2.2 Microsecond Storage 2-21
3.6 Microsecond Storage 2-20
Cycle Control Counter See: CCC
Cycle Steal
Circuits 2-30
Controls 2-31
General Description 1-20
Request 2-30
Cycle Steal Circuits 2-30
Cycle Steal Principle
Reason for Cycle Steal 1-9
X-Clock 1-9, 2-30
Cycle Steal Priority 1-9
Cycle Timer
Control 2-3
Flip-Flops 2-3
Indicators 6-4
Cycles
Flow Chart, Cycle Timer 2-4
Machine Cycles 1-4
Sequence, Typical 1-10
Cylinder Concept 4-2

D-Register 2-5
Data Flow
CPU 1-4, 1-5
System 1-4, 1-5
Data Format 1-12
Data Table 1-12
Data Transfer
Cycle Steal 1-20
Direct Program Control 1-20
Methods 1-19
Decimal Equivalents of Binary Bits 1-19
Double Word 1-12
Single Word 1-12
Decrement Sample 2-5
Description, System 1-1
Device Status Word See: DSW
Direct Program Control 1-20
Disk Storage Timing
Access 4-3
CPU Time 4-3
Read/Write 4-3
Disk Storage See: Single Disk Storage
DISP Mode 6-5
Display Core Storage (DISP) Mode 6-5
Display Panel 6-3
Display Storage 6-7
Divide by Zero 3-20
Divide
Circuit Description 3-20
Data Flow 1-15
Examples 3-21
E1 Overflow Checks 3-20
E2 Overflow Checks 3-20
General Description 1-15
Double Precision Carry 3-15
Drive Current
Current Control Circuit 2-20
2.2 Microsecond Storage 2-21
3.6 Microsecond Storage 2-20
DSW
Console Printer and Keyboard 3-47
Disk Storage 4-8
General Description 1-8

E Cycle Times 2-3
E-Cycles
Flip-Flops 3-6
Load Accumulator E1 Cycle 3-6
EA Generation 3-1
Effective Address Generation 3-1
Electrical Functions, Keyboard 2-40
Electrical Requirements A-1
Emergency Power Off 6-5
End-of-Field 3-43, 6-1
EOR 3-23
EPO 6-5
Erase-Field 3-43, 6-1
Exclusive OR (EOR) 3-23
Execute I/O See: XIO
Execution Cycles
Circuit Description 3-6
Flow Chart 2-4
General Description 1-6
Extended T7 2-3
E1 Cycle, Load Accumulator 3-6
E1, E2, E3 See: Cycle Timer, CPU Cycles

Features
Basic 1-3, 4-1
Optional 1-3
File Check Counter
Circuit Description 4-5
Read 4-4
Write 4-2
File Core Address Register 4-5
File Data Register
Description 4-4

File Data Register (Continued)
Operation During Read 4-5
Operation During Write 4-5
Flip-Flop, Multi-Input 2-1, B-1
Format Register 2-5
Forced BSI 2-31
Function Codes 1-19
Functional Characteristics
Cycle Time A-1
Instruction Length A-1
Word Length A-1

Hysteresis Curve 2-8

I-Register
Description 2-7
Incrementing 2-8
I/O Instruction 3-39
I/O Units 1-1
IA Cycles
Circuit Description 3-5
Description 1-6
Indicator 6-4
IAR
Increment 2-7
Indicators 6-3
ILSW
Level 2 4-7
Level 4 3-45
Index Register
Operation during IX Cycles 3-5
Selection by Tag 1-11
Indexing 1-6
Indexing Cycle See: IX Cycles
Indicators See: Lights
Indirect Address 3-5
Indirect Addressing 1-6
Indirect Addressing Cycle See: IA Cycles
Inhibit/Sense Lines
Illustration 2-15
2.2 Microsecond Storage 2-22
3.6 Microsecond Storage 2-22
Input from Keyboard 3-43
Input/Output 1-19
Input/Output Control Command See: IOCC
Input/Output Instruction 3-39
Instruction Address Register 2-7, 6-3
Instruction Cycle 1 See: I1 Cycles
Instruction Cycle 2 See: I2 Cycles
Instruction Cycles
Chart 3-1
Circuit Description 3-1
Flow Chart 2-4
Indexing 1-6
Indirect Addressing 1-6
I1 1-6
I2 1-6
Overlap 3-2
Instruction Format
One-Word 1-11
Two-Word 1-11
Instruction Groups
Arithmetic 3-14
Branch or Skip 3-31
Load and Store 3-8
Shift Left 3-24
Shift Right 3-28
Instruction Set 1-12
INT RUN Mode 6-5
Interlocks, Keyboard
Description 2-40
Illustration 2-42
Interrupt Circuits
Forced BSI Indirect 2-32
Forced Instruction, Format 2-32
Objectives 2-30
Priority 2-32
Requests 2-32
Two Primary Functions 2-31
Interrupt Delay 6-7
Interrupt Forced BSI
General Description 1-17
Circuit Description 2-31
Interrupt Level Status Word See also: ILSW
Description 1-8
Interrupt Levels
Description 1-8
Indicators 6-4
Priorities 1-9
Interrupt Principle
Causes 1-7
Data Transfer 1-7
Device Status Word 1-8
Example 1-7
Interrupt Level Status Word 1-8
Interrupt Levels and Priority 1-8
Interrupt Servicing Subroutine 1-8
Objective 1-6
Review of 1-8
Status Transfer 1-7
Interrupt Priority 2-32
Interrupt Request Key 6-1
Interrupt Run (INT RUN) Mode 6-5
Interrupt Sequences 2-36
Interrupt Servicing Subroutine 1-8
Introduction 1-1
IOCC
Console Printer Write 3-47
Format 3-39
Function Codes 1-19
General Description 1-19
Sense Interrupt 3-45
IX Cycles
Circuit Description 3-4
Indicator 6-4
Purpose of Indexing 1-6
I1 Cycles
Circuit Description 3-1
General Description 1-6
Flow Charts 3-3, 3-4
Indicator 6-4
Setting Control Registers 3-2

I2 Cycle
Circuit Description 3-4
General Description 1-6
Indicator 6-4
Keyboard
Character Code 3-44
Electrical Functions 2-40
Input 3-43
Interlocks 2-40
Mechanics 2-37
Read 3-45
Response 3-43
Select 3-43
Sense Interrupt 3-43
Units 2-37
Keyboard Input 3-43
Keyboard Restore
Description 2-39
Illustration 2-42
Latch Latched 2-40
Latch Unlatched 2-40
Restoring Bail Contact 2-40
Keyboard Select 3-43
Keys
Backspace 6-1
End of Field 6-1
Erase Field 6-1
Interrupt Request 6-1
Mechanical Operation 2-38
Numeric 2-37, 6-2
Restore Keyboard 2-37, 6-1
Keystem Numbering
Chart 2-41
Description 2-39

Lamp Test 6-7
Lamps See: Lights
Latch Contacts 2-38
Latch Pull Bar 2-38
Lights
Console 6-3
Keyboard 6-2
Load
Accumulator (LD) 3-8
Double (LDD) 3-9
General Description 1-13
Index (LDX) 3-11
Status (LDS) 3-12
Load Accumulator
D- to A-Register Transfer 3-8
E1 Cycle 3-6
Load Core Storage (LOAD) Mode 6-5
Load Index (LDX)
Tag = 00 3-11
Tag ≠ 00 3-12
LOAD Mode 6-5
Long Time
2.2 Microsecond 2-27
3.6 Microsecond 2-23

M-Register See: Storage Address Register
Machine Characteristics
Electrical Requirements A-1
Functional Characteristics A-1
Operating Environment A-1
Physical Characteristics A-1
Machine Cycle 2-2
Machine Language 1-10
Magnetic Core Theory 2-7
Maintenance Features 6-1
MDX 3-34
Mid-Pac
Power Off Sequence 5-4
Power On Sequence 5-4
Midpack See: Mid-Pac
Mode Switch 6-5
Model Variations 1-1, 1-3
Modifier Register 2-5
Modify Index and Skip
Functions 1-18
General Description 1-18
Long Format MDX, Tag = 00 3-35
Long Format MDX Tag ≠ 00 3-37
Short Format MDX, Tag = 00 3-34
Short Format MDX, Tag ≠ 00 3-35
Modulo 4 4-4
MPS
Power Off Sequence 5-4
Power On Sequence 5-2
Multi-Input Flip-Flop
AC Set Input B-1
Binary Operation B-1
DC Set Input B-1
Reference 2-1
Multiple Interrupt Requests
Higher Level Interrupting Lower Level 2-37
Same Time 2-35
Sequence 1 2-33
Sequence 2 2-34
Sequence 3 2-35
Multiply
Algorithm 3-17
Circuit Description 3-16
Cycles 3-18
Data Flow 1-14
Examples of 3-19
E1 Cycles 3-18
E2 Cycles 3-19
General Description 1-14

Negative Numbers 1-12
Non-Storage Load and Cycle 6-7
Numeric Key 6-2

One-Word Instruction 1-11
OP Register See: Operation Register
Operating Environment A-1
Operation Flags See: Format, Tag, Modifier Registers
Operation Register 2-5
Operations, General Description
Arithmetic 1-13
Branch or Skip 1-16

Operations, General Description (Continued)
Input/Output 1-19
Load and Store 1-13
Shift Left 1-15
Shift Right 1-16
Wait 1-19
OR 3-22
Oscillator 2-1
Overflow Checks in Divide
End Operation 3-22
E1 3-20
E2 3-20
Overflow FF
Add or Subtract Operations 3-14
During Divide 3-20

Parity
Description 1-11
Indicators P1 and P2 6-4
Parity Run 6-7
Permutation Bar 2-38
Physical Characteristics
Dimensions A-1
Service Clearance A-1
Power Off Sequence 5-4
Power On Reset 5-4
Power on Sequence
Mid-Pac 5-4
MPS 5-2
Power Sequencing
Mid-Pac 5-4
MPS 5-2
Power Supplies
Input 5-1
Outputs 5-1
Sequencing 5-1
Primary Power
Effect of Features 5-1
Effect of Model Number 5-1
World Trade Corporation 5-1
Printer, Console 2-40
Priority See: Cycle Steal Priority, Interrupt Priority
Program Execution
Description 1-9
Typical Cycle Sequence 1-10
Program Run (RUN) Mode 6-5
P1, P2 See: Parity

Q-Register See: Accumulator Extension

Read Drivers See: Drive Current
Read Gates See: Drive Current
Read-Check Operation 4-13
Reading Sync Word 4-12
Registers, CPU
Data Flow Between 1-4
Request Interrupt
Levels 2-33
Multiple Requests 2-33
Reset 2-33
Reset D Reg SP 2-28
Reset, Power On 5-4
Restore Keyboard Key 6-1
Rotate Right (RTE) 3-29
RUN Mode 6-5

SAR See: Storage Address Register
SC Indicator 6-5
Sector Comparison 4-10
Sector Counter
Description 4-6
Operation 4-7
Sector Format 4-2
Sector Numbering 4-2
Sector Register 4-6
Sense/Inhibit Lines
Illustration 2-15
2.2 Microsecond Storage 2-22
3.6 Microsecond Storage 2-22
Servicing Subroutines 2-33
Settle Single-Shot 4-9
Shift Count 3-24
Shift Left
Accumulator 3-24
Accumulator and Extension 3-25
General Description 1-15
Theory of Operation 3-24
Shift Left Accumulator and Extension (SLT) 3-25
Shift Left Accumulator (SLA) 3-24
Shift Left and Count
Accumulator and Extension (SLC) 3-26
Accumulator (SLCA) 3-25
Description 1-15
Shift Right
Accumulator and Extension (SRT) 3-29
Accumulator (SRA) 3-28
General Description 1-16
Shift Right Rotate 16/48 3-30
Short Time
2.2 Microsecond 2-27
3.6 Microsecond 2-23
SI Mode 6-5
Single Disk Storage
Access Mechanism 4-1
Accessing 4-8
Adapter 4-4
Cartridge 4-1
Cylinder 4-1
Data Checking 4-3
Data Word 4-3
Disk Capacity 4-2
Disk Organization 4-2
Disk Storage Drive 0 4-1
Initiate Write 4-9
Sectors 4-1
XIO Initiate Read 4-11
XIO Initiate Write 4-10
Single Instruction (SI) Mode 6-5
Single Machine Cycle (SMC) Mode 6-5
Single Step Mode 6-5
Skip 1-18
SMC Mode 6-5

Special Circuits B-1
SS Mode 6-5
Storage Address Register 2-7, 6-3
Storage Buffer Register See: B-Register
Storage Clock
2.2 Microsecond 2-26
3.6 Microsecond 2-23
Storage Index (STX)
Tag = 00 3-12
Tag ≠ 00 3-12
Storage Load 6-7
Store
Accumulator (STO) 3-10
Double (STD) 3-10
General Description 1-13
Index (STX) 3-12
Status (STS) 3-13
Subtract
Circuit Description 3-16
Circuits 2-29
Data Flow 1-13
Double 3-16
General Description 1-13
Operation 2-29
Subtract Double (SD)
Circuit Description 3-16
Data Flow 1-14
General Description 1-13
Subtract Gate 2-29
Switches
CE 6-7
Console 6-5
Display Storage 6-7
Interrupt Delay 6-7
Keyboard 6-2
Lamp Test 6-7
Non-Storage Load and Cycle 6-7
Parity Run 6-7
Storage Load 6-7
Sync Word
Reading 4-12
Writing 4-10
System Configuration 1-2
System Description 1-1

T-Clock
Clock Advance Pulses 2-2
Control 2-1
Purpose 1-6
Single Step Mode 2-3
Stop 2-3
T-Clock Advance SP 2-2
Tag
Index Register Code 1-11
Register, Functional Unit 2-5
TC See: Temporary Carry
Temporary Accumulator 2-6
Temporary Carry
In Add Double 3-15
Indicator 6-5
Test Conditions 1-16, 3-31
Timings
Basic 1-6
T-Clock 2-1
Two-Word Instruction 1-11
Typewriter, See: Console Printer

U-Register 2-6

Voltages
Input 5-1
Outputs 5-1
World Trade Corporation 5-1

W See: Wait
Wait
Circuit Description 3-37
General Description 1-19
Indicator 6-4
Word Count Register
Incrementing 4-4
Loading 4-4
Write Drivers See: Drive Current
Write Gate See: Drive Current
Writing Sync Word 4-10

X-Clock
Operation 2-28
Purpose 1-19
XIO
Decoding Functions 3-39
Function Codes 3-39
Reading IOCC 3-39
XIO Control 3-40
XIO Initiate Read, Disk Storage
Data Transfer 4-12
E2 Cycle 4-12
First CS Cycle 4-12
Objectives 4-11
Operation Complete 4-12
Reading Data 4-12
Reading the Sync Word 4-12
Sector Comparison 4-12
XIO Initiate Read, General 3-41
XIO Initiate Write, Disk Storage
Data Transfer CS Cycles 4-10
E2 Cycle 4-10
First CS Cycle 4-10
Objectives 4-9
Operation Complete 4-11
Sector Comparison 4-10
Writing Data 4-11
Writing Sync Word 4-10
Zeros at End of Sector 4-11
XIO Initiate Write, General 3-41
XIO Read
Area 7 3-42
General 3-41

XIO Sense DSW
Area 7 3-42
General 3-40
XIO Sense ILSW 3-40
XIO Write 3-41

Zero Remainder (ZR) Indicator 6-5

2.2 Microsecond Storage
Clock 2-26
Current Flow 2-21
Drive Current 2-21
Inhibit Driver 2-22
Inhibit/Sense 2-22
Long Time 2-27
Sense Amplifier 2-22
Short Time 2-27
Timing 2-23
3.6 Microsecond Storage
Clock 2-23
Current Flow 2-20
Drive Current 2-20
3.6 Microsecond Storage (Continued)
Inhibit Driver 2-22
Inhibit/Sense 2-22
Long Time 2-23
Sense Amplifier 2-22
Short Time 2-23
Storage Clock 2-24
Time Delay Circuit 2-23
Timing 2-23
4k Array
Array Windings 2-20
Connections 2-19
Construction 2-19
Diode Board Terminals 2-20
Select Lines for Location/0000 2-18
8k Array
Array Windings 2-19
Bottom Board Terminals 2-19
Connections 2-19
Construction 2-14
Diode Board Terminals 2-19
Select Lines for Location/0000 2-17

## READER'S COMMENT FORM

1130 FETO SY26-5978-4

- How did you use this publication?

  As a reference source ☐
  As a classroom text ☐
  As..... ☐

- Based on your own experience, rate this publication...

| | | | | | |
|---|---|---|---|---|---|
| As a reference source: | ..... Very Good | ..... Good | ..... Fair | ..... Poor | ..... Very Poor |
| As a text: | ..... Very Good | ..... Good | ..... Fair | ..... Poor | ..... Very Poor |

- What is your occupation? .....

- We would appreciate your other comments; please give specific page and line references where appropriate. If you wish a reply, be sure to include your name and address.

- Thank you for your cooperation. No postage necessary if mailed in the U.S.A.

## YOUR COMMENTS PLEASE . . .

Your answers to the questions on the back of this form, together with your comments, will help us produce better publications for your use. Each reply will be carefully reviewed by the persons responsible for writing and publishing this material. All comments and suggestions become the property of IBM.

Please note: Requests for copies of publications and for assistance in utilizing your IBM system should be directed to your IBM representative or to the IBM sales office serving your locality.

fold fold

FIRST CLASS
PERMIT NO. 2078
SAN JOSE, CALIF.

**BUSINESS REPLY MAIL**
NO POSTAGE STAMP NECESSARY IF MAILED IN U. S. A.

POSTAGE WILL BE PAID BY . . .

IBM Corporation
Monterey & Cottle Rds.
San Jose, California
95114

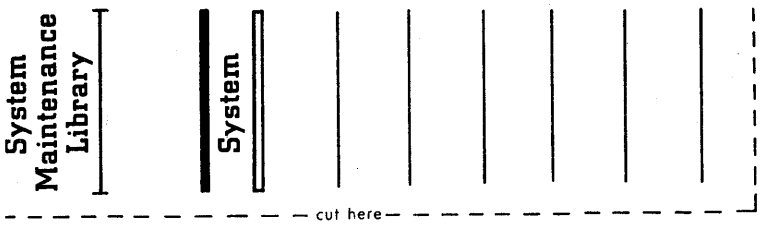
Attention: Product Publications, Dept. 455

fold fold

IBM

**International Business Machines Corporation**
**Data Processing Division**
**1133 Westchester Avenue, White Plains, New York 10604**
**[U.S.A. only]**

**IBM World Trade Corporation**
**821 United Nations Plaza, New York, New York 10017**
**[International]**




SY26-5978-4

IBM 1130 Printed in U.S.A. SY26-5978-4

IBM

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
[U.S.A. only]

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
[International]