IBM®

Field Engineering

Theory of Operation

Restricted Distribution

Specifications contained herein are subject to change without notice. Revisions and/or supplements to this publication will be issued periodically.

System/360 Model 50

Multiplexor Channel

PREFACE

This manual describes input/output (I/O) operations on the IBM System/360 Model 50 Multiplexor Channel. To understand channel theory as presented, the reader should first study other System/360 Model 50 Manuals that explain parts of the 2050 Processing Unit.

Field Engineering Manuals on the 2050 Processing Unit for System/360 Model 50 include:

Comprehensive Introduction, Form Y22-2821
Read Only Storage, Form Y22-2823
Op Codes RR and RX, Form Y22-2824
Op Codes, RS, SI, and SS, Form Y22-2825
Main Storage, Local Storage, Storage Protect, Form Y22-2828

Third Edition
This manual obsoletes Form Z22-2827-1. Changes to the text of the earlier publication are indicated by a vertical line to the left of the change; revised or new illustrations are denoted by the symbol (•) to the left of the caption. "Multiplexor Channel ROS Routines," which begins on page 44, is extensively revised and should be reviewed in its entirety.

Significant changes or additions to the specifications contained in this publication will be reported in subsequent revisions or FE Supplements.

This manual has been prepared by the IBM Systems Development Division, Product Publications, Dept. B96, PO Box 390, Poughkeepsie, New York 12602. A form is provided at the back of this publication for reader's comments. If the form has been removed, comments may be sent to the above address.

CONTENTS

ILLUSTRATIONS

- Channels allow CPU and I/O operations to proceed concurrently.
- Channels assemble or disassemble data and control I/O operations.

Channels control input/output (I/O) operations on IBM System/360 Model 50 by directing all signal exchanges between input/output (I/O) units and the IBM 2050 Processing Unit. Channels permit concurrent data processing within the 2050 and I/O operations.

An instruction in the customer program initiates the I/O operation. Before signaling the channel to begin the operation, the program must form control words in fixed main storage locations. In executing the operation, the channel adheres to the rules that control data establish.

After the I/O operation begins, the channel assembles or disassembles data and synchronizes the transfer of data bytes to the I/O device with main storage cycles. To accomplish this, the channel maintains and updates an address and a byte count. The address locates the destination or source of data in main storage; the count specifies the number of bytes to be transferred. When an I/O device transmits signals to be brought to the attention of the program, the channel converts the signals to a format compatible to that used in the central processing unit (CPU).

MODES OF OPERATION

- In burst mode, one I/O unit reserves all channel facilities.
- In byte mode, several I/O units may share channel facilities.

In transferring data between main storage and I/O units the channel functions in one of two modes: burst mode or byte mode.

In burst mode, one I/O device monopolizes all channel facilities and is logically connected to the I/O interface for the transfer of a burst of data. The burst can consist of:

1. Several bytes of data
2. A block of data (a block is successive main storage locations)
3. A block, or blocks, of data with the associated control sequences to begin the operation and transfer the status byte.

An I/O unit forces burst mode operation by holding the operational in interface line active. An I/O unit performing an operation in burst mode remains logically connected to the interface after the sequence to transfer a byte of data is complete. Even when no interface sequence is in process, a burst mode operation blocks communication between the channel and any other I/O unit.

In byte mode, several I/O units may share channel facilities. One byte mode operation occupies channel facilities only for the time required to complete the sequence to:

1. Initiate the operation
2. Transfer an address byte and one or several data bytes
3. Transfer an address byte, then a status byte.

Although the channel can communicate with only one I/O unit at a time, the channel services concurrent byte mode operations serially. To illustrate this action, assume that I/O units 1, 2, and 3 are performing byte mode operations and that the channel is committed to the I/O unit 1 operation. Neither I/O unit 2 nor 3 can interrupt the channel sequence to service unit 1; therefore, units 2 and 3 must wait until the sequence is complete. When I/O unit 1 drops operational in, freeing the channel, the next unit requesting services establishes communications with the channel.

TYPES OF CHANNELS

- Channel facilities required for sustaining a single I/O operation are called a subchannel.
- The selector channel has one subchannel and can operate only in burst mode.
- The multiplexor channel contains multiple subchannels and can operate in either burst or byte mode.

An IBM System/360 Model 50 can be equipped with two types of channels: selector and multiplexor. A channel is classified according to the mode of operation that it can sustain.

The selector channel has one subchannel; therefore, it can operate only in burst mode. The burst always extends from the initial selection sequence to the end of the operation. The selector channel cannot control concurrent data-transferring operations. While the selector channel is committed to one operation, other I/O devices cannot interrupt the operation to establish communications with the channel.

The multiplexor channel contains multiple subchannels and can operate in either byte mode or burst mode. Because the multiplexor channel can switch between the two modes at any time, an operation on any subchannel can occur partially in byte mode and partially in burst mode.

In byte mode, the multiplexor channel can concurrently sustain one I/O operation per subchannel (if the total load on the channel does not exceed its

capacity). When the multiplexor channel operates in burst mode, the subchannel associated with the burst operation reserves all channel facilities; the multiplexor channel cannot respond to other I/O units until the burst is complete, and "operational in" falls.

CHANNEL/SYSTEM RELATION

- Common channel allows one channel at a time to use CPU facilities for the I/O operation.
- The I/O interface attaches the channel to I/O units.
- The channel can perform limited actions without microprogram control.
- While servicing the channel, ROS operates in I/O mode.
- The series of microinstructions that specify actions to service the channel are called I/O routines.
- ROS contains twelve selector channel routines and thirty multiplexor channel routines.
- The four I/O instructions are: test channel, halt I/O, test I/O, and start I/O.

Three selector channels and one multiplexor channel can be attached to an IBM System/360 Model 50. All Model 50 channels are physically part of the IBM 2050 Processing Unit (Figure 1).

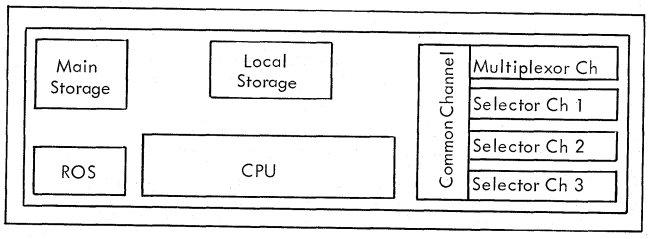


FIGURE 1. IBM 2050 PROCESSING UNIT

Common Channel

Although channels have exclusive circuits and data flow paths to control I/O operations, all channels use CPU facilities at selected points in the execution I/O operations. Only one channel at a time can use CPU circuits; the common channel directs traffic between the channels and CPU facilities. Logic functions in the common channel can be categorized as selection or switching type actions. Common channel processes all selector and multiplexor channel requests for service.

In initiating an I/O operation, the CPU signals the common channel. Common channel, in turn, processes the signal to the designated channel.

I/O Interface

The I/O interface provides a uniform method of attaching I/O control units to a channel (Figure 2). The interface consists of 34 lines and can accommodate up to eight control units and 256 directly addressable I/O devices. A system equipped with four channels has four I/O interfaces, each operating independently.

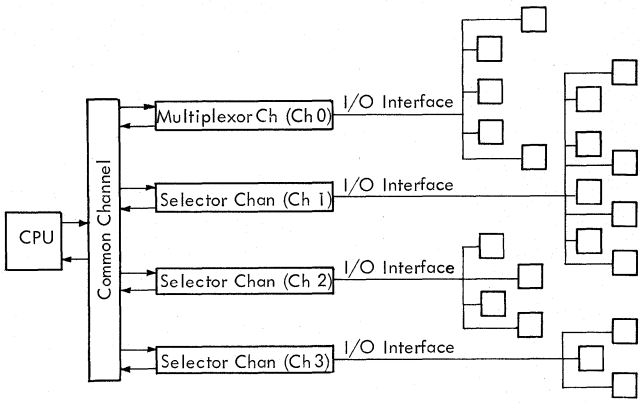


FIGURE 2. CHANNEL RELATION TO I/O INTERFACE AND TO COMMON CHANNEL

Read Only Storage and Central Processing Unit

Logic operations and data transfers in all parts of the 2050 except the selector channel are under complete control of read only storage (ROS) microinstructions. Although the selector channel can perform limited actions without direction from a microprogram, even the selector channel cannot execute any I/O operation independent of read only storage (ROS).

ROS contains 2,816 microinstructions; each microinstruction specifies processing unit actions for one machine cycle.

When CPU and channel are performing different operations concurrently, ROS directs CPU actions while the channel communicates with the I/O interface. When the channel requires service from another part of the 2050 Processing Unit to transfer control information or data, the channel transmits a request signal to the common channel. The common channel causes ROS to break from the CPU operation, execute a series of microinstructions to

service a channel, and then return to the CPU operation. While executing the microprogram to service the channel, ROS operates in I/O mode and uses CPU components to transfer information to or from the channel. When the I/O operation uses CPU facilities, data associated with the CPU operation are preserved.

Switching between CPU mode and I/O mode is called "break in." The break in operation consists of a break in cycle to change ROS from CPU mode to I/O mode, and a break out cycle to switch ROS from I/O mode to CPU mode (Figure 3).

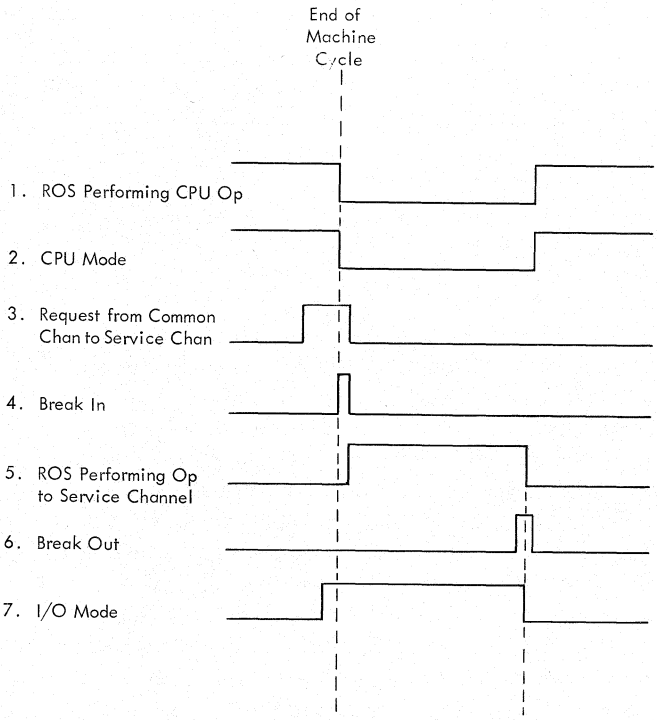


FIGURE 3. BREAK IN AND BREAK OUT CYCLES

I/O Routines

The series of microinstructions that specify actions necessary to service the channel are called I/O routines. I/O routines are arbitrarily limited to six machine cycles to prevent overruns on waiting channels; one channel cannot monopolize the system.

When a channel transmits the request for service to the common channel, the channel must also generate bits of a ROS address to locate the first microinstruction in the required routine. The first microinstruction in each routine has a fixed address. To specify a routine, either the channel or common channel need supply only necessary bits to form the fixed address of the first microinstruction in the routine. Each microinstruction specifies the address of the subsequent microinstruction. When ROS executes the last microinstruction in a routine, it either services another request from the same or other channels, or performs a break out cycle to return to the CPU microprogram if no channel requires service at that time.

ROS contains 12 routines for the selector channel and 30 routines for the multiplexor channel.

I/O INSTRUCTIONS

- Multiplexor channel executes four instructions.
- Start I/O and test I/O instructions are executed with microprogram routines.
- Halt I/O instruction is executed with microprograms when the channel is in byte mode or with logic circuits when the channel is in burst mode.
- Test channel is executed with logic circuits.

IBM System/360 Model 50 recognizes four I/O instructions in the customer program: test channel, halt I/O, test I/O, and start I/O. The multiplexor channel executes these instructions (Figure 4).

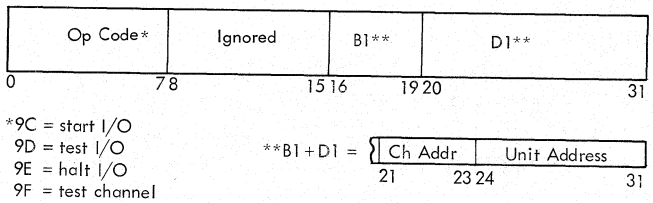


FIGURE 4. I/O INSTRUCTIONS

The test channel instruction allows CPU to test the state of the addressed channel; execution does not disturb the current channel operation. The channel does not require I/O microprogram routines to complete the test channel operation.

The halt I/O instruction causes the channel to terminate the current operation on the addressed I/O unit, if the unit requires further service from the channel to proceed to the normal end. For example, the halt I/O instruction will not stop a rewind operation on a tape unit, but the instruction will terminate data transfers across the I/O interface and release the channel.

The test I/O instruction allows CPU to determine the status of the channel and the address I/O unit. If the designated unit can respond to the test I/O instruction, it transmits status information to the channel.

The start I/O instruction causes the channel to initiate a read, read backward, write, sense, or control operation on the specified unit.

In addition to executing four I/O instructions, the multiplexor channel detects certain I/O end conditions and requests an I/O interrupt. Also, multiplexor channel, in response to a CPU proceed with interrupt signal, executes the interrupt test I/O microprogram routine. Interrupt test I/O microprogram routine is similar to the test I/O instruction.

CHANNEL WORDS

- Channel address word is fetched by the start I/O instruction.
- Channel command word is fetched by the channel address word.
- Channel command word designates the type of I/O instruction.
- Channel status word stores end conditions that exist when an I/O operation terminates.
- Unit control word is not program addressable.
- Unit control word contains information that enables the multiplexor channel to sustain I/O operations.

The start I/O instruction indicates only that the specified channel is to execute an operation on the selected I/O unit. The instruction does not identify a specific operation, nor does it provide the control information that the channel requires. Channel words supplement the start I/O instruction; these words contain control data necessary to execute an operation.

Channel Address Word

The start I/O instruction causes the microprogram unconditionally to fetch the channel address word (CAW) from location 72 in main storage (Figure 5). Instructions preceding the start I/O instruction in the customer program must cause the CPU to store the channel address word in this predetermined location. Bit positions 8-31 designate the main storage location that contains the channel command word.

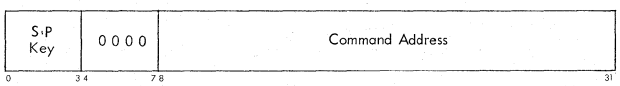


FIGURE 5. CHANNEL ADDRESS WORD (CAW)

Channel Command Word

The 64-bit channel command word (CCW) specifies one of six operations, and governs execution of the operation. (Figure 6.) The operations are: sense, read backward, write, read, control, and transfer-in-channel (TIC). The channel command word that the channel address word specifies should not designate a TIC command.

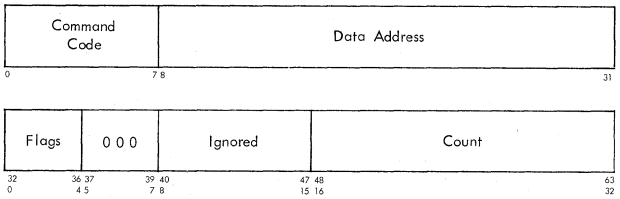


FIGURE 6. CHANNEL COMMAND WORD (CCW)

The channel must obtain the channel command word (CCW-1) before starting a data transferring operation. For example, assume that CPU has decoded the start I/O instruction. The CPU microprogram fetches the first half of the CCW and stores these 32 bits in a CPU register. If the designated channel is free, the channel requests an I/O routine to transfer the first half of the CCW to the channel facilities. The channel requests another I/O routine to transfer the second half of the CCW before the routine processing the first half is complete. If more than one CCW is associated with an operation, they are located in adjacent main storage locations.

Fields in the CCW are allocated for the following purposes:

Command Code: Bits in Position 0-7 specify the operation to be performed.

Data Address: In read, read backward, and sense operations, the data address designates the main storage location to which the first word is stored. In write and control (for some I/O units) operations, the data address specifies the main storage location from which the first word from the I/O unit is taken.

Chain Data (CD) Flag: A 1 in position 32 indicates that another channel command word is associated with the current operation. The data address in the next CCW specifies another storage area into

which more words are to be loaded (input operations) or from which more words are to be taken (output operations).

Chain Command (CC) Flag: The channel holds the I/O unit selected to perform a new command when the present operation proceeds to the normal end if:

1. Position 33 contains a 1, and
2. Position 32 contains a 0.

The command code in the next CCW specifies the succeeding operation.

Suppress Incorrect Length Indication (SILI): When the channel detects an incorrect length record, it does not indicate the condition to CPU if position 34 contains a 1, and position 32 in the last channel command word associated with the operation contains a 0.

Skip Flag: A 1 in position 35 suppresses the transfer of data to storage during a read, read backward, or sense operation.

Program Controlled Interruption (PCI) Flag: A 1 in position 36 causes the channel to generate an interrupt condition to present status information to the CPU. The interruption occurs as soon as the CPU will permit after fetching the channel command word containing the program-controlled interruption flag. The interruption does not disturb the current I/O operation.

Positions 37, 38, and 39 of every channel command word other than one specifying transfer-in-channel must contain zeros. The channel ignores the contents of positions 40-47 in every CCW.

Count: Positions 48-63 indicate the number of eight-bit byte locations reserved in the storage area that the data address specifies.

Channel Status Word

The channel status word (CSW) provides to the customer program the status of an I/O device or the conditions under which an I/O operation has been terminated. The CSW is formed (or parts of it are replaced) in the process of I/O interruptions and during execution of the start I/O, test I/O, and halt I/O instructions. The channel status word is placed in main storage at location 64; it is available to the program at this location until such time as the next I/O interrupt occurs or until another I/O instruction causes its contents to be replaced (Figure 7).

Storage Protect Key: Bits in position 0-3 form the storage protect key.

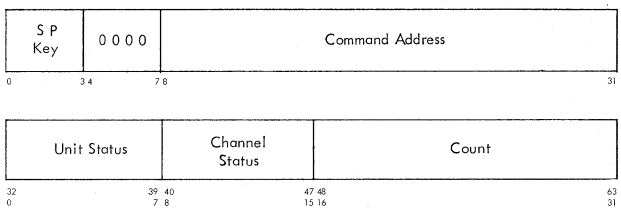


FIGURE 7. CHANNEL STATUS WORD (CSW)

Command Address: Bits 8-31 address a main storage location eight bytes higher than the address of the last channel command word used.

Status: Bits in positions 32-47 identify the conditions in the I/O device and channel that caused the channel status word to be stored. The I/O device supplies bits in the status byte for positions 32-39. The channel provides bits for positions 40-47 to indicate conditions associated with the subchannel. Each of the sixteen bits (32-47) represent one condition.

Bit 32 Attention	Bit 40 Program-controlled interruption
Bit 33 Status modifier	Bit 41 Incorrect Length
Bit 34 Control unit end	Bit 42 Program check
Bit 35 Busy	Bit 43 Protection check
Bit 36 Channel end	Bit 44 Channel data check
Bit 37 Device end	Bit 45 Channel control check
Bit 38 Unit Check	Bit 46 Interface control check
Bit 39 Unit exception	Bit 47 Chaining Check

Count: Bits in positions 48-63 form the residual count for the last CCW used.

Unit Control Word

The 128-bit unit control word (UCW) stores the information necessary to sustain a single I/O operation. Bump storage has 256 unit control words; each consists of four 32-bit words (Figure 8).

The address of each unit control word is developed from the I/O device address; however, the UCW is not program addressable. The initial UCW is established in bump storage during the execute phase of the start I/O instruction.

At the start of an I/O operation, the associated unit control word is fetched from bump storage and stored in local storage; this permits modification of the UCW without addressing main storage if more than one data byte is transferred. When the channel transfers a data byte, the data count and data address portions of the UCW are modified; thus, channel maintains an updated data count and data

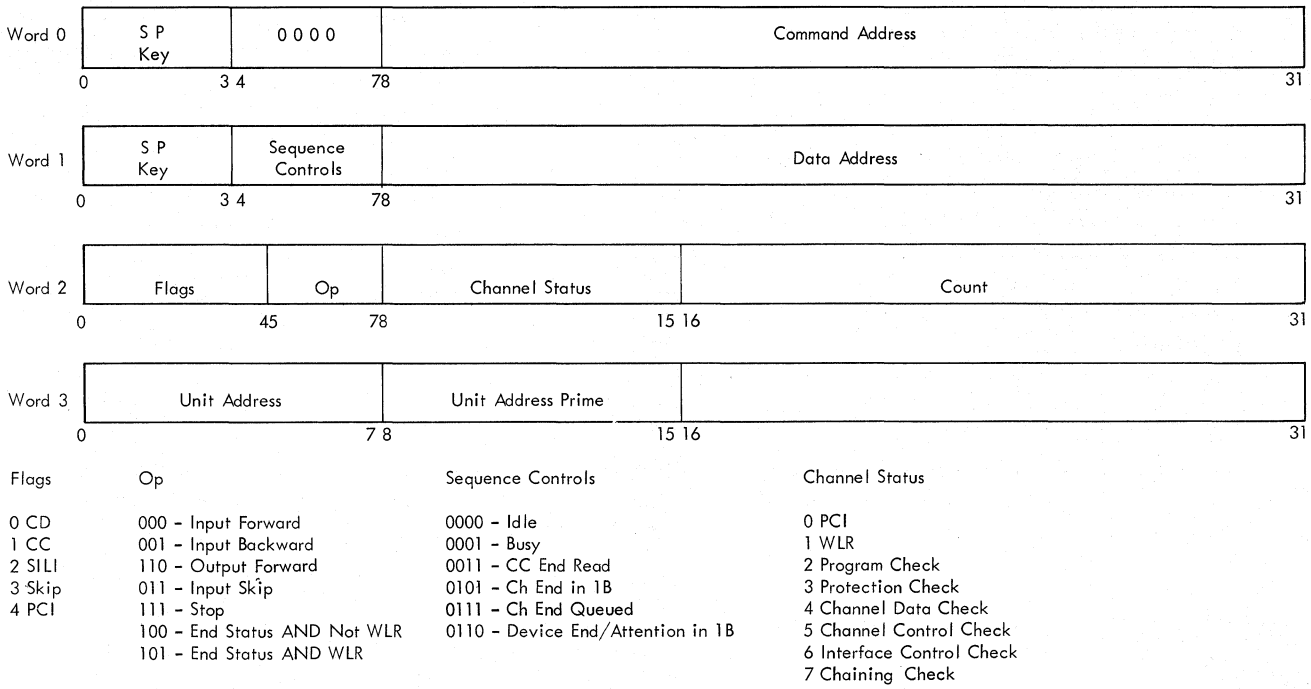


FIGURE 8. UNIT CONTROL WORD (UCW)

address in local storage. When data transfers stop, the updated UCW is returned to bump storage.

Fields in the UCW are allocated for the following purposes:

Word Zero

Storage Protect Key: Bits in positions 0-3 are the storage protect key for the I/O operation. The storage protect key is copied from the channel address word. Bit positions 4-7 are not used.

Command Address: Bits in positions 8-31 designate the main storage location that contains the CCW.

Word One

Storage Protect Key: Bits in positions 0-3 are the storage protect key for the I/O operation.

Sequence Controls: Bits in positions 4-7 are used to show the state of the subchannel. The sequence controls are modified during I/O operations to provide a current record (the state) of the subchannel activity. The current state of the sequence controls is used by the multiplexor channel routines to direct I/O operations. Six possible states of the sequence controls are defined for the following purposes:

Idle (0000) -- The addressed subchannel, control unit, and I/O device are not engaged in the execution of any previously initiated operations, and do not contain any pending interrupt conditions.
Busy (0001) -- The addressed subchannel is executing a previously initiated operation and has not yet reached the channel end.
Channel End in the Interrupt Buffer (0101) -- The I/O device has received or provided all information associated with the operation and no longer needs channel facilities, and the interrupt buffer is full.
Device End or Attention in the Interrupt Buffer (0110) -- The I/O device has terminated the operation, and the interrupt buffer is full.
Command Chaining End Received (0011) -- Channel end has been received prior to device end when the operation specifies command chaining. If the sequence controls equal command chaining end received, receipt of device end causes the channel to execute the command chaining routines. If the command chaining flag is set, receipt of channel end does not cause the multiplexor channel to set the interrupt buffer full.
Channel End Queued (0111) -- Channel end, issued to the multiplexor channel when the interrupt buffer contained interrupt data related to another subchannel operation, causes the channel to set the sequence controls equal to channel end queued.

Data Address: Bit positions 8-31 designate the first main storage location for data fetched or stored by an I/O operation.

Word Two

Flags In: Bits in positions 0-4 contain the flags copied from the channel command word. The flags are:

Chain-Data (CD) Flag -- specifying the chaining of data. Chaining data causes the storage area designated by the next channel command word to be used with the current operation.

Chain-Command (CC) Flag -- specifying chaining of commands when the chain data flag is zero. Chaining commands causes the operation called for by the command code in the next channel command word to be initiated on normal completion of the current operation.

Suppress-Length-Indication (SLI) Flag -- controls whether an incorrect length condition is to be indicated to the program. When this bit is 1 and the chain data flag is 0 in the last CCW used, the incorrect length indication is suppressed. When both the chain command and the suppress-length-indication flags are 1, command chaining takes place regardless of the presence of an incorrect length indication. Absence of the suppress-length-indication flag or the presence of the chain data flag causes the program to be notified of the incorrect-length condition when it occurs.

Skip (SKIP) Flag -- specifies suppression of transfer of information to storage during a read, read-backward, or sense operation.

Program-Control-Interruption (PCI) Flag -- causes the channel to generate an interruption condition upon fetching the channel command word.

Op Code: Bits in positions 5-7 are initially coded from the eight-bit command code contained in the CCW. The multiplexor channel modifies the three-bit op code when certain I/O conditions occur.

Channel Status: Bits in positions 8-15 contain information necessary to generate the channel status word. Channel status is updated during I/O operations; the unit control word channel status becomes the channel status byte of the CSW.

Count: The count designates the maximum number of data bytes required. The multiplexor channel updates the count during an I/O operation.

Word Three

Unit Address (UA): Bits in positions 0-7 are the unit address that is copied from the I/O instruction. For the multiplexor channel, the unit address designates the I/O device address as well as the subchannel address in bump storage.

Unit Address Prime (UA*) is generated by modifying the unit address. Unit address prime is used to address a shared subchannel in bump storage. When unit address prime is generated, it is stored in local storage, never in bump storage.

CHANNEL COMMANDS

- The six channel commands are: write, control, read, read backward, sense, and TIC.

Write

- In a write operation, the channel transfers a byte at a time from main storage to the I/O unit.
- The channel prefetches bytes in write operations.

In a write operation, ROS directs the transfer of data from main storage, through CPU components, to the channel. The channel transmits eight-bit data bytes (nine bits including parity) over the interface to the I/O unit. The I/O unit writing the record must request service from the channel to initiate output data transfers across the interface. ROS fetches data from main storage in ascending order of addresses; the data address in the first channel command word (CCW) designates the starting address.

"Prefetching" is the process of obtaining a new byte from storage before the I/O operation requires the information. In a write operation, the channel prefetches a data byte and holds the byte in the channel buffer out latches for transmission to the unit. When the unit requires data, the unit stores the prefetched byte and signals channel the byte is stored. Each time channel buffer out latches are unloaded, the channel prefetches a new data byte until the unit data requests stop, or until all the words in the storage block are transferred to the unit.

Control

- The control operation usually does not require data transfers across the interface.
- If the control operation requires data transfers to the I/O unit, the channel processes data as in a write operation.

Execution of the control command usually does not require data transfers across the interface. After the channel designates the specific control operation in the command byte, the addressed unit performs the operation without intervention from the channel. For example, a control command can indicate rewind, backspace, or forward space on a magnetic tape unit. The device to which the control command is addressed determines the actions that the command will cause.

If the command code does not specify the entire control function, the data address field of the CCW designates the memory location of additional information required for the operation. The channel does not distinguish between the control and write commands.

Read and Read Backward

- In read and read backward operations, the channel accepts a byte at a time from the I/O unit and transfers words to main storage.
- In a read operation, words are stored in main storage in ascending order of addresses.
- In read backward operations, words are stored in main storage in descending order of addresses.

In read and read backward operations, the channel accepts eight-bit bytes (nine bits including parity) from the interface. The channel transfers data through CPU components to main storage.

In a normal read operation, ROS stores data in main storage in ascending order of addresses. In a read backward operation, ROS gates data to main storage in descending order of addresses. The data address in the CCW specifies the storage location in which the first word is stored in read and read backward operations.

Sense

- Sense and read operations differ only in the type of information processed when executing the respective commands.
- The channel does not recognize the difference between read and sense commands.

When executing either a read or sense operation, the addressed unit transfers data across the interface to the channel. Sense and read operations differ in the type of information transferred when executing the respective commands. In a sense operation, data bytes describe the current status of the I/O device and unusual conditions that the device detected in the last operation. Because the channel does not examine data bytes, it does not recognize any difference between read and sense operations.

Transfer-In-Channel

- TIC allows a CCW to refer to the channel to a different main storage location for the next CCW.
- TIC is associated only with chain operations.

Actions caused by the transfer-in-channel (TIC) command differ from actions that other channel commands initiate in that:

1. The transfer-in-channel (TIC) command does not begin an I/O operation.
2. An I/O unit does not participate in the execution of the TIC.
3. The start I/O instruction cannot designate a TIC; therefore, the CCW that the CAW locates may not contain a TIC.

The TIC command allows a CCW to refer the channel to a different storage location to obtain the next CCW to continue the chain data or chain command operation. The data address in the CCW specifying a TIC designates the storage address of the next CCW to be used. A CCW specifying a TIC may not address another CCW that also designates a TIC. When CCW-1 contains a TIC command, the channel is still committed to fetch CCW-2. In this case, however, control information in CCW-2 does not affect the operation.

MULTIPLEXOR CHANNEL I/O OPERATIONS

- I/O operations are divided into four major phases: initial selection, data handling, ending, and interrupt (Figure 9).
- The type of operation, control data, and errors that occur determine the number of phases executed in an operation, and the order in which they are performed.

Because the multiplexor channel can perform only limited action independent of microprogram control, the channel requests microprogram routines to perform each phase of the operation. In most phases, required channel actions exceed the six-microinstruction limitation of a microprogram routine; therefore, the channel must request a series of routines to perform its tasks in each phase. Because conditions revealed in one routine usually determine subsequent channel actions in the phase, the routine series in each phase is flexible. Each

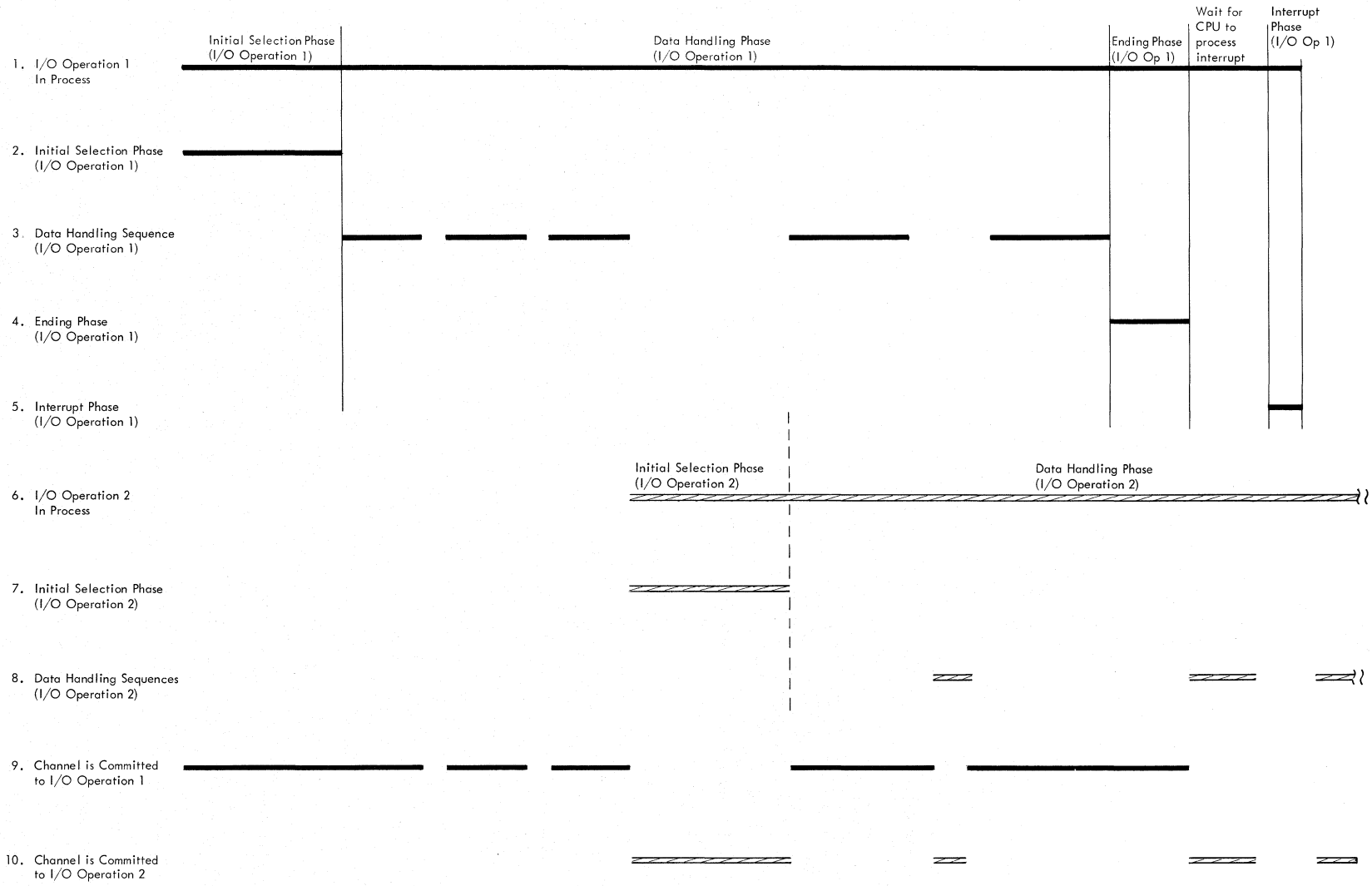


FIGURE 9. MULTIPLEXOR CHANNEL I/O OPERATIONS

phase can be divided into parts equal to the number of routines that the channel requires.

Initial Selection

- In the initial selection phase, the channel and I/O unit prepare to execute the operation.

Before the common channel signals the multiplexor channel to initiate an I/O operation, CPU:

1. Fetches the channel address word (CAW)
2. Fetches the channel control word (CCW-1 and CCW-2)
3. Stores, in the L register, the address of the I/O unit designed to perform the operation.
4. Enters a timed countdown loop to wait for a reply.

If the channel is free, the signal from common channel starts the initial selection phase of the operation. In this phase, the channel requests routines to:

1. Fetch the channel control words and the I/O unit address to select bump storage locations reserved for multiplexor channel operations.
2. Initiate the interface sequence to select the unit. If the unit is available, the channel identifies the operation to be performed and prepares the unit to execute the operation.

If the channel does not detect errors in the Channel Control Word or in the selection of the I/O unit, it signals common channel to release CPU from the countdown loop. The CPU then proceeds to execute other instructions in the customer program while the channel directs the I/O operation.

If, during the initial selection phase, the channel determines that it cannot perform the operation, it (1) provides information to the CPU indicating the reason, (2) signals common channel to release CPU from the countdown loop.

Data Handling

- In the data handling phase, the channel executes the data transferring operation that CPU requested.

In the data handling phase of the operation, the channel requests routines to:

1. Transfer bytes between main storage and the I/O unit
2. Check for parity errors
3. Update the count and data address
4. Determine whether the current transfer is the last in the operation.

The channel requests the routine or sequence of routines to perform these actions:

1. When the I/O unit initiates a data transfer to main storage for input operations, or
2. Each time (after the initial selection phase) that the I/O unit accepts data in output operations.

Unlike other phases in the operation, the data handling phase need not proceed uninterrupted. Each time that the I/O unit drops "operational in," the channel is free to break from the data handling phase to service another I/O unit or to answer a request from common channel to begin a new operation on another unit. The data handling phase resumes at the point of interruption when the I/O unit can capture the channel again (Figure 9).

The data handling phase is terminated:

1. By the channel, when either a parity error is detected or control information indicates that the channel has processed the last data byte to or from the I/O unit
2. By the I/O unit, when the unit transfers the last byte of available data.

Ending

- In the ending phase, the channel performs a "mop-up" operation and prepares and end of job report for CPU.

If neither the chain data nor the chain command flag is set in the ending phase of the operation, channel routines:

1. Form control information to indicate that the channel has performed the last data transfer in the operation
2. Accept, then analyze, the status byte from the I/O unit
3. Load the interrupt buffer and set the interrupt buffer full latch when conditions permit
4. Update control words in bump storage to indicate channel conditions at the end of the operation.

Interrupt

- CPU performs the interrupt phase to take the end of job report that channel prepared in the ending phase.
- Operations in process on the multiplexor channel are not disturbed while CPU executes the interrupt phase.

The channel transmits the interrput buffer full indication to the comman channel and to CPU. The common channel requests an I/O interrupt. During the next I-fetch, CPU signals common channel to proceed with the interruption and enters another countdown loop. In response, common channel unconditionally sets a CPU stat to allow CPU to break from the countdown loop immediately. With the interrupt buffer full indication still active after CPU leaves the countdown loop, the stat setting

causes CPU to fetch data from both the interrupt buffer and a specified section of bump storage to form the channel status word (CSW).

The contents of the CSW indicate to the customer program information about the execution of the completed I/O operation.

BUFFER 1 AND BUFFER 2

- The buffer-out-bus transfers bits from the mover to buffer 1 or buffer 2.
- The buffer-in-bus transfers bits from buffer 1 or buffer 2 to the mover.
- Interface bus in lines transfer bytes from I/O units to buffer 1 or buffer 2.
- Only buffer 2 can transmit bytes over bus out lines to I/O units.

Two nine-bit buffers in the multiplexor channel serve as temporary storage registers in I/O operations. Interface bus in lines and the mover are input sources for both buffers. The nine-position buffer out bus connects mover outputs to buffer 1 and buffer 2. The buffer in bus, also nine positions, connects both buffer outputs to the mover. The multiplex gate control field, positions 49, 50, and 51 of ROSDR, controls input and output buffer gating (Figure 10).

Buffer 1 stores: (1) control bytes from interface bus in lines and (2) control bits from the mover.

Buffer 2 is a temporary storage register for both control and data bytes. All byte transfers from the multiplexor channel to an I/O unit are transmitted from buffer 2; only buffer 2 sets interface bus out latches to transmit bytes over bus out lines. All input data transfers from I/O units set buffer 2. Buffer 2 also stores control bytes from interface bus in lines.

Figure 11 shows buffer 1 and buffer 2 input and output paths.

MULTIPLEXOR I/O BYTE STATS

- I/O byte stats designate byte(s) to be replaced in the word read from main storage.
- Channel does not use I/O byte stats in transferring data bytes from main storage in output operations.

To change bytes in storage, a multiplexor channel routine causes a word to set the storage data register (SDR). The microprogram replaces one or more bytes in the SDR word with new information, then returns the new word to storage. Multiplexor I/O byte stats designate the byte(s) to be changed. The multiplexor channel can process only one data byte at a time to main storage; therefore, only one of the four I/O byte stats is set when a channel routine transfers data to main storage. The channel does not use outputs from I/O byte stats in processing a data byte from main storage to an I/O unit.

In a byte mode read operation, the channel sets the I/O byte stat corresponding to the DAB. For example, if the DAB is 11 (3), the channel sets I/O byte stat 3; I/O byte stat 0 is set if the DAB is 00. When receiving a burst of bytes in a read operation, the channel sets the I/O byte stat:

1. Corresponding to the DAB when storing the first byte in the burst
2. One higher than the DAB when storing all bytes in the burst (except the first, if bytes are to be stored in a forward sequence)
3. One lower than the DAB when storing all bytes in the burst (except the first, if bytes are to be stored in a backward sequence).

For example, if the DAB is 11 (3), when the routine is in process to store the second in a burst of bytes in a forward sequence, the channel sets I/O byte stat 0. If the DAB is 00, when the routine is in process to store the second in a burst of bytes in a backward sequence, the channel sets I/O byte stat 3.

The two multiplexor channel routines (A1 and B6) that fetch, update, and store the data address always require a word transfer from storage to the SDR. In all cases, these routines replace bytes 1, 2, and 3 in the SDR word; therefore, execution of either routine unconditionally sets I/O byte stats 1, 2, and 3. The routine to fetch and update the count (A0) transfers a word from storage to the SDR, then replaces bytes 2 and 3 in the SDR word. The channel always sets I/O byte stats 2 and 3 when this routine is in process.

DATA FLOW

Figure 12 shows the data path between main storage and I/O units attached to the multiplexor channel.

Input

- A data byte transfer from the I/O unit sets buffer 2; later the channel transfers the byte through the mover to the L register, from the L register through the adder to the SDR, and from SDR to main storage.

In an input operation, a selected I/O unit transfers a byte at a time on bus in interface lines to buffer 2 in the multiplexor channel. The data address word, stored either in bump or local storage, designates the main storage location to receive the byte in buffer 2.

A multiplexor channel microprogram fetches the data address word to the R register and transmits the byte in position 3 of the data address word (the byte containing the DAB) through the mover to

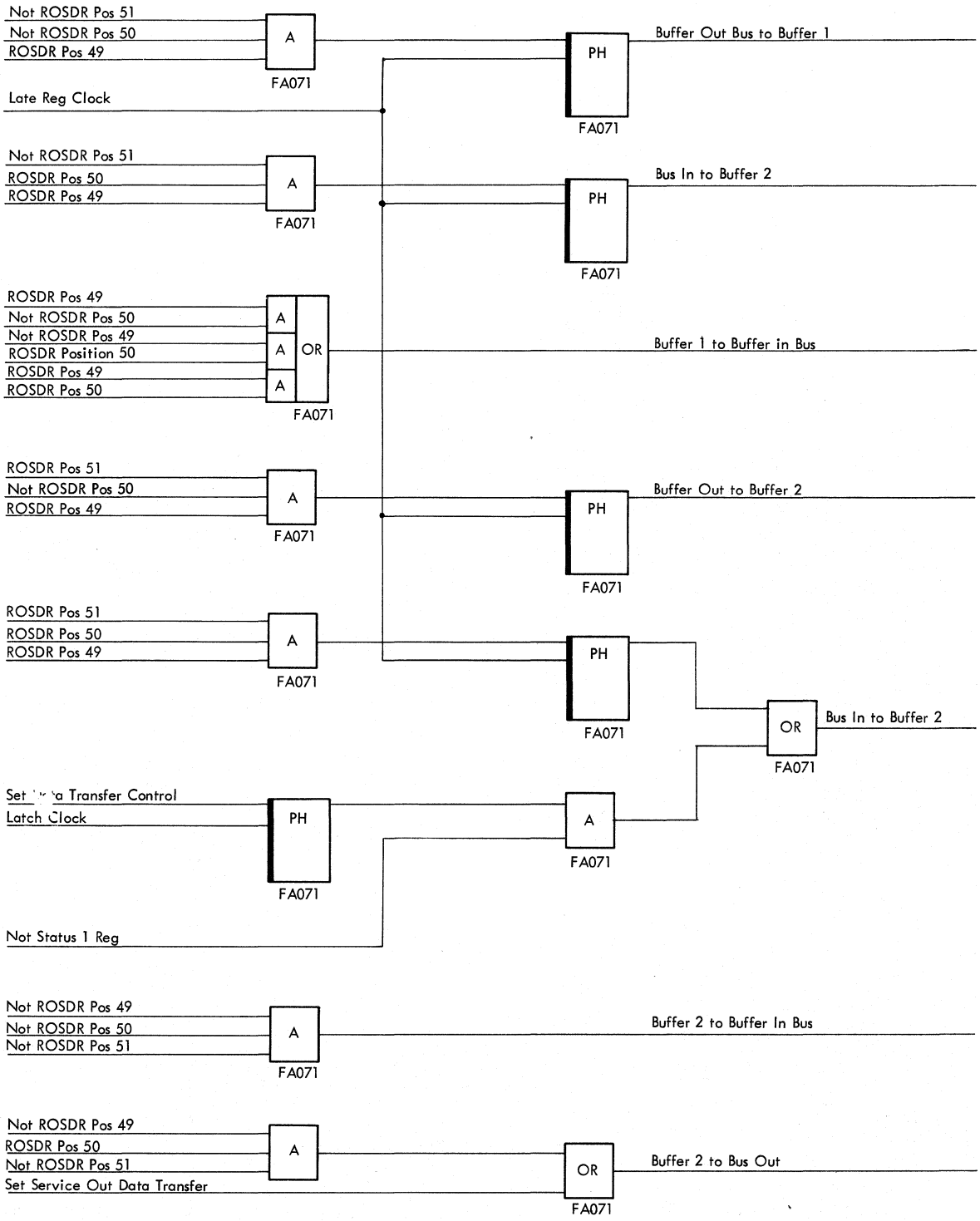


FIGURE 10. INPUT AND OUTPUT BUFFER GATES

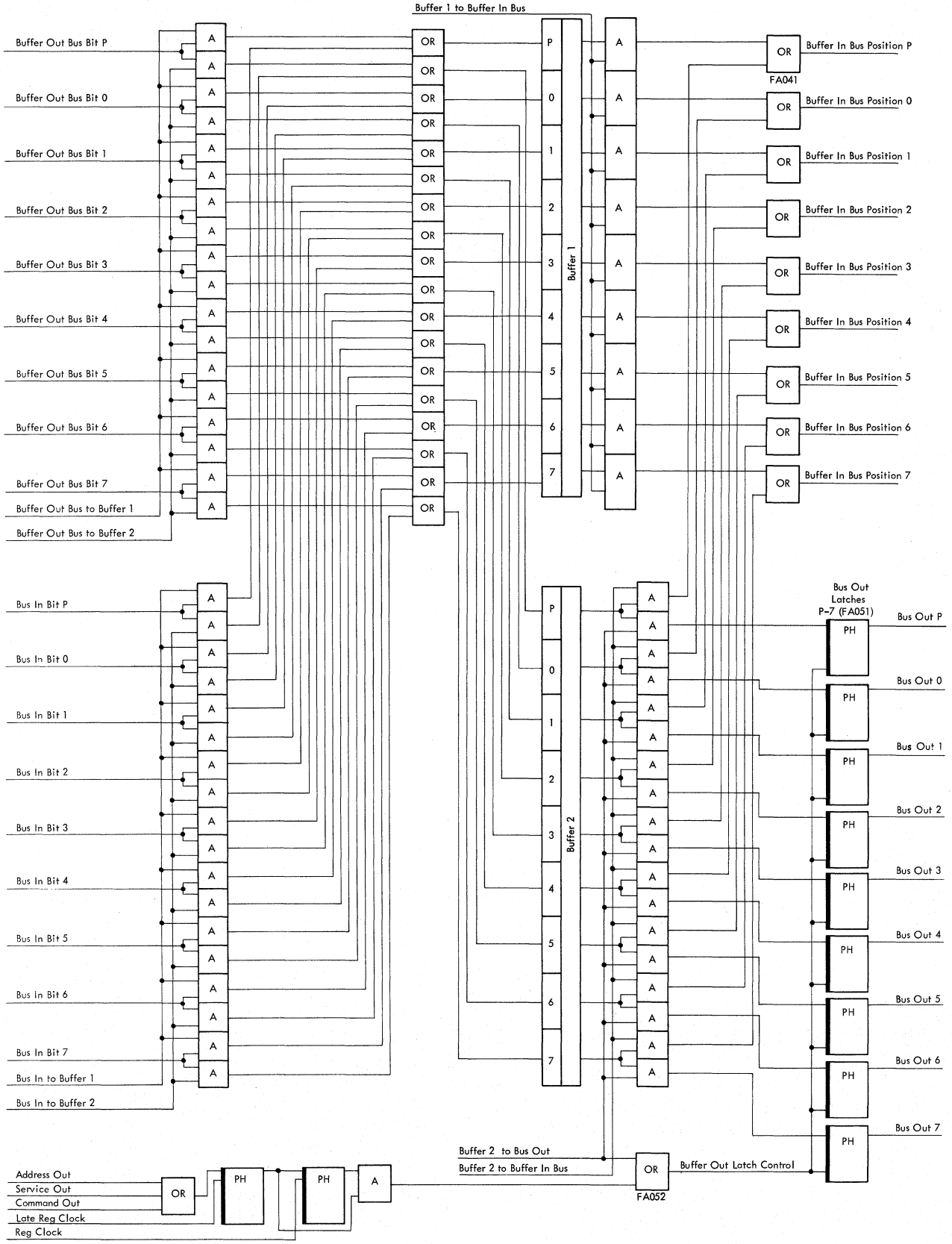
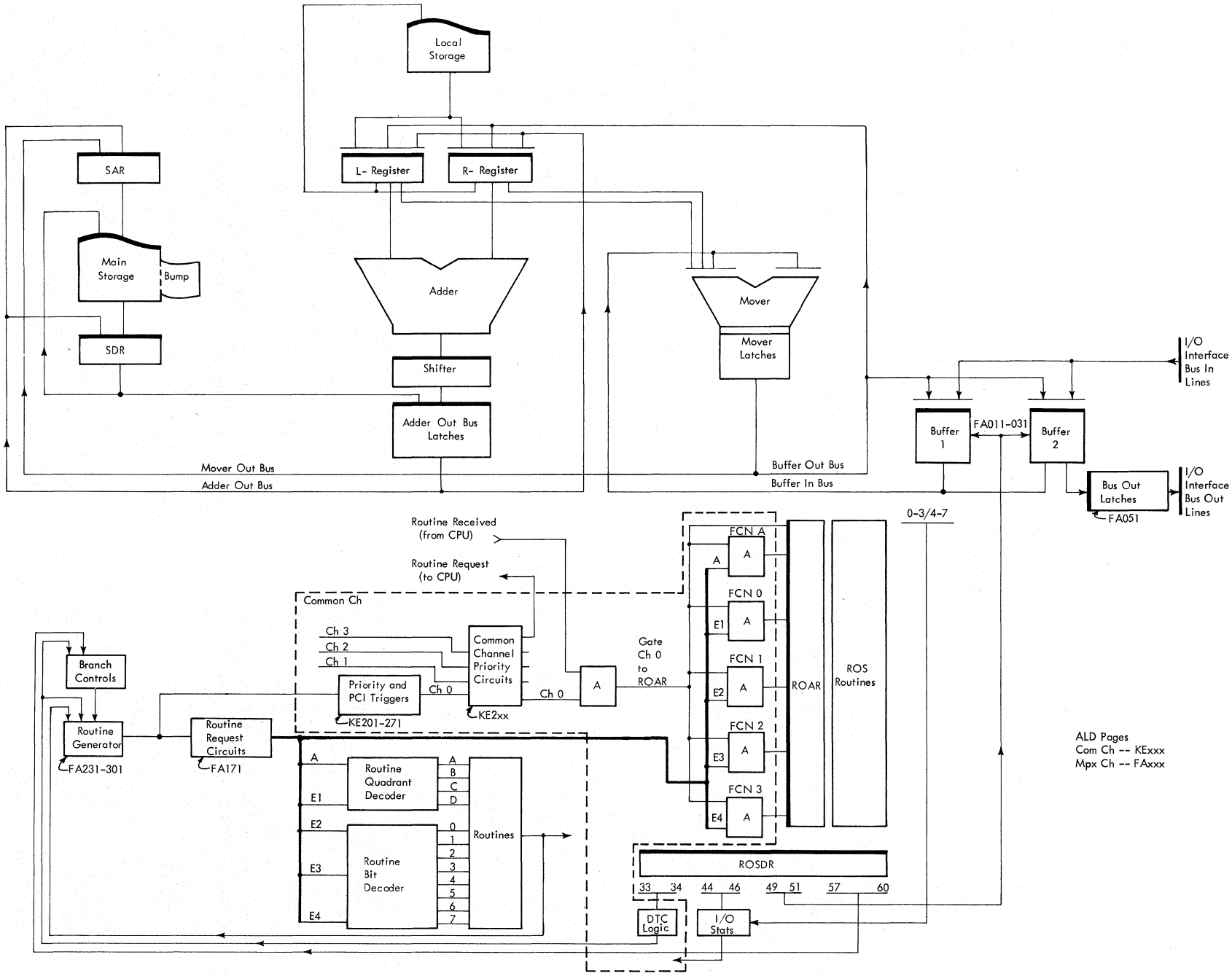


FIGURE 11. BUFFER 1 AND BUFFER 2 SETS AND RESETS



● FIGURE 12. DATA FLOW IN MULTIPLEXOR CHANNEL OPERATIONS

buffer 1. Buffer 1 bit positions 6 and 7 (the DAB) set a multiplexor I/O byte stat.

A microinstruction gates the byte in buffer 1 onto the buffer in bus to the mover, setting mover latches. A subsequent microinstruction gates the data in buffer 2 onto the buffer out bus, through the mover to the L register. The settings of mover latches 6 and 7 correspond to the two low-order bits in the data address word (the DAB) and specify the L register byte position to receive the data byte from buffer 2.

The data address word in the R register sets the storage address register (SAR), causing four bytes (one word) to read out of main storage to the storage data register (SDR).

Although only one byte of usable data is stored in the L register, all positions of the L register transfer to the adder-out-bus. The multiplexor I/O byte stat, set previously in the operation, gates the valid byte in the word on the adder-out-bus to the corresponding position of the SDR. The new word, containing three bytes of the word read from main storage and one byte of new data, is stored in the same main storage locations read out earlier.

Output

- In an output operation, the channel transfers a byte at a time from main storage to the selected I/O unit.

In each output data transfer, the multiplexor channel:

1. Fetches a data word from main storage to the L register
2. Selects one byte in the word in the L register and transfers the byte through the mover onto the buffer-out-bus to buffer 2
3. Transmits the byte in buffer 2 over bus out interface lines to a selected I/O unit.

The data address word, stored either in bump or local storage, designates the main storage location containing the byte to be transferred.

A multiplexor channel microprogram fetches the data address word to the R register. Subsequent steps in the microprogram transfer the byte in position 3 of the data address word (the byte containing the DAB) to buffer 1, and later, gate the byte in buffer 1 onto the buffer in bus to the mover, setting mover latches. The setting of mover latches 6 and 7 corresponds to the two low-order bits in the data address (the DAB).

The data address word in the R register sets the storage address register (SAR), causing four bytes (one word) to read out of main storage to the storage data register (SDR). Although the word read from main storage contains only one byte of useful data, the complete word transfers from the SDR to the L register. The settings of mover latches 6 and 7 identify the L register position that contains the useful data and gates that byte to buffer 2 for transmission to the selected I/O unit. Other data bytes in the L register are not used in this data transfer.

I/O STATS

- The four I/O stats in the multiplexor channel provide temporary storage for control information in I/O operations.
- Varied use of I/O stats minimizes the number of storage facilities in the multiplexor channel.

Because the channel has unrestricted use of I/O stats, information stored in the stats can be used for several purposes and can represent different conditions. Use of the I/O stats can be categorized as follows:

1. To indicate branch conditions for microprogram routines, and
2. To provide gates for other multiplexor channel circuits.

Normal inputs to the I/O stats are: (1) the emit field, (2) the buffer in bus, and (3) the buffer in bus under control of the emit field. If the stat control field, positions 44, 45, and 46 of ROSDR, does not contain all zeros, the bit combination designates one input source to set the I/O stats (Figure 13). I/O stat settings have a specific meaning in each routine; therefore, each routine decodes I/O stats for a specific, predetermined interpretation.

DATA TRANSFER OR CONTROL (DTC) LEVELS

- DTC levels are gating lines that identify a specific period in the microprogram.
- Microinstructions that generate DTC levels have no assigned position in each routine.
- A routine generates DTC-1 only after the channel can predict subsequent actions in the operation.
- A routine generates DTC-2 to initiate various actions, depending on the routine that issues the signal.

In executing multiplexor channel routines, ROS directs actions in both CPU and the multiplexor channel. In many cases, however, the concurrent operations in CPU and channel facilities are not directly related. For example, while ROS is performing a routine to transfer a data byte to an I/O device, special channel circuits may generate a request for the next routine. To synchronize the indirectly related actions, selected microinstructions in multiplexor channel routines contain DTC-1 and DTC-2 levels.

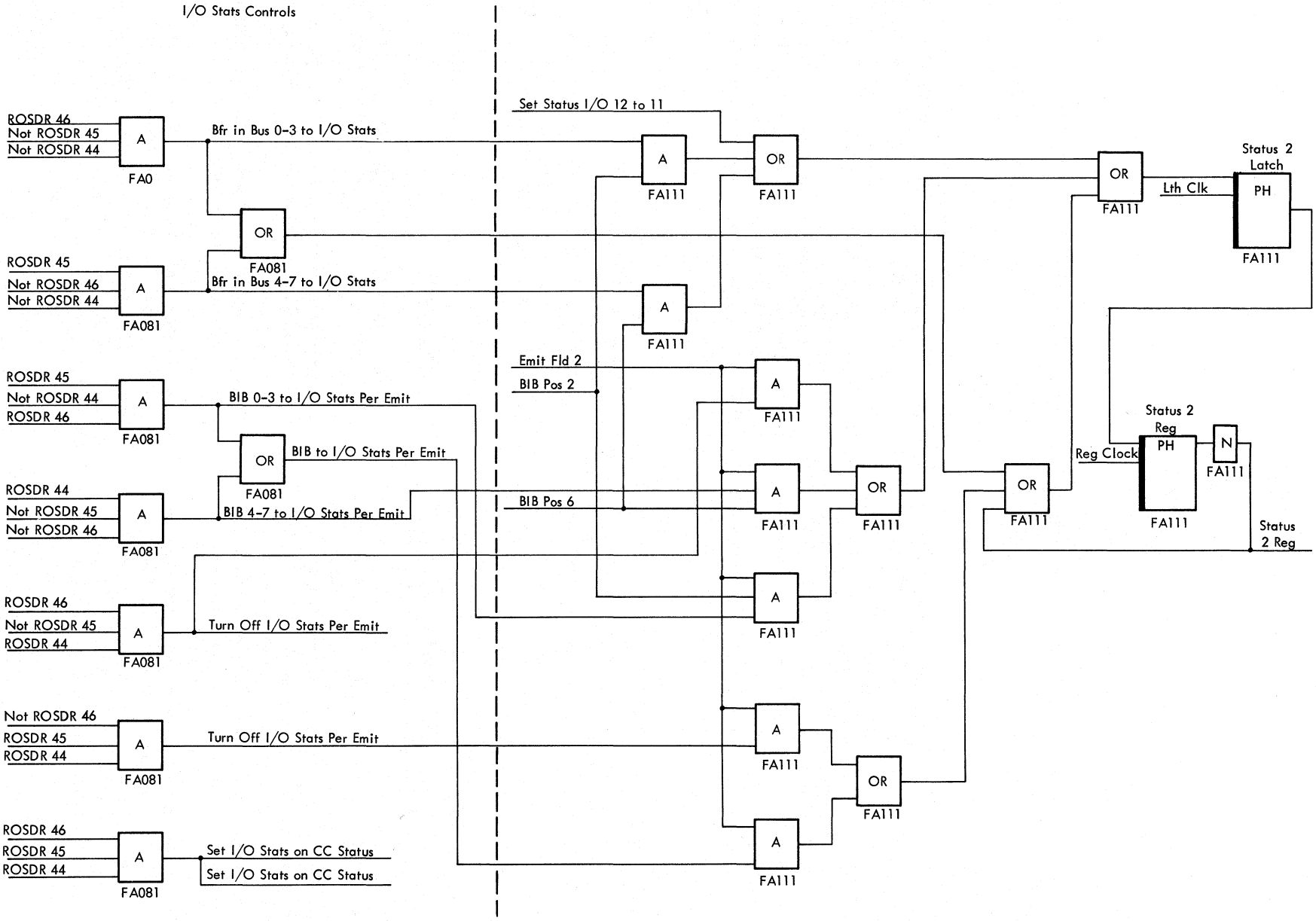


FIGURE 13. MULTIPLEXOR I/O STAT 2 AND ASSOCIATED GATING CIRCUITS

DTC-1: ROS bit positions 33 and 34 are 1 and 0, causing the common channel to send the DTC-1 signal to the multiplexor channel. The primary use of the DTC-1 line is to lock the branch registers, giving specific meaning to the outputs of the branch control circuits.

Although DTC levels are active for only one cycle, DTC-1 sets the DTC-1 latch in the channel. If the multiplexor channel requests another routine while a previously initiated multiplexor channel routine is in process, the DTC-1 latch is reset after the request is made but before the routine in process ends. If the DTC-1 latch is set and the multiplexor channel does not request further service before the present routine ends, the reset DTC latch is set to turn off the DTC-1 latch.

DTC-2: ROS bit positions 33 and 34 are 1 and 1, causing the common channel to send the DTC-2 signal to the multiplexor channel. The DTC-2 level has unique uses in the following routines:

1. B7 (check handling) resets the check latches.
2. C5 (control unit busy) resets select out.
3. A1 (DA fetch and update) resets the logout control flip-flop.
4. A2 and A3 (data handling) allow requests for the PCI routine (C0) before normal routine requests, if the PCI flag is set.

BRANCH CONTROL CIRCUITS

- The branch register is allowed to set to the emit field until it is 'frozen' by the DTC-1 latch.
- Branch control circuits, set during the execution of a ROS routine, store conditions that help determine subsequent channel actions in the operation.

A four-position branch register and a decoder make up branch control circuits (Figure 14) in the multiplexor channel.

When the DTC-1 latch is off, the branch register is in a "free run" state; the emit field in each microinstruction sets corresponding positions of the branch register. The ON output of the DTC-1 latch locks in the branch register the emit field of the microinstruction that sets the DTC-1 latch.

Outputs from branch register positions are assigned binary values and decoded to produce branch control lines. For example, active outputs from branch register positions 1 and 3 bring up the branch control 5 line; branch control 7 is conditioned when branch register positions 1, 2, and 3 are set.

In all cases, branch control lines combine with either the output of the DTC-1 latch or the DTC-1 pulse to effect any action in the multiplexor channel. Therefore, branch control circuits are useful only after the microprogram has generated the DTC-1 signal.

UNIT CONTROL WORD (UCW) STORE TRIGGER

- The UCW store trigger causes the channel to transfer words from the count and data address positions of local storage to corresponding positions in the bump storage UCW.

The channel sets the UCW store trigger (Figure 15) when:

1. A routine processes the second in a burst of data bytes. In transferring a burst of bytes, the channel updates the count and data address in local storage, but does not change the UCW in bump storage until the transfer is complete.
2. The channel requires a CCW to begin or continue an operation. Channel routines that fetch a CCW form the data address and count words in local storage. If no error occurs, routines transfer the words to bump storage when the I/O unit drops operational in.
3. The count equals 1 in an output operation or 0 in an input operation. If the chain data flag is off, a routine inserts the stop op code in the count word in local storage and, in output operations, decrements the count by 1. A subsequent routine updates the bump storage UCW when the I/O unit drops operational in.

ROUTINE REQUEST CIRCUITS

- ROS contains thirty multiplexor channel microprogram routines.
- Figure 16 shows a diagram of routine request circuits in the multiplexor and common channels.

Routine Generator

- The six-stage routine generator accepts inputs from other multiplexor channel circuits to determine the next microprogram routine to be executed in the I/O operation, and to produce coded outputs to identify that routine.

The routine sequence in the operation being performed determines input requirements to the generator. Examples follow:

1. In most cases, actions that occur in one routine, as indicated by branch control settings, designate the next routine. Inputs to the generator must show the last routine requested and branch control settings.
2. Several channel routines have an established sequence in which one routine unconditionally follows another. Inputs to the generator need only identify

the routine in process.

3. Conditions external to the routine in process may influence the next routine requested. Among these conditions are: UCW store trigger, command chain trigger, interface lines, check latches, etc.

The generator produces combinations of the six output lines (set A, set E1, set E2, set E3, set E4, and reset quadrant) which are a code representation of the ROS address of the routine to be requested. Generator outputs do not request the routine; they identify the routine and initiate the channel actions that may make the request.

Exceptions in designating channel routines to be requested are:

1. The routine generator neither identifies nor initiates actions to request routines A0, B0, and C0.

2. The active state of the any check latch line changes the routine request from B5 to B7, in order to request B7. During execution of routine B7, any check latch drops, allowing the generator to designate routine B5 as the next routine to be requested.

3. The on output from the control check latch combines with the off outputs from the PCI request and priority triggers to designate routine C4 (Figure 16). The channel uses the routine generator in making the request for routine C4, but the generator is not used in making the decision to request the routine.

Request Buffer and Decoders

- The routine request buffer contains five input latches and five corresponding output latches that condition address lines from the multiplexor channel to ROAR, and set the routine quadrant and routine bit decoders.
- The routine quadrant decoder indicates the quadrant of the routine in process.
- The routine bit decoder indicates the numeric designation of the routine in process.

The routine generator and special circuits to identify routines A0, B0, and C4 set the five input latches to the request buffer (Figure 16). When common channel and ROS honor the channel 0 request for service, input request buffer latches set ROAR positions 6 - 9 and A, and corresponding latches at the output of the request buffer.

Two of the five latches at the output of the routine request buffer (positions A and E1) set the routine quadrant decoder; the other three latches (positions E2, E3, and E4) set the routine bit decoder (Figure 16).

Each of the four possible input combinations to the routine quadrant decoder produces one of four output lines (routine quadrant decoder A, B, C, or D). Each of the eight input combinations to the routine bit decoder produces one of eight outout lines (routine bit decoder 0, 1, 2, 3, 4, 5, 6, or 7). The active output line from the quadrant decoder combines with the active bit decoder output to identify the routine in process. For example, when ROS begins execution of routine B0, the request buffer supplies inputs to: (1) the routine quadrant decoder to produce the output routine quadrant decoder B, and (2) the routine bit decoder to produce the routine bit decoder 0 output. Normally, request buffer latches remain set, holding decoder outputs active, until the channel requests the next routine.

Routine Sequences

A predetermined sequence controls the order in which several channel routines are executed. For example, in the routine sequence B0 -- B1 -- B2 -- B3, the channel can request routine B1 only when ROS is executing routine B0; the request for routine B3 is made only when routine B2 is in process. Routines in sequence have the same quadrant designation; in the example, routines have the quadrant designation B. When ROS begins execution of the first routine in the sequence (B0 in the example), the routine request buffer sets the quadrant and bit decoders. To request the second routine in sequence (B1 in the example), the two input request buffers that control the input to the routine quadrant decoder are not reset. Therefore, inputs to the quadrant decoder do not change when ROS begins execution of the second routine. To show this action, assume that ROS is executing routine B0 and that the routine generator has determined that routine B1 is next to be requested:

In designating routine B1, the routine generator resets the three request buffer input latches that control the input to the bit decoder (E2, E3, and E4), then sets latch E4. When ROS begins execution of routine B1, latch E4 causes the request buffer to supply the input to the bit decoder to produce the routine bit decoder 1 output. Routine quadrant decoder B, conditioned during the execution of routine B0, does not change.

Among other channel predetermined routine sequences are:

A0 -- A1
B5 -- B6
C1 -- C2 -- C3
D0 -- D1 -- D2 -- D3
D0 -- D1 -- D4

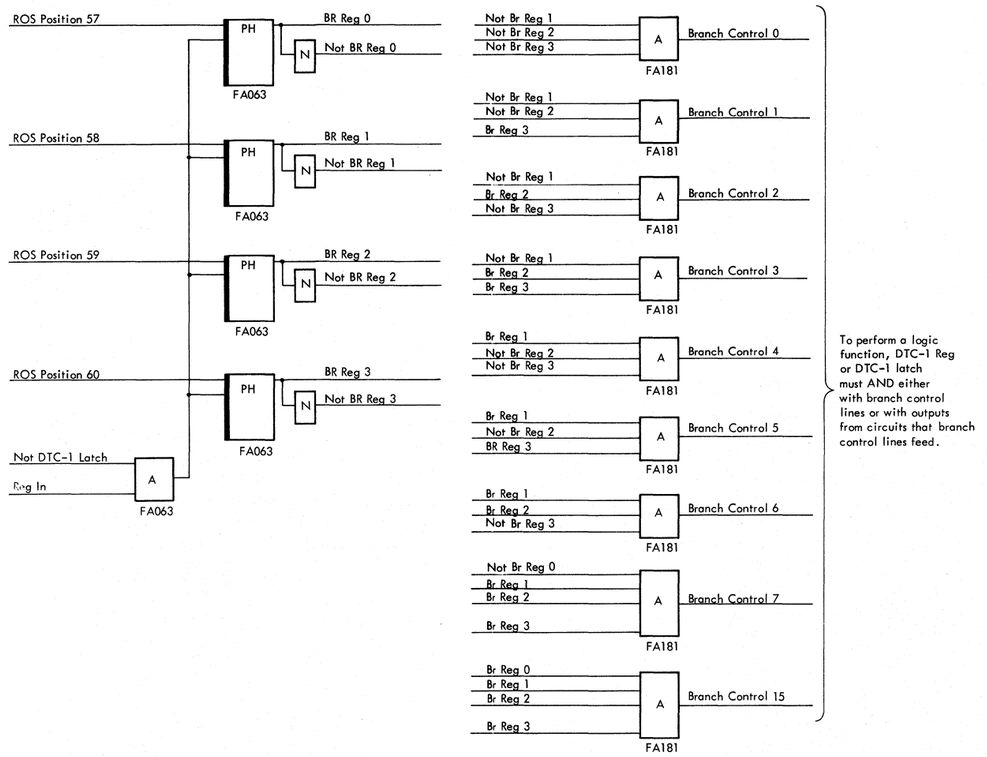


FIGURE 14. BRANCH CONTROL REGISTER AND DECODING

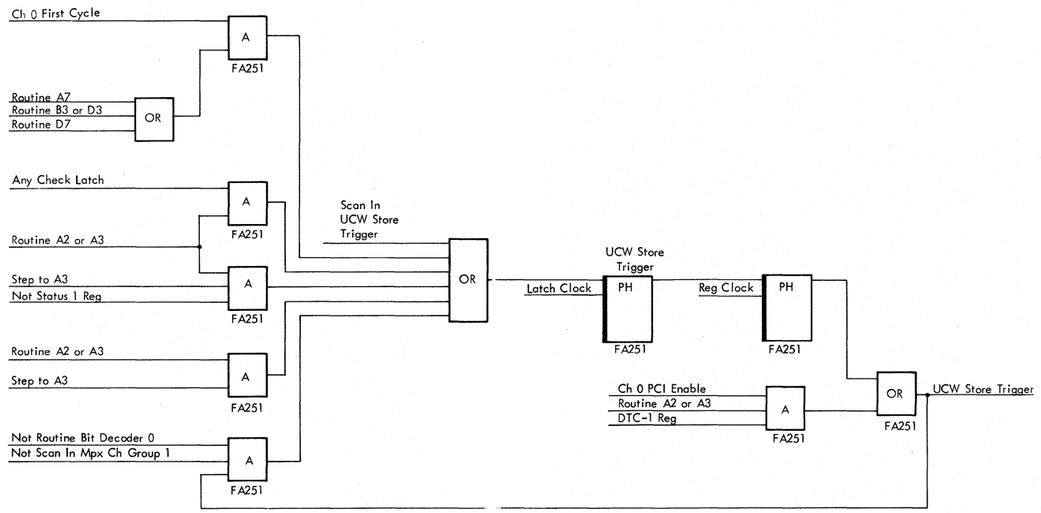
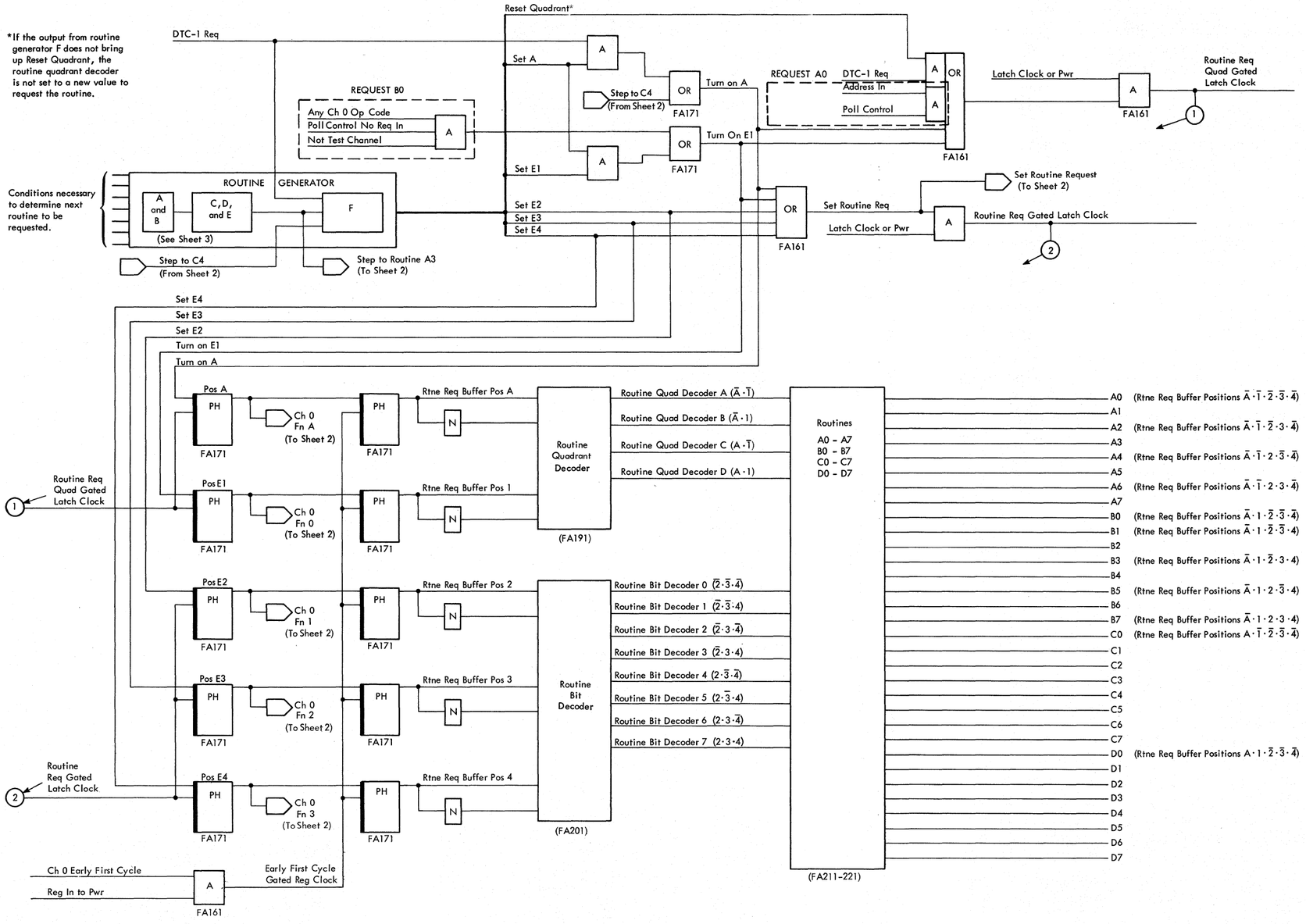
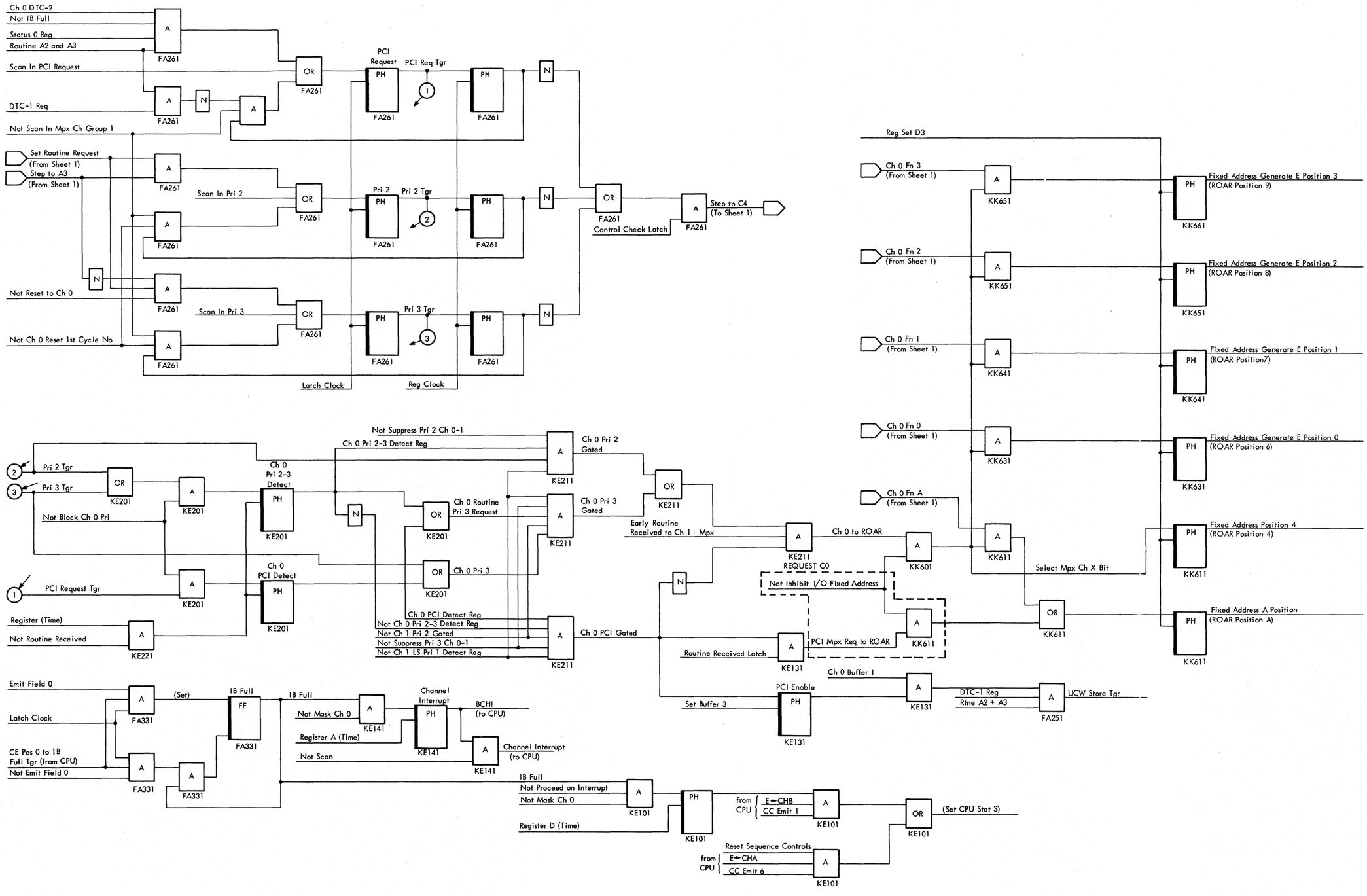


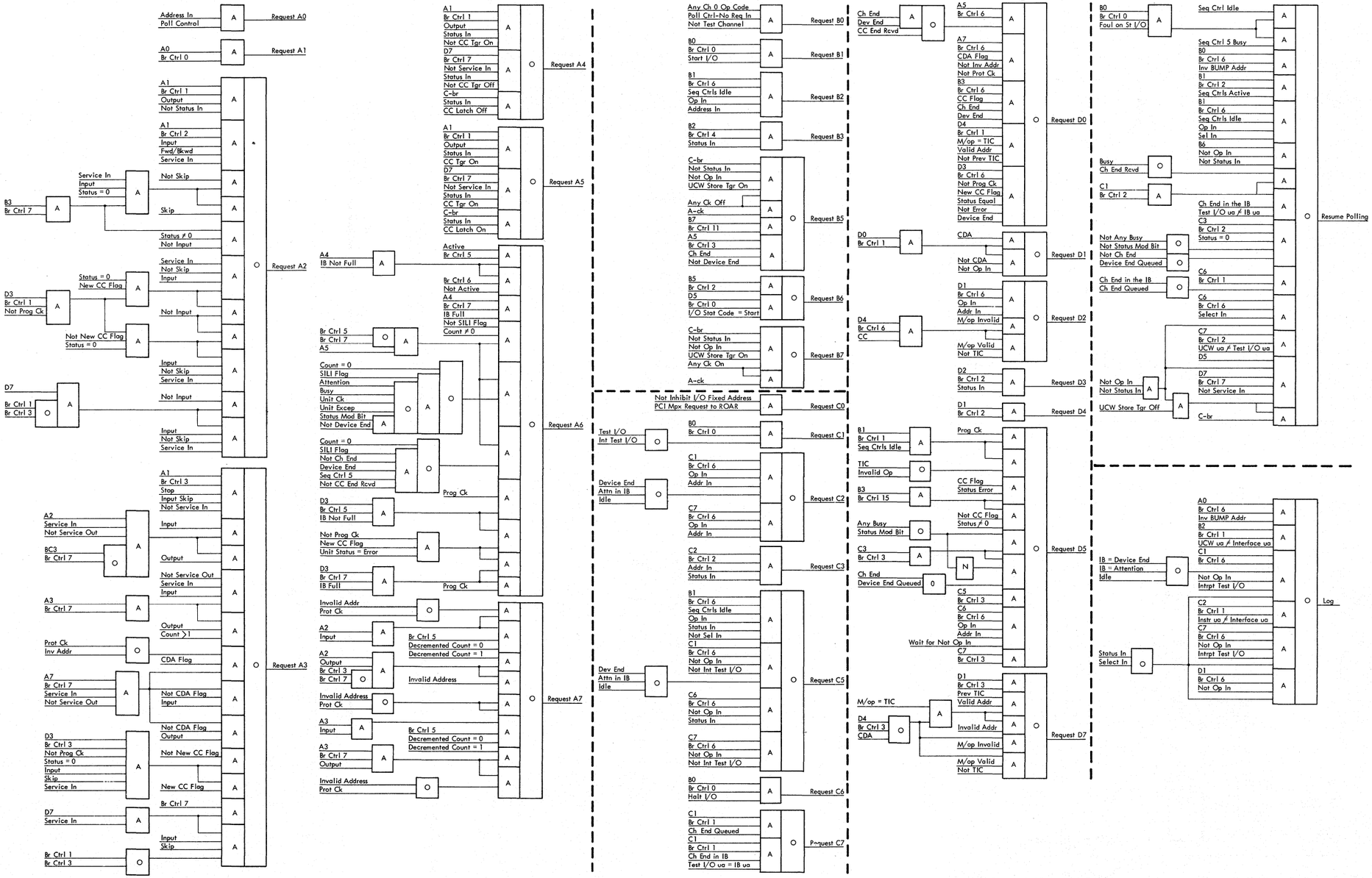
FIGURE 15. UNIT CONTROL WORD (UCW) STORE TRIGGER



● FIGURE 16. ROUTINE REQUEST CIRCUITS (SHEET 1 of 3)



● FIGURE 16. ROUTINE REQUEST CIRCUITS (SHEET 2 of 3)



● FIGURE 16. ROUTINE REQUEST CIRCUITS (SHEET 3 OF 3)

Priorities

- The multiplexor channel assigns priority 2 or priority 3 to requests for routines.
- Common channel processes requests for service from all channels attached to the system.

To request all routines except C0, the multiplexor channel sets one of two priority latches, then transmits the priority latch output to the common channel. Outputs from the routine generator condition set routine request to enable the channel to set priority 2 trigger, to request routine A3, or priority 3 trigger, to request all other routines. The multiplexor channel cannot assign a priority 1 to any request.

The channel that assigns the highest priority to its request for service is first to receive attention in common channel. Should two or more channels simultaneously request service with the same priority, the common channel processes the requests in ascending order of channel numbers (Figure 17). Because channel 0 cannot make a priority 1 request, priority 1 from channel 1 (selector channel) is the most urgent request that common channel can receive.

Ch 1 Local Store Request
Ch 2 Local Store Request
Ch 3 Local Store Request
Ch 1 Priority 1 Request
Ch 2 Priority 1 Request
Ch 3 Priority 1 Request
Ch 0 Priority 2 Request*
Ch 1 Priority 2 Request
Ch 2 Priority 2 Request
Ch 3 Priority 2 Request
Ch 0 Priority 3 Request
Ch 0 PCI Request
Ch 1 Priority 3 Request
Ch 1 PCI Request
Ch 2 Priority 3 Request
Ch 2 PCI Request
Ch 3 Priority 3 Request
Ch 3 PCI Request

* Ch 0 cannot make a local store or priority 1 request.

FIGURE 17. ORDER OF COMMON CHANNEL PRIORITIES

In requesting service, the multiplexor channel must hold its request by blocking the reset of the priority trigger until common channel and ROS permit channel 0 to gate address bits from the request buffer input latches to ROAR.

Common channel conditions routine request to CPU to indicate that a channel requires service. If, at the end of the current machine cycle, main storage is not in the first memory read cycle, CPU conditions routine received to the common channel to honor the request for service to the waiting channel (Figure 16).

If the common channel has designated channel 0 to receive the service, routine received conditions gate channel 0 to ROAR. Bits from the request buffer in the channel set ROAR positions 6 - 9 and A; common channel sets ROAR position 4 for all multiplexor channel requests. The inserted bits complete the fixed ROS address of the first microinstruction in the requested routine.

PCI Request

- The PCI flag in the channel control word causes the channel to make a PCI request.

A 1 bit in the PCI position of CCW-2 causes the multiplexor channel to set I/O stat 0. If the interrupt buffer is not full when ROS executes routine A2 in an input operation or routine A3 in an output operation, I/O stat 0 sets the PCI request trigger, conditioning PCI request trigger to common channel (Figure 16). Common channel assigns a priority 3 to the channel 0 PCI request. When common channel and ROS grant the channel 0 PCI request, the common channel sets ROAR positions A and 5 to complete the ROS address of the first microinstruction in routine C0. Multiplexor channel operation is not disturbed while common channel processes the channel 0 PCI request.

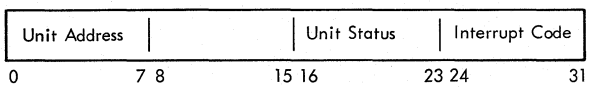
MULTIPLEXOR CHANNEL INTERRUPTS

- CPU routines process multiplexor channel interrupts without disturbing the multiplexor channel operation.
- The interrupt buffer full latch in the channel initiates the request for an interrupt.

ROS executes a channel routine to load the interrupt buffer as shown in Figure 18 and set the interrupt buffer full (IB full) latch when:

1. Common channel and ROS grant a channel 0 PCI request, or
2. An I/O unit transfers an end status byte to the channel (routine A6).

The channel transmits IB full to the common channel and to CPU.



INTERRUPT CODE (The interrupt code indicates the cause of the interrupt)

0000 0000 Program Controlled Interrupt Channel End
0000 0001 Program Controlled Interrupt
0000 0010 Device End
0000 0011 Channel End

FIGURE 18. INTERRUPT BUFFER

If channel 0 is not masked, interrupt buffer full causes common channel to condition branch on channel interrupt (BCHI) to CPU (Figure 16). During the next I-fetch, CPU responds to BCHI by raising CECC B and CC Emit 1 to allow common channel to proceed with the interrupt and by entering a count-down loop to wait for a reply. Common channel unconditionally sets stat 3 to break the CPU count-down loop.

With the IB full latch set, stat 3 causes CPU to branch to routines to process the multiplexor channel interrupt and form the channel status word. While processing the interrupt, CPU unloads the interrupt buffer and resets the IB full latch without disturbing the multiplexor channel operation.

I/O BREAK-IN

- The break-in operation switches ROSDR decoding from CPU mode to I/O mode.
- A break-out cycle switches ROSDR coding from I/O mode to CPU mode.

An I/O break-in operation forces a branch from the normal CPU flow of microinstructions to a new set of microinstructions called an I/O routine. If ROS is not required to provide further service to a channel, a break-out cycle in the operation returns ROS to the flow of CPU microinstructions at the conclusion of the I/O microprogram routine.

I/O microprogram routines establish and control communication between the channel and other parts of the 2050 processing unit. To begin a ROS I/O routine, the channel causes a break-in operation if ROS is currently executing a CPU microinstruction. Normally, a break-in operation occurs at the end of a machine cycle if main storage is not in the first memory read cycle. The break-in operation consists of a priority cycle and a break-in cycle before the ROS routine begins and a break-out cycle at the end of the routine (Example 1, Figure 19).

Priority Cycle

While executing CPU microinstructions, CPU examines the state of the routine request line from common channel. If, at the end of the current machine cycle, memory is not in the first memory read cycle, routine request causes CPU to condition routine received to common channel, indicating that the next cycle will be a break-in cycle. The cycle during which CPU conditions routine received is the priority cycle (Figure 20). At the end of the priority cycle, ROSDR decoding switches from CPU mode to I/O mode.

Break-In Cycle

During the break-in cycle, these actions occur:

1. Routine received gates the starting address of the I/O routine to ROAR. The routine request buffer and/or common channel generates this address to designate the starting ROS address of the routine to be performed.
2. The location of the next CPU instruction is stored in the ROAR backup register during each CPU cycle. While ROS is performing the I/O routine, the input to the ROAR backup register is blocked to preserve the address of the next CPU microinstruction. This allows ROS to proceed in the CPU flow of microinstructions from the point of interruption after the I/O routine(s) is complete.
3. Routine received prevents the ROS word addressed during the priority cycle from setting the ROS data register; 1 bits in the ROS data register switch to 0 bits.
4. When the ROS data register contains all 0 bits and it is in I/O mode, its output is decoded to store the contents of the R register in the R backup position of local storage. The R register is free for use in executing the I/O routine.

Break-Out Cycle

The last microinstruction in all I/O routines addresses the same microinstruction. The microinstruction common to all I/O routines is executed during the break-out cycle, causing the contents of the R backup register in local storage to transfer to the R register.

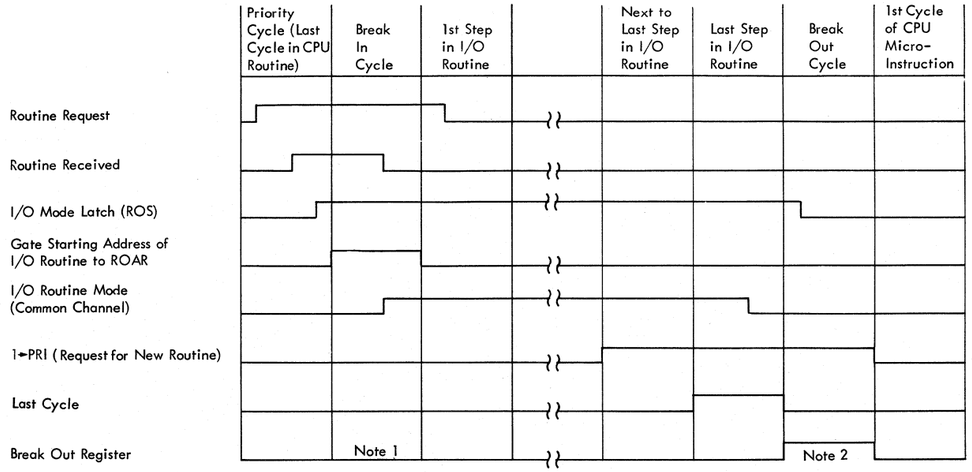
In routine chaining, a break-in cycle may follow a break-out cycle; otherwise, ROS executes a CPU microinstruction after the break-out cycle. If I/O routine chaining is not indicated, the common microinstruction executed during the break-out cycle also causes ROAR to be loaded with the address of the next CPU microinstruction and ROSDR decoding to change from I/O mode to CPU mode.

I/O Routine Chaining

I/O routines are limited to six microinstructions to prevent one channel from monopolizing CPU facilities for an extended period, causing overruns on waiting channels. This six-microinstruction limitation permits high data rates on all channels by insuring that all requests from channels are considered in common channel priority circuits without long delays.

During the last three cycles of each I/O routine (including the break-out cycle) ROS usually generates a 1→PRI signal to allow a channel to request a new routine. If the routine request line from the common channel is conditioned at this time, ROS executes another I/O routine before returning to CPU mode.

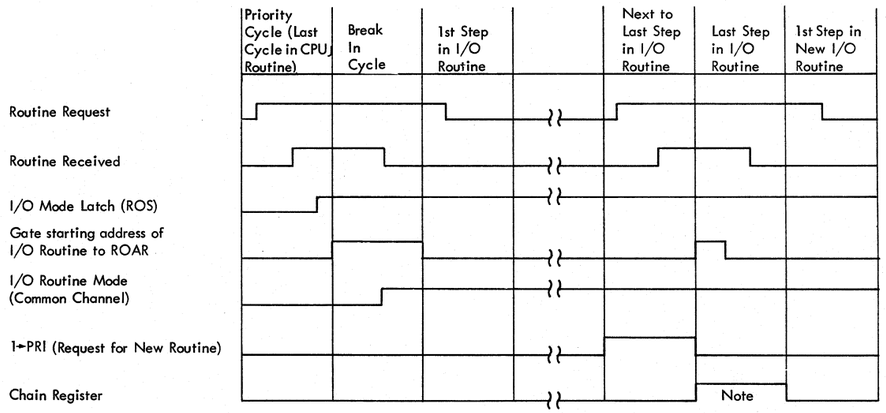
Examples 2, 3, and 4 in Figure 19 show the different possibilities in routine chaining. Figure 20 charts actions to initiate I/O microprogram routines.



Note 1: a) Load ROAR with address generated in channel; b) Transfer contents of R register to R backup register in local storage; c) Block ROSDR.

Note 2: a) Load ROAR with address of next CPU microinstruction. This address is stored in ROAR buffers; b) Transfer contents of R backup register in local storage to R register.

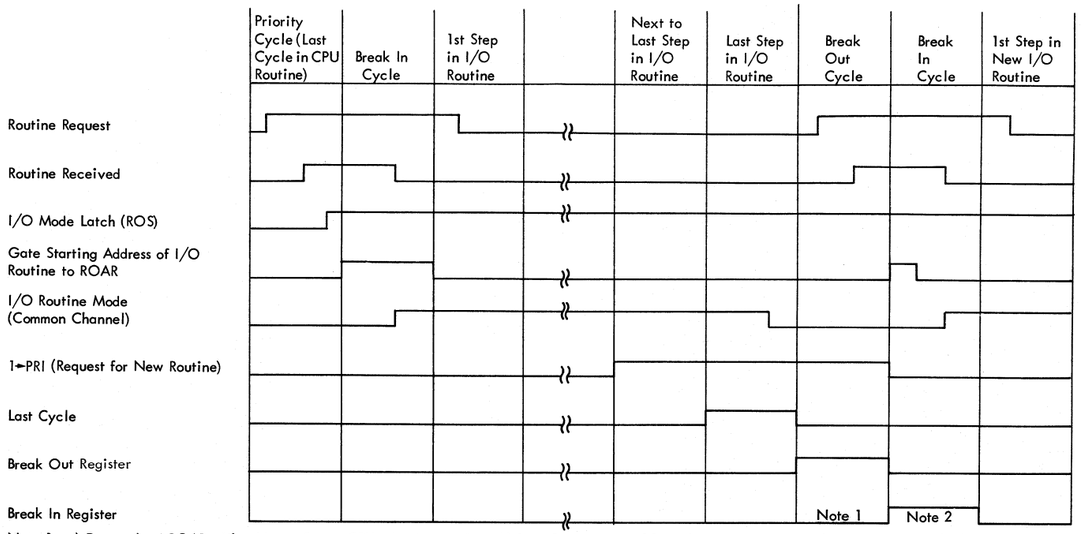
Example 1. Normal Break In - Break Out



Note: Load ROAR with address generated in channel. This address identifies starting position of new I/O routine.

Example 2. Normal Routine Chaining

FIGURE 19. BREAK-IN OPERATIONS (SHEET 1 of 2)



Note 1: a) Do not load ROAR with address of next CPU microinstruction; b) Transfer contents of R backup register to R registers; register; c) Block ROSDR.

Note 2: a) Load ROAR with address generated in channel; b) Transfer contents of R register to R backup register.

Example 3. Routine Chaining with Two Cycle Gap

Note: a) Do not load ROAR with address of next CPU microinstruction; b) Do not transfer contents of R backup register to R register; c) Load ROAR with address generated in channel.

Example 4. Routine Chaining with One Cycle Gap

FIGURE 19. BREAK-IN OPERATIONS (SHEET 2 of 2)

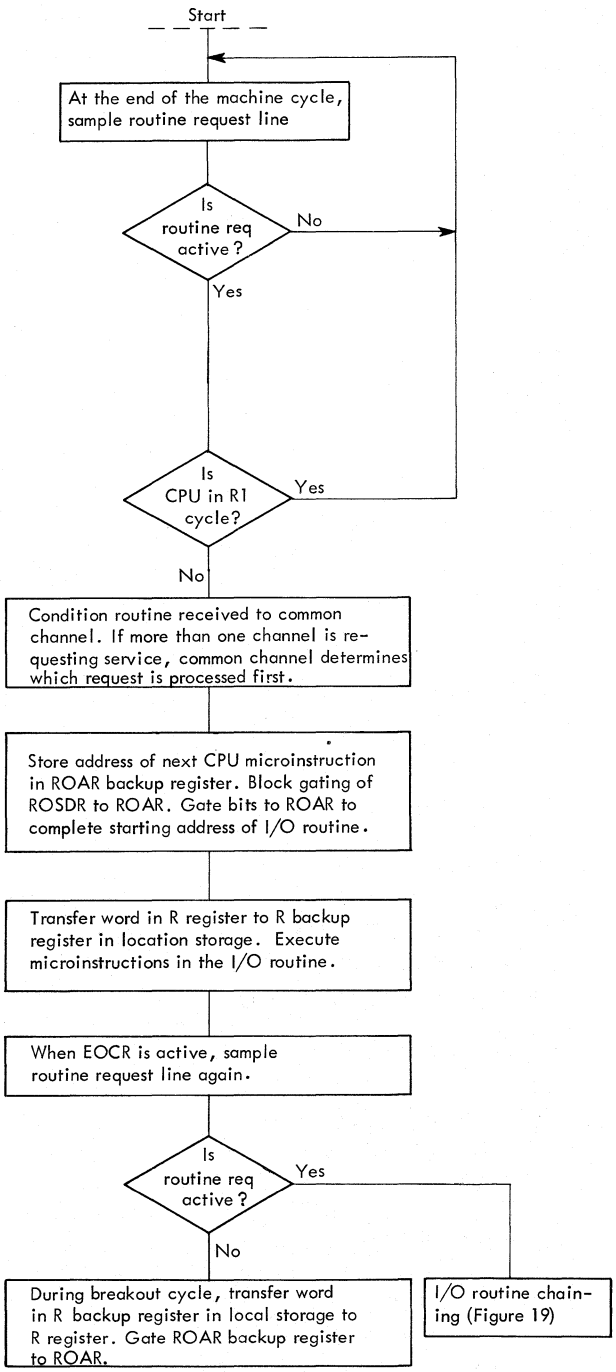


FIGURE 20. INITIATE CHANNEL MICROPROGRAM ROUTINE

COMMON CHANNEL BUFFER LATCHES

- Common channel contains twelve buffer latches, three for each channel.
- While ROS is performing an I/O routine, the buffer latches identify the channel being serviced and the channel serviced in the preceeding routine.

The channel 0 buffer 1 latch gates channel 0 I/O stat settings to CPU to cause branching in the channel 0 microprograms.

In routine chaining, the CPU can detect an error in a routine after that routine is complete and the next routine has begun. The channel uses the buffer 2 and buffer 3 latches to identify the channel that is to receive the error indication. To illustrate this use of buffer 2 and buffer 3 latches:

Assume that channel 0, channel 1 and channel 2 have requested service.

Further assume that ROS has completed the microprogram to service channel 0 and has begun a microprogram to service channel 1. The channel 0 buffer 2 latch is reset during the first cycle of the channel 1 routine; the channel 0 buffer 3 latch is not reset until ROS recognizes the request to begin the channel 2 routine. If, after the channel 1 routine begins, the CPU detects an error associated with the channel 0 routine, either the channel 0 buffer 2 or buffer 3 latch indicates that channel 0 was the previous channel serviced and causes channel 0 to receive the error indication.

Figure 21 shows channel 0 buffer latches; buffer latches for other channels are the same. Figure 22 shows buffer latches action during the execution of I/O routines.

LOCAL STORAGE

I/O Channel Status Area

- I/O channel status area consists of sixteen 36-bit locations and is divided into four groups, one for each channel.
- The multiplexor channel group has four word locations: command address, data address, count, and unit address.
- Local storage word positions have six-bit addresses.

A sector in local storage, called the I/O channel status area, is reserved exclusively for I/O operations. The sector consists of sixteen 36-bit locations (including four parity bits); the sector is divided into four groups, one for each of the four channels attached to the Model 50 system. All transfers to or from local storage are under Read Only Storage (ROS) control.

Local storage word positions have six-bit addresses. The two high-order bits are both zeros to designate the channel sector 00. Bits in positions 3 and 4 identify one of four channels; the common channel inserts bits in these address

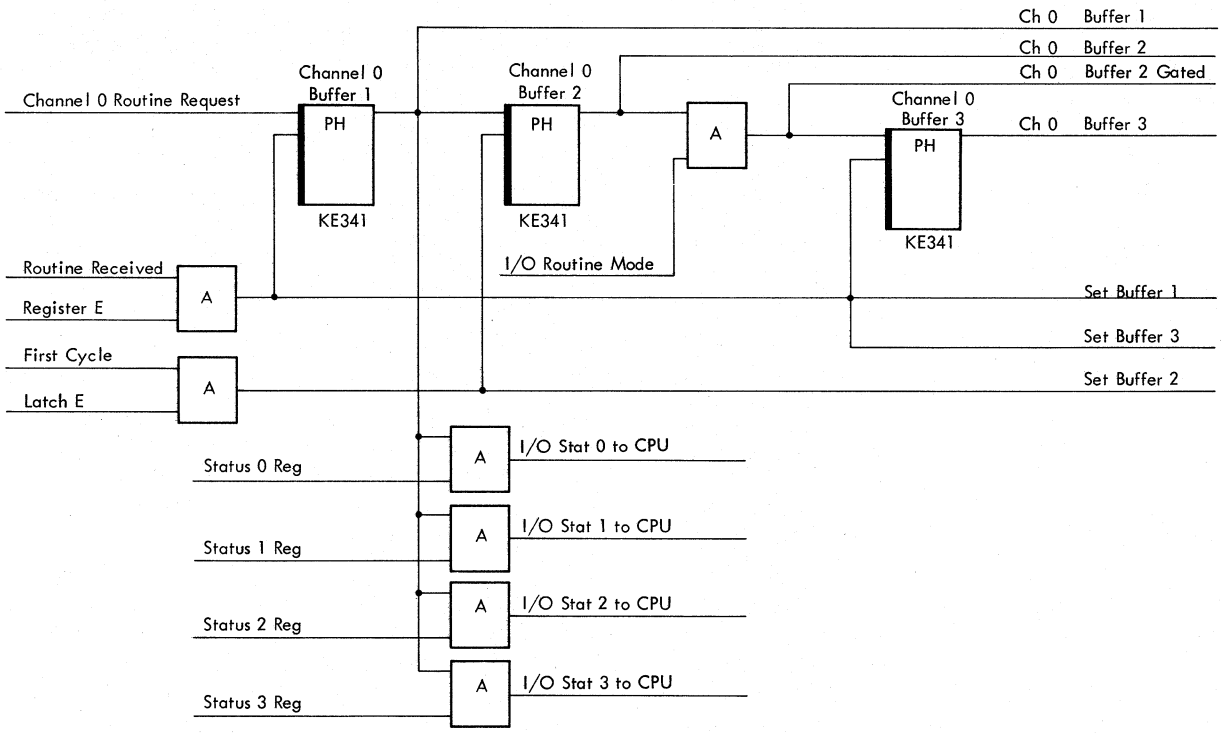


FIGURE 21. CHANNEL 0 BUFFER LATCHES

positions. The two low-order bits in the address select one of four word locations assigned to the channel.

Figure 23 shows a layout of the I/O channel status area of local storage. Figure 24 shows contents of the four multiplexor channel positions after channel routines process CCW-1 and CCW-2.

When the channel requires a new CCW to initiate execution or to continue an I/O operation, ROS routines assemble control information in multiplexor channel positions of local storage. If the channel executes an I/O operation in byte mode, data in local storage transfer to bump storage when the I/O unit releases the channel by allowing operational in to fall. Although bump is the permanent storage area for control information in byte mode operations, control data cannot direct the operation from bump storage locations. Therefore, channel routines process control information between bump and local storage at selected intervals throughout the operation.

In burst mode operations, local storage is the permanent storage area for control data. In performing the operation, ROS routines process and update control information in local storage positions. Because the I/O unit does not transmit its unit address prior to conditioning interface tag lines in burst mode, the channel cannot use bump storage facilities. An I/O unit identifies a burst mode operation by holding operational in active, blocking the transfer of control data from local storage to bump storage. If the unit allows operational in to fall prior to the end status byte transfer, the channel reverts to byte mode operation. The unit must then transmit its unit address and wait for the command out response from the channel before conditioning the service in or status in tag line.

I/O Common Channel Area

- During break-in, CPU transfers the contents of the R register to the R backup register in local storage.
- I/O common channel area of local storage consists of eight 36-bit word positions, half of sector 2 (Figure 23).

While executing instructions in the customer program, ROS decodes ROSDR in CPU mode. ROS must function in I/O mode to decode I/O micro-orders to service the multiplexor channel. Actions required to cause ROS to switch from CPU mode to I/O mode are called break-in; actions to change ROS from I/O mode to CPU mode are called "break-out." When break-in occurs, CPU registers contain information associated with the CPU operation. The I/O routine must not destroy this information.

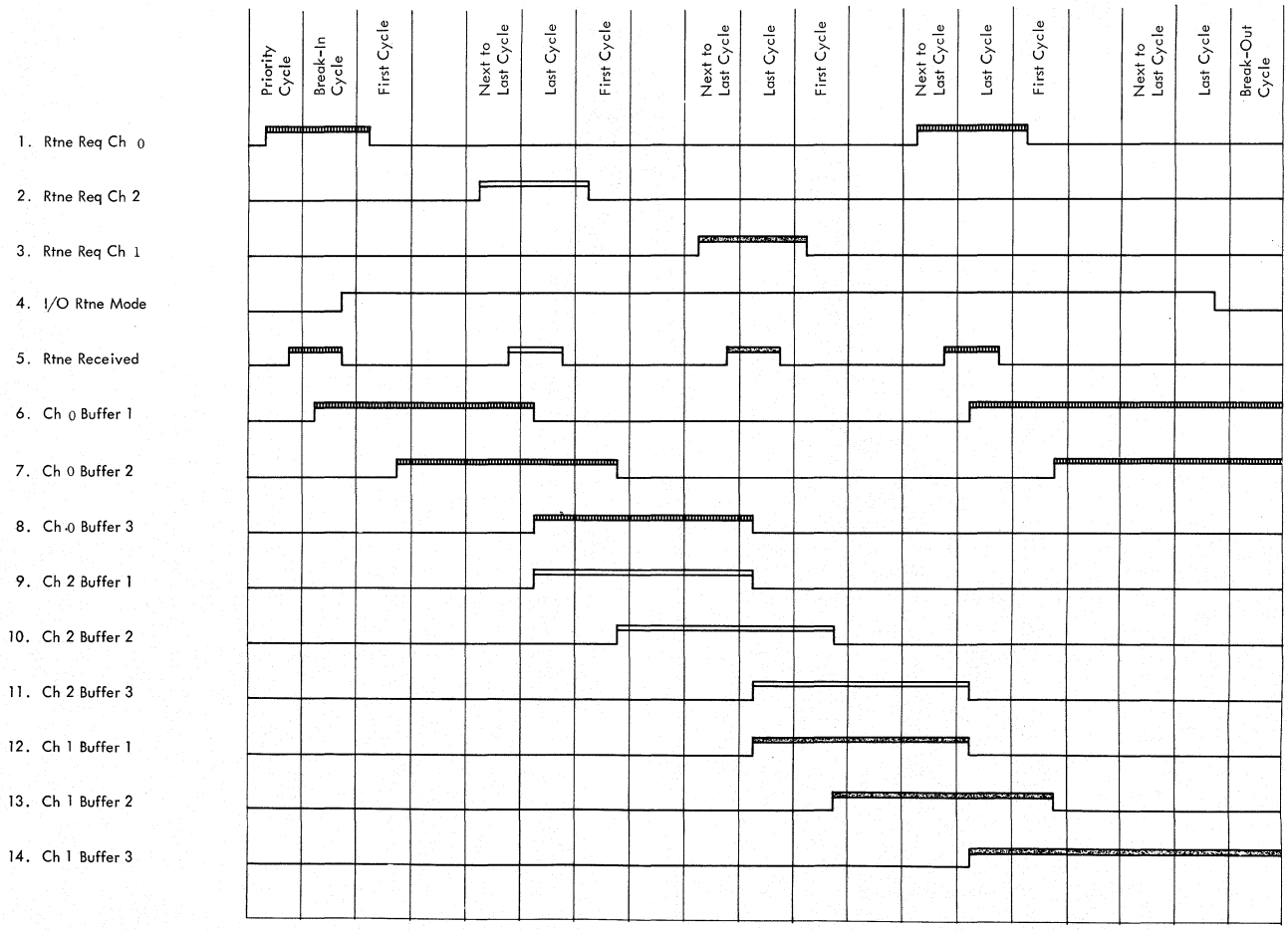


FIGURE 22. BUFFER LATCHES SEQUENCING

During break-in, ROS transfers the contents of the R register to the R backup position in the I/O common channel area of local storage. This transfer frees the R register for use in performing the channel routine. In the break-out, ROS returns the contents of the R backup position to the R register. Thus, the data associated with the CPU operation are preserved during the I/O routine.

The I/O common channel area, half of sector 2 of local storage, contains eight 36-bit word positions (including four parity bits). Four of the word positions are not used. One R register backup position services all channels. Two other positions in the I/O common channel area, interrupt buffer and backup buffer 3, are reserved for multiplexor channel use. The CPU uses the contents of the interrupt buffer in executing multiplexor channel interrupts. The channel uses backup buffer 3 in performing command chaining operations.

POLL CONTROL LATCH

- The poll control latch must be set to allow the channel to process new requests for service.

The channel can control concurrent multiplex mode operations; however, the channel services I/O units serially. At any given time, only one I/O operation can use channel facilities. When an I/O unit raises operational in, the channel cannot recognize a request from either common channel or another I/O unit until ROS has performed a series of routines to service the unit holding operational in active. Conversely, when the channel accepts a signal from the common channel to begin an operation, the channel cannot service an I/O unit until it has initiated or rejected the new operation. The channel requires ROS to perform a series of routines to service requests from common channel or

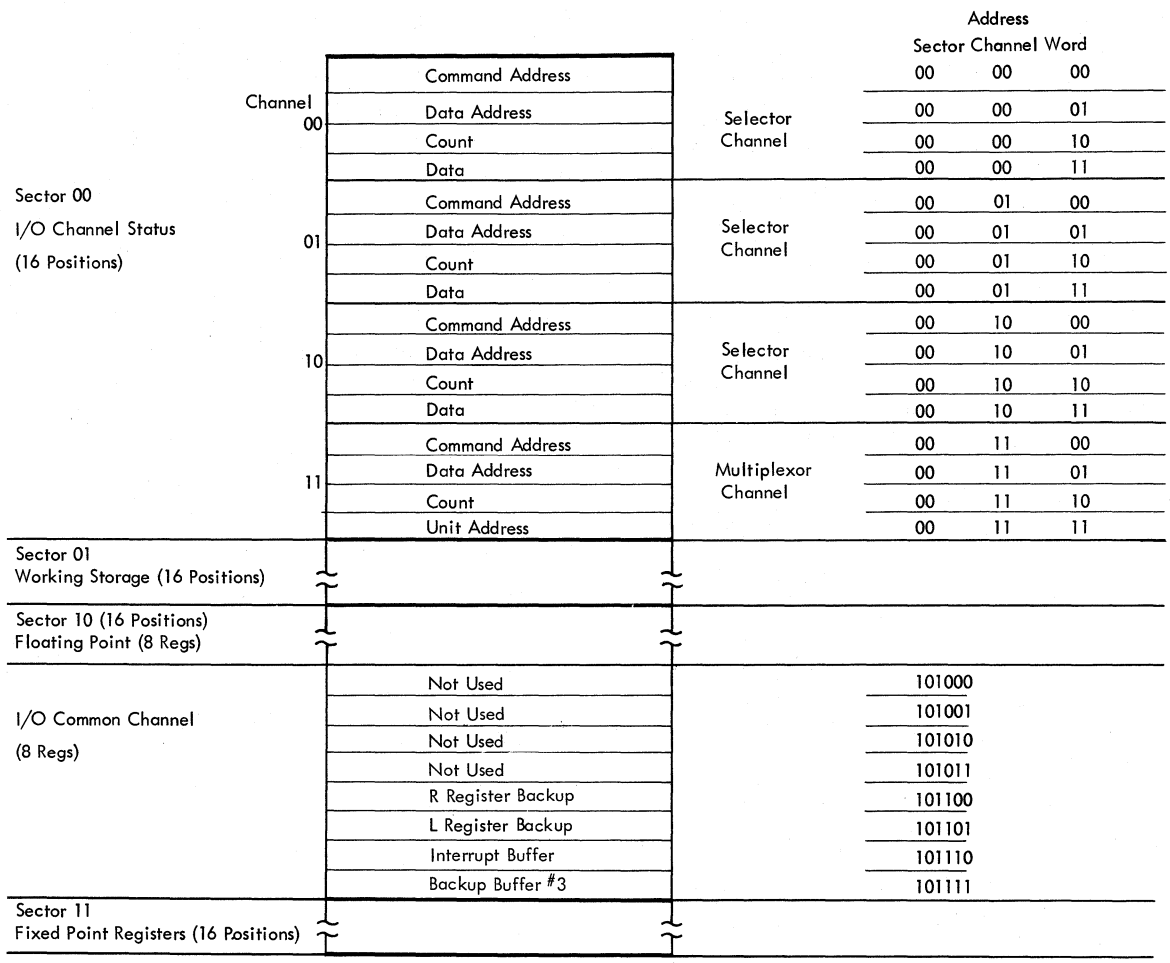


FIGURE 23. LOCAL STORAGE

an I/O unit; a different operation cannot cause channel to terminate the series of routines before completion.

The ON state of the poll control latch defines the period during which the channel can initiate actions to service an I/O unit or the common channel. Not poll control prevents new requests for service from initiating any channel action to process the request. Figure 25 shows the poll control latch set and reset circuits.

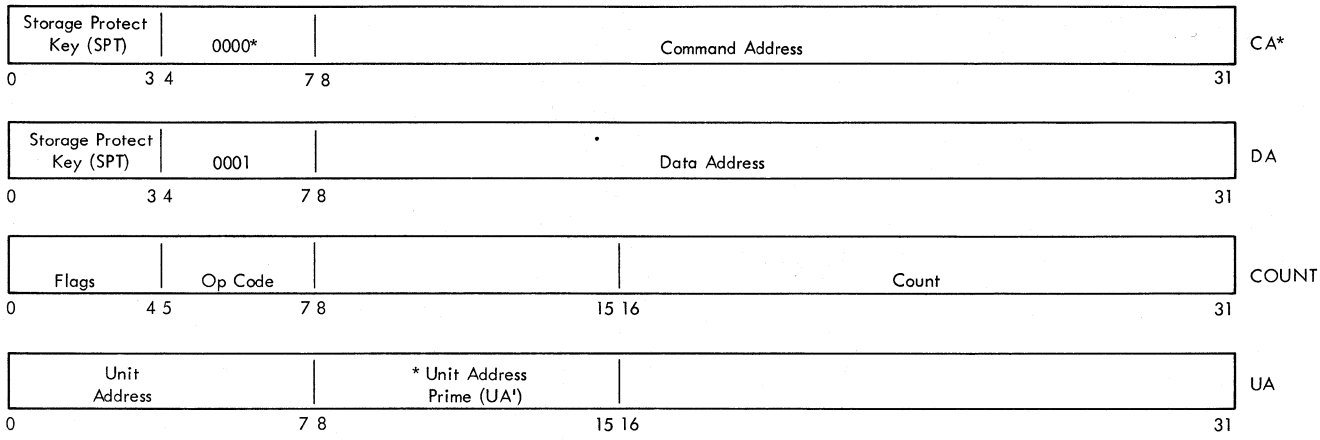
BUMP STORAGE

- Bump storage is an addition to main storage that provides the facility for the multiplexor channel to store control words.

To direct concurrent I/O operations, the multiplexor channel must store the four control words associated with each operation in process. To provide an accessible storage area for multiplexor channel control data, the capacity of main storage is increased. This addition to main storage, called bump storage, is reserved exclusively for multiplexor subchannel control words. Normally, bump storage is accessible to the CPU program only when CPU processes a channel interrupt.

Four bump storage word locations are assigned to each multiplexor subchannel. Figure 26 shows the format of each subchannel word. A multiplexor channel to which 256 subchannels are attached requires a bump storage of 1024 words (4096 bytes). Other bump storage sizes are 512 words (2048 bytes), and 256 words (1024 bytes).

Channel routines fetch appropriate bump storage words to local storage to service an I/O unit. If the unit control word (UCW) store trigger is set when



* Bit Positions 4-7 in the word in the CA position of local storage contain zeros to prevent changing storage protect bit while modifying the command address

FIGURE 24. CONTENTS OF MULTIPLEXOR CHANNEL LOCAL STORAGE POSITIONS AFTER INITIAL SELECTION

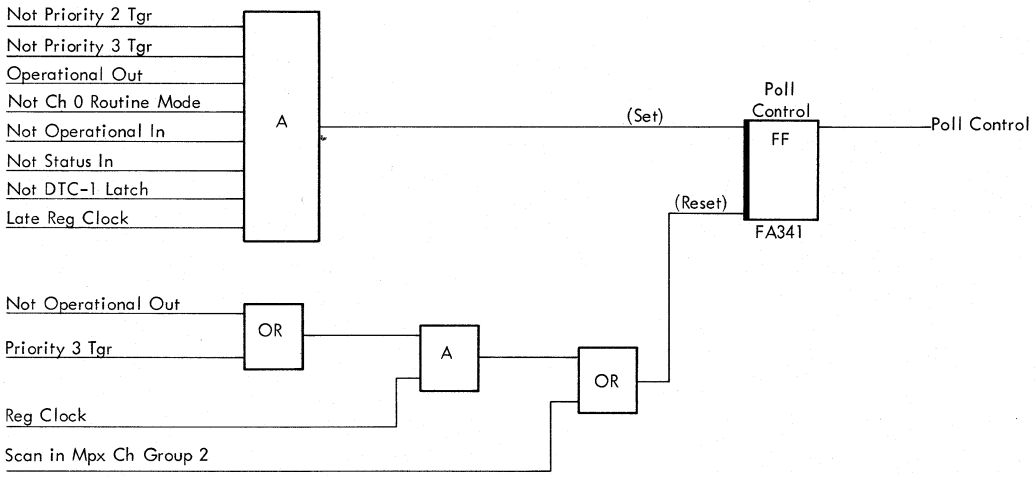
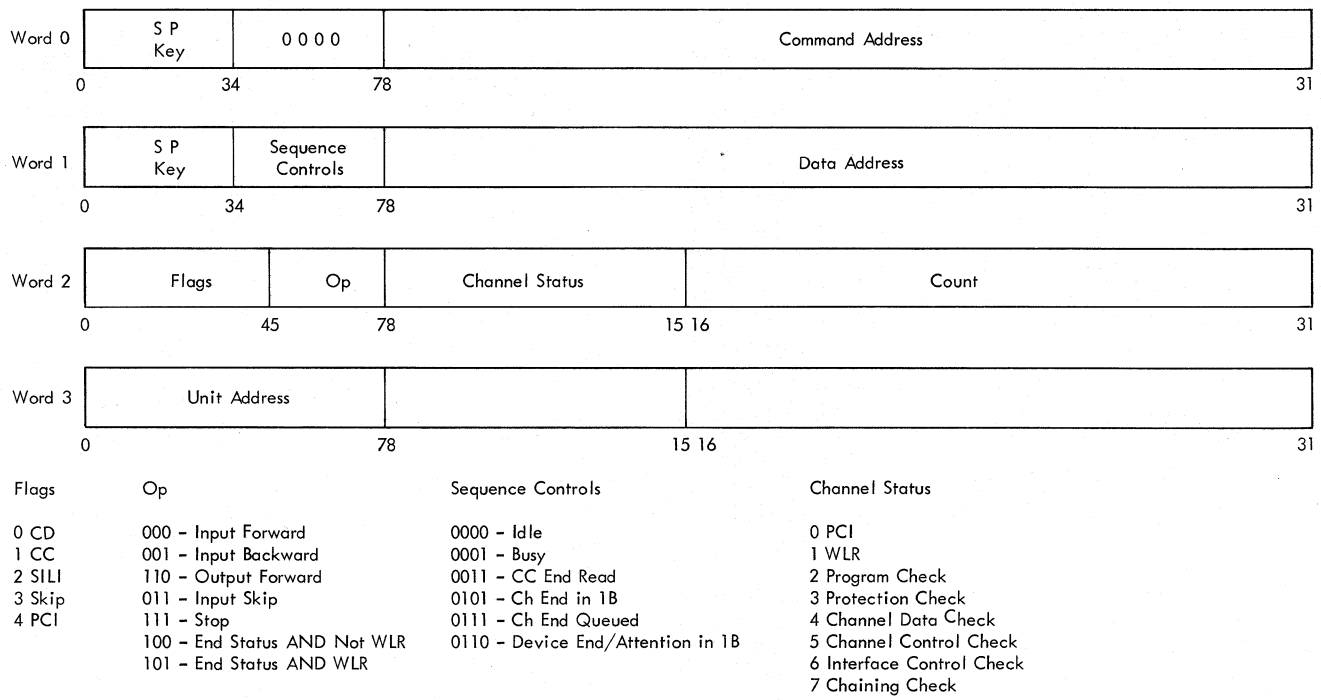


FIGURE 25. POLL CONTROL TRIGGER



● FIGURE 26. FORMAT OF SUBCHANNEL WORDS IN BUMP STORAGE

the unit releases the channel, channel routines return updated control words to bump storage locations reserved for that subchannel.

To address bump storage, a channel routine transfers to the mover the address of the unit being serviced and uses either the W,E→A or the W,E→A(BMP)S micro-order to:

1. Gate mover positions 0-1 (the two high-order bits of the unit address) to SAR positions 14 and 15
2. Gate mover positions 2-7 (the six low-order bits of the unit address) to SAR positions 24-29
3. Gate positions 2 and 3 of the emit field of SAR positions 16 and 17 to select word 00, 01, 10, or 11 in the reserved subchannel area of bump storage
4. Set the SAR B latch to indicate a bump storage address

The addressed bump storage word sets the SDR.

MISCELLANEOUS LATCHES

Burst Mode Latch

- The channel sets the burst mode latch to indicate that a burst mode operation is in process or that the halt I/O command has been executed.

An I/O instruction in the customer program causes CPU to:

1. Store the unit and channel addresses in the L register
2. Form control words if the channel requires control data to execute the instruction
3. Transmit a command to the common channel
4. Enter a countdown loop to wait for a reply.

If the channel address in the L register specifies channel 0, the common channel relays the CPU command to the multiplexor channel. The multiplexor channel does not respond if the command is halt I/O, or if the command is not halt I/O and a burst mode operation is currently in process. With no reply, CPU steps through the countdown loop, then causes common channel to transmit time out to the multiplexor channel. Time out sets the burst mode latch (Figure 27).

If the command was not halt I/O, burst mode raises condition code 2, to indicate to CPU that a previously initiated burst mode operation prohibited channel action on the new command. If the command was halt I/O, burst mode raises condition code 2 when the interface disconnect latch is set, indicating that the halt I/O command has been executed.

The burst mode latch is reset when the I/O unit allows the channel to return to poll status. While the burst mode latch is set, the channel unconditionally responds to all commands from common channel except halt I/O with condition code 2.

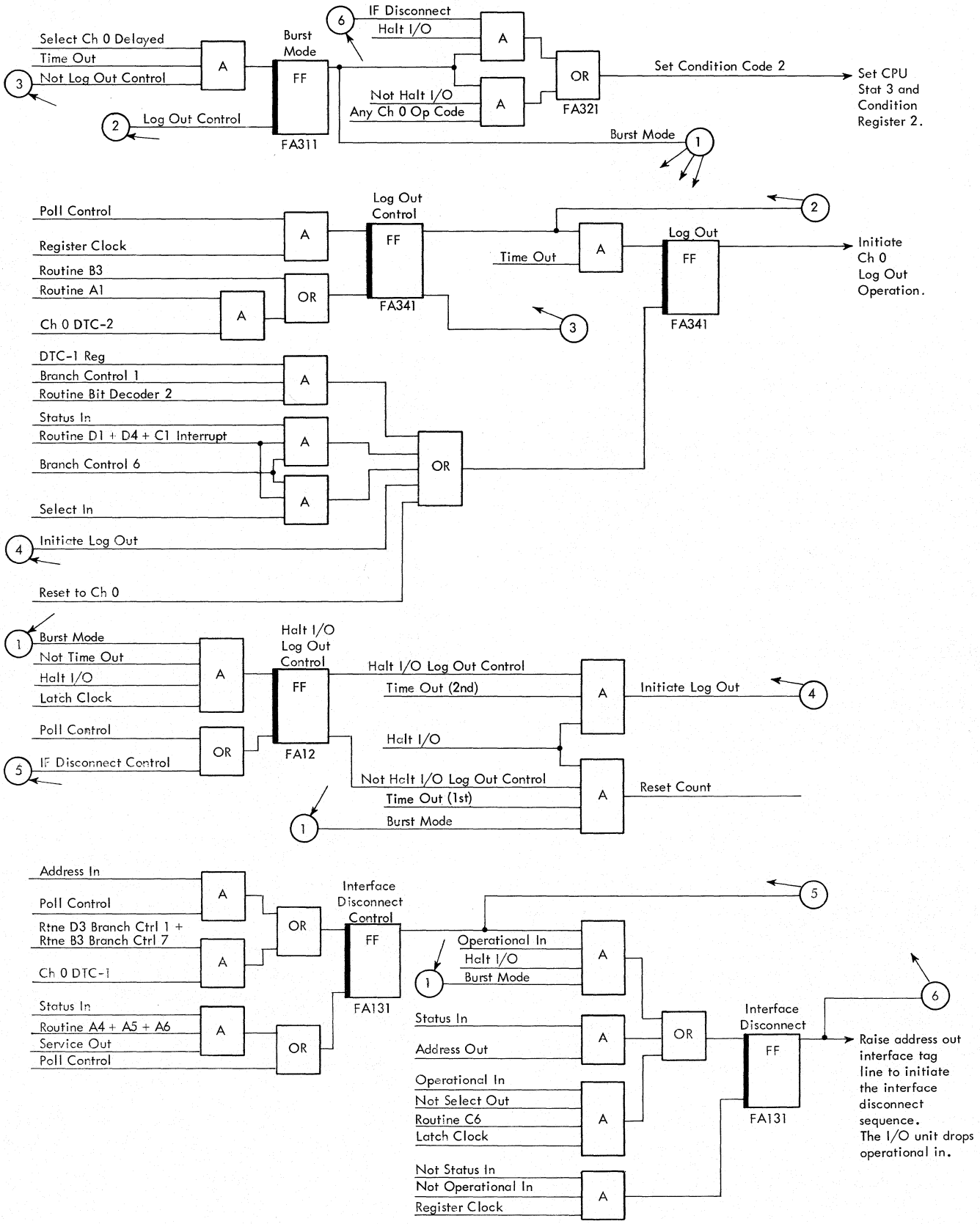


FIGURE 27. MISCELLANEOUS LATCHES

Log Out Control Latch

- Conditions other than a burst mode operation in process can prohibit the channel from replying to a CPU command; the log out control latch permits the channel to transmit the proper response to a time out signal.

The on output from the poll control latch sets the log out control latch. With the log out control latch set, CPU causes common channel to transmit time out to the multiplexor channel only when a failure blocks the channel response to a CPU command. Log out control resets the burst mode latch and causes a subsequent time out to set the log out latch, indicating a failure, rather than allow time out to set the burst mode latch (Figure 27). The on output from the log out latch initiates a channel 0 log out operation.

The channel resets the log out control latch while ROS is executing either routine A1 or routine B3. The channel requests routine A1 in processing a request for service from an I/O unit. In executing a command from common channel, the channel either causes ROS to perform routine B3 or common channel to break the countdown loop before CPU issues time out.

Interface Disconnect Control Latch

The interface disconnect control latch allows halt I/O unconditionally to reset the I/O unit in communication with the channel when the burst mode latch is set (Figure 27). The channel sets the interface disconnect control latch:

1. When an I/O unit raises address in to initiate an interface sequence to transfer data or a status byte
2. If branch control 7 is set while ROS is executing the routine (B3) to process the status byte in the initial selection sequence
3. If branch control 1 is set while ROS is executing the routine (D3) to analyze the initial status byte in a chain command operation

In all cases, the interface disconnect control latch is set throughout the interface sequence to transfer data bytes between the channel and an I/O unit.

The interface disconnect control latch is reset by either the on output from the poll control latch or by the service out response status in, while ROS is performing routine A4, A5, or A6.

Interface Disconnect Latch

The interface disconnect latch initiates the interface disconnect sequence by raising address out (Figure 27). The channel sets the interface disconnect latch:

1. If the poll status latch is set when the channel receives halt I/O
2. If the interface disconnect control and burst mode latches are set while the channel is processing the halt I/O command
3. If, in the initial selection sequence, the I/O unit answers address out with status in

The I/O unit responds to the interface disconnect by dropping all interface signals to the channel. When operational in falls, the channel resets the interface disconnect latch, dropping address out to complete the disconnect sequence.

Halt I/O Log Out Control Latch

After causing common channel to issue time out to the multiplexor channel, CPU enters a second countdown loop to wait for a reply. If CPU has requested the halt I/O operation, time out sets the burst mode latch and resets the count. When time out drops, the channel sets the halt I/O log out control latch (Figure 27).

After stepping through the second countdown loop, CPU causes common channel to transmit another time out signal to the channel. The second time out initiates a channel 0 log out operation if the halt I/O log out control latch is still set. To prevent the channel 0 log out operation, the multiplexor channel must:

1. Initiate the interface disconnect sequence and transmit the condition code 2 reply to block the transmission of the second time out or
2. Set the interface disconnect control latch to reset the halt I/O log out control latch before common channel transmits the second time out.

The following section describes I/O operations on the System/360 Model 50 Multiplexor Channel. Included are:

1. Handling of I/O instructions in common and multiplexor channels
2. Descriptions of multiplexor channel microprograms (routines) covering initial selection, data handling, end status analysis, test I/O, halt I/O, interrupt test I/O, and test channel.

Descriptions of I/O instructions begin when the CPU enters the countdown loop. CPU handling of I/O instructions is described in System/360 Model 50 Operation Codes RS, SI, SS, Field Engineering Manual of Instruction, Form Y22-2825.

I/O INSTRUCTIONS

The IBM System/360 Model 50 recognizes four I/O instructions: Start I/O, Test I/O, Test Channel, and Halt I/O. I/O instructions initiate all channel operations except interrupts and can be categorized in the following manner:

1. The Test I/O, Test Channel, and Halt I/O instructions do not initiate operations on an I/O unit. To execute these instructions, neither the channel nor the I/O unit need transfer data bytes across the interface.
2. The Test Channel instruction determines whether the channel is in the busy or not busy state. Because the Test Channel instruction does not require the channel to process data, execution of the instruction does not disturb an operation currently in process on the selected channel.

Execution of the Start I/O instruction causes the specified channel and I/O unit to change from the not busy to the busy state. Only the Start I/O instruction can initiate a read, read backward, write, sense, or control operation. The control operation may not require data byte transfers across the interface.

CPU Actions in Initiating an I/O Operation

- The I/O instruction designates the channel and unit address.
- If the instruction is Start I/O, CPU loads the M and R registers with the CCW-1 and CAW.
- After signaling the common channel, CPU enters a countdown loop to wait for a reply.

Each I/O instruction is in the SI format and supplies information to identify a channel and an I/O unit. After decoding an I/O instruction, the CPU:

1. Loads the L register with bits to designate the channel and unit.
2. Uses WCC or CECC micro-orders to transfer bits, to common channel, that specify the instruction.
3. Makes available to common channel bits in L register positions 21-23 to select channel 0, 1, 2, 3, or 4.

If the instruction is Start I/O, in addition to the foregoing the CPU must: (1) fetch the channel address word (CAW) from main storage location 72, and store it in the R register; (2) fetch the first half of the channel command word (CCW-1) from the location that the CAW designates and store the CCW-1 in the M register. No CAW or CCW is associated with the test I/O, halt I/O, or test channel instructions.

Before proceeding to execute the next instruction in the customer program, the CPU must receive a signal from the channel, indicating whether the I/O instruction has been accepted. After signaling the common channel, CPU enters countdown loop to wait for a reply.

Channel Actions in Initiating an I/O Operation

- Common channel and the multiplexor channel test to determine if the channel is free to perform an operation.
- Either a multiplexor channel microprogram or the common channel sets CPU stats and a condition code to indicate the action that the I/O instruction causes.
- CPU stat 3 is set to allow the CPU to break from the countdown loop.

The common channel relays the request from CPU to the channel. The channel:

1. Makes logic tests and transfers results of these tests to the common channel or, in some cases, transfers status information to CPU registers if the instruction is start I/O, test I/O, or halt I/O. When this occurs, the channel does not indicate the response to the start I/O or test I/O instruction to the common channel.
2. Initiates execution of the test I/O, start I/O, or halt I/O instruction if no operation is in process and the channel is idle.
3. Terminates the current I/O operation if conditions permit and the instruction is halt I/O.

The common channel accepts signals from the selected channel and sets CPU stats and a condition code. When the channel transfers status information to CPU registers, a channel microprogram sets CPU stats. CPU stats and condition code settings

indicate to CPU the selected channel response to the I/O instruction. In all channel replies, CPU stat 3 is set to allow the CPU to break from the countdown loop. Stats 0, 1, and 2 and condition code settings indicate specific channel actions that the I/O instruction caused. The following example shows a use of CPU stats and condition codes.

The common channel examines the three bits from the L register that select a channel to perform the I/O operation. Only channels 0, 1, 2, 3, and 4 are available. If the bits designate channel 5, 6, or 7, the common channel sets: (1) condition code 3 to indicate unavailable reject, and (2) CPU stat 3 to allow the CPU to break from the countdown loop. Because neither stat 1 nor stat 2 is set, the CPU returns to I-fetch to process the next instruction in the customer program.

When the channel is busy and cannot perform the I/O instruction before a time out, CPU issues time out to the channel. Time out sets the channel burst mode latch; the burst mode latch permits common channel to set CPU stat 3 and condition code 2. Common channel's reply permits CPU to return to I-fetch.

Maintenance diagrams show: (1) CPU requests to common channel to begin an I/O operation, (2) common and multiplexor actions that the CPU request initiates, and (3) channel responses to the CPU.

Common channel sets condition code 3 and CPU stat 3 when the multiplexor channel sequence controls fetch routine detects an invalid bump address. Start I/O, foul on start I/O, test I/O, interrupt test I/O, and multiplex mode halt I/O operations begin with the sequence control fetch routine; therefore, an invalid bump address reply, condition code 3 and CPU stat 3, is possible for any of these operations if an invalid bump address is detected.

Start I/O

- If an error is found in the CAW, the CPU gates foul on start I/O to the common channel rather than start I/O.
- Foul on start I/O causes the common channel to indicate to CPU whether the selected subchannel was free to perform an operation had the CAW not contained an error.
- If no error occurs and the selected channel is in poll status, the channel processes CCW-1 and CCW-2, and performs the unit select operation before setting CPU stat 3.
- If an error occurs after common channel receives start I/O, CPU stat 3 is set during execution of the interrupt preparation routine.

After decoding the start I/O instruction, the CPU loads the L register with bits identifying an I/O unit and a channel, and fetches the CAW from main storage location 72. Bits in positions 8-31 of the CAW must contain a valid main storage address, and bits in positions 4-7 and 29-31 of the CAW must be zeros. If no error is detected in the CAW, the CPU loads the M and R registers, gates start I/O to the common channel rather than foul on start I/O. After transmitting either start I/O or foul on start I/O, the CPU enters a countdown loop to wait for the CPU stat 3 reply.

Foul on Start I/O

Foul on start I/O alerts the common channel that the CPU has detected an error in the CAW. The common channel selects the multiplexor channel and transfers the foul on start I/O to the multiplexor channel.

If the multiplexor channel is in poll status and the subchannel is free to perform an I/O operation, the common channel sets CPU stats 1, 2, and 3. With CPU stat 3 on, the CPU breaks from the countdown loop and examines CPU stats 1 and 2. Because CPU stats 1 and 2 are set, the CPU stores zeros in bit positions 32-41 and 43-47 of the channel status word (CSW), and a 1 bit in the program check (bit 42) position of the CSW. Then, the CPU returns to I-fetch to process the next instruction.

If the multiplexor channel is in poll status and the subchannel is busy, the channel cannot execute the I/O operation. Foul on start I/O causes the common channel to: (1) set condition code 2 indicating the subchannel is busy and not free to execute the start I/O if the CAW was valid, and (2) set CPU stat 3, allowing the CPU to break from the countdown loop. Because neither CPU stat 1 or 2 is set, the CPU returns to I-fetch.

Start I/O to Common Channel

If the CPU does not detect an error in the CAW, the common channel receives start I/O. The common channel selects the multiplexor channel and signals start I/O to the channel.

If the multiplexor channel is in poll status, the channel is currently free of I/O interface requests; therefore, the channel is free to execute start I/O. Start I/O causes the multiplexor channel to request the microprogram routines that execute the operation.

A multiplexor channel routine fetches the unit address from the L register and uses the unit address to address the subchannel in bump storage. If the subchannel has not received device end status from the specified I/O device, the subchannel is busy. Start I/O causes the multiplexor channel to

test the state of the subchannel; if the subchannel is busy, the multiplexor channel cannot execute the start I/O operation. When the subchannel is busy, start I/O causes the common channel to set: (1) condition code 2, indicating busy reject, and (2) CPU stat 3 to permit the CPU to break from the countdown loop and return to I-fetch. If the channel is holding device end status, the status condition is cleared and priority is given to the start I/O operation. When the channel is not busy, the microprogram routines fetch CCW-1 and CCW-2, select the I/O device, and analyze the device status.

Programming Error in the CCW: The first byte in CCW-1 contains the command code. These bits must specify a read, read backward, write, sense, or control operation. If the command code in CCW-1 designates transfer-in-channel (TIC), or contains zeros in positions 4-7, the channel sets the program check latch and requests the load 64 preparation routine. The channel routine transfers channel and unit status information to CPU registers and sets CPU stats 2 and 3. Stat 3 allows CPU to break from the countdown loop. Stat 2 causes a CPU routine to replace parts of the CSW with information stored in CPU registers during the load 64 preparation routine.

If no programming error is detected in CCW-1, the channel routines test for a programming error in CCW-2. If the flags in bit positions 37 - 39 do not equal 0, or the data count in bit positions 48 - 63 equals 0, the channel sets the program check latch. A programming error in CCW-2 causes the channel to request the load 64 preparation routine. The results of executing the load 64 preparation routine are the same as those described for a CCW-1 programming error.

Unit Select: If no programming error is detected in the channel command word, the channel selects the unit after the CCW-2 fetch is complete. The channel begins the selection by raising address out, transmitting the unit address on bus out lines, and conditioning select out.

If no unit recognizes the address, the unit with the lowest priority propagates select in to the channel. The channel causes the common channel to set condition code 3, indicating unavailable reject, CPU stat 3, to allow the CPU to break from the countdown loop. Because CPU stats 1 and 2 are not set, the CPU returns to I-fetch. If the addressed unit recognizes the address, it transmits its address over bus in lines to the channel after decoding the address on bus out lines. The channel then sends a command byte to the unit to identify the requested operation. The unit accepts the command byte, prepares to execute the operation, and transmits a status byte to the channel at the end of the initial selection sequence.

If the unit is not free to perform an operation, it decodes the address on bus out lines, and responds by transmitting a status byte to the channel rather than an address byte. The status in reply to address out signals the channel that the addressed unit is busy; the channel then executes the control unit busy and load 64 preparation routines. The channel terminates the sequence without transmitting the command byte to the unit.

Regardless of whether the unit is busy or free to perform an operation, it transmits a status byte to the channel at some point in the initial selection sequence. If the unit status byte contains zeros in bit positions 0-7, the channel proceeds to execute the I/O operation and causes the common channel to set condition code 0. The common channel sets condition code 0, indicating operation initiated, and CPU stat 3 to allow the CPU to break from the countdown loop and return to I-fetch.

The channel terminates the initial selection sequence with the 64 preparation routine if the status byte does not contain zeros in positions 0-7 and the following conditions exist:

1. Position 0, 2, 3, 6, or 7 contains a 1 bit
2. Position 4 contains a 0 bit
3. The chain command flag is not set, or the chain command flag is set, position 5 contains a 0 bit, and positions 1 and 4 contain a 1 bit.

During the execution of the load 64 preparation routine, the channel transfers status information to CPU registers and sets CPU stats 2 and 3. Stat 3 allows the CPU to break from the countdown loop. Stat 2 causes the CPU microprogram to replace parts of the CSW with information stored in the CPU registers by control unit busy and load 64 preparation routines.

Test I/O

- Test I/O clears interrupt conditions on the specified unit, if an interrupt is pending.
- Test I/O indicates busy reject, if the subchannel is busy.
- Test I/O determines the status of the specified unit, if the subchannel status permits device selection.

After decoding the test I/O instruction, the CPU:

1. Loads the L register with bits identifying the channel and unit to perform the operation
2. Gates test I/O to common channel and makes L register positions 21-23 available to the common channel

3. Enters the countdown loop to wait for the CPU stat 3 reply. If the channel is in poll status, it is currently free of interface requests; therefore, the channel executes test I/O. Test I/O causes the channel to fetch the subchannel sequence controls; the contents of the sequence controls determine which routines the channel must execute to complete the test I/O operation.

Figure 28 is an example of the routines that perform the test I/O sequence.

Interrupt Test I/O

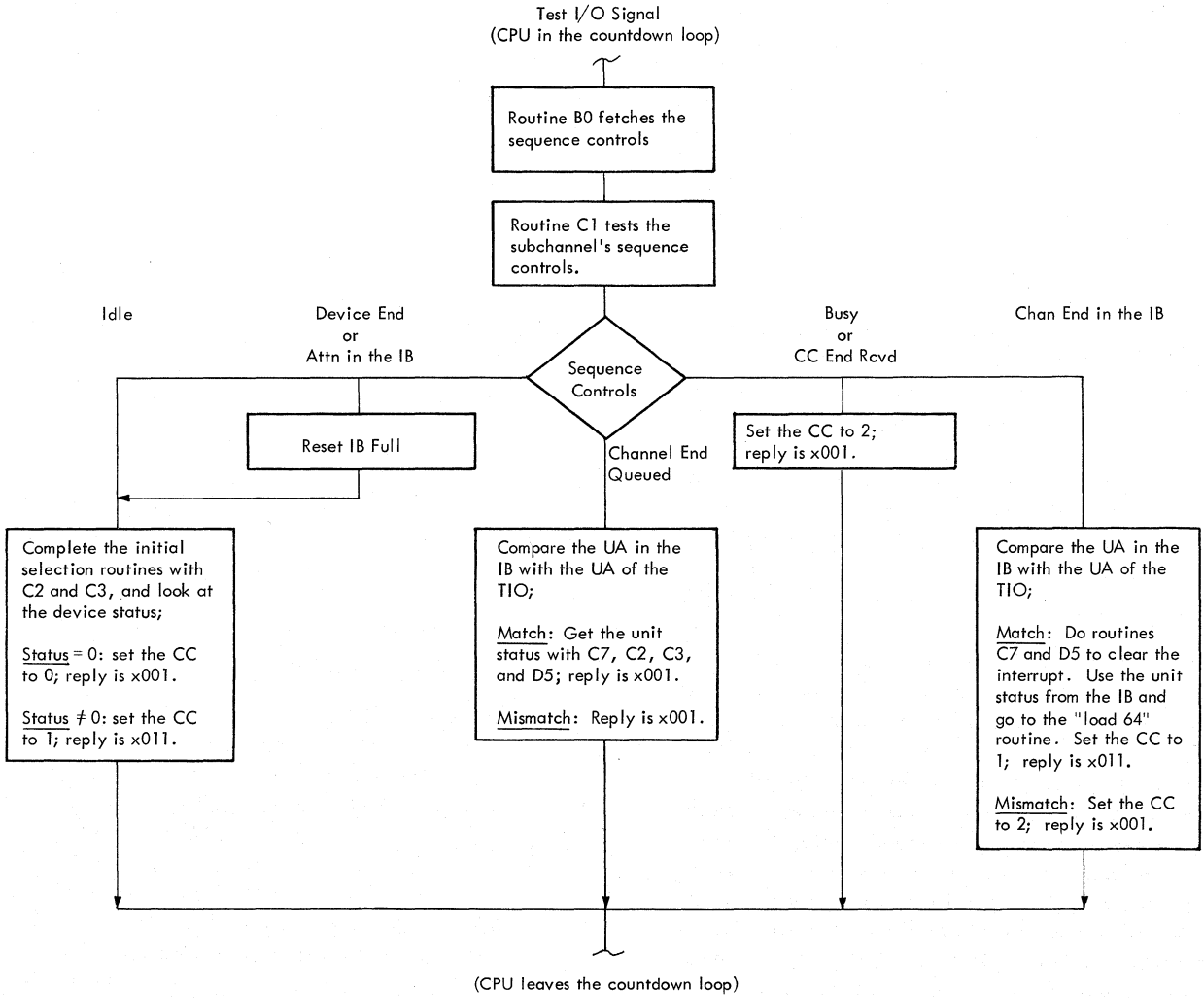
When CPU issues interrupt test I/O to common channel, the channel operation is similar to that required to execute test I/O. A 1 bit in position zero of the CPU buffer out bus and the multiplexor channel select code identify interrupt test I/O to the multiplexor channel. Because interrupt test I/O is not recognized as an I/O instruction by the channel, interrupt test I/O instruction does not set condition codes.

The channel reply to interrupt test I/O is an x001 to CPU that indicates the operation is complete.

Interrupt test I/O only results from a device end interrupt.

Halt I/O

- The halt I/O instruction terminates I/O interface operation if the channel is not in poll status and in burst mode.
- If the channel is in poll status, channel halt I/O routines determine the action necessary to execute the operation.



● FIGURE 28. EXAMPLE OF TEST I/O ROUTINES

After decoding the halt I/O instruction, the CPU:

1. Loads the L register with bits identifying the channel and I/O unit
2. Gates halt I/O to common channel and makes L register positions 21-23 available to the common channel
3. Enters the countdown loop to wait for the stat 3 reply

The common channel transmits halt I/O to the channel. If the multiplexor channel is in poll status, it is currently free of I/O interface requests; therefore, the channel requests the microprogram routines to execute halt I/O. If the multiplexor channel is not in poll status when halt I/O is received, the channel must wait for either a CPU time out or return to poll status before the channel can execute halt I/O. If a CPU time out occurs before the channel returns to poll status, channel logic circuits terminate certain multiplexor channel I/O operations. Furthermore, if the channel logic circuits have not terminated the operation prior to a second CPU time out and the channel returns to poll status, the channel requests the routines that execute the halt I/O operation.

Channel in Poll Status

If the multiplexor channel is in poll status, the channel fetches the subchannel sequence controls. The contents of the sequence controls determine the following channel operation.

Not Busy or Not Command Chaining End Received: The common channel sets condition code 0, indicating the operation had previously proceeded to a normal end and CPU stat 3 to permit CPU to break from the countdown loop and return to I-fetch.

Busy or Command Chaining End Received: The multiplexor channel initiates the interface selection sequence. If the unit responds with operational in, the channel executes the selective reset sequence. When the unit drops operational in, the channel completes the halt I/O sequence by executing the load 64 preparation routine.

If the unit fails to respond to the interface selection sequence with operational in but instead returns status, the channel completes the halt I/O operation by executing the load 64 preparation routine.

If neither operational in nor status in are returned by the unit in reply to the selection sequence, the lowest priority unit propagates select in. Select in causes common channel to set condition code 3, indicating the unit is not available, and CPU stat 3 to permit CPU to break from the countdown loop and return to I-fetch.

Channel Not in Poll Status

If a CPU time out occurs and the channel has not returned to poll status, CPU issues time out to the common channel; time out sets the channel burst mode latch. When the burst mode latch is set, channel logic circuits determine the type of current channel operation, and how the I/O operation is to be terminated. If it is determined that the channel is currently performing a data operation, the channel issues a selective reset to terminate the data operation, regardless of the address of the active unit. Common channel sets condition code 2, indicating to CPU that a data operation was terminated.

If it is determined that the current operation is command chaining and the unit has presented channel and device end status bits to the channel, the logic circuits reset the command chaining flag; the reset command chain flag causes the operation to end. The channel executes the load 64 preparation routine to complete the halt I/O sequence.

Test Channel

- Execution of test channel does not interfere with the current channel operation.
- If no operation is in process, execution of the test channel instruction does not cause the channel to switch from the not busy state.
- The test channel operation makes available to the CPU program information indicating the state of the channel.

After decoding the test channel instruction, the CPU:

1. Loads the L register with bits identifying the channel to perform the operation
2. Gates test channel to the common channel, makes L register positions 21-23 available to the common channel
3. Enters a countdown loop to wait for the CPU stat 3 reply

Test channel causes the common channel to select the multiplexor channel and to test the state of the interrupt buffer full latch. Common channel sets: (1) condition code 0 and stat 3 if the interrupt buffer full latch is reset, indicating to CPU that the channel is free of interrupt information, or (2) condition code 1 and stat 3, indicating to CPU that the channel is holding interrupt information. If the channel is processing a previously initiated operation, test channel causes the channel to test the burst mode latch; the burst mode latch causes common channel to set condition code 2 and stat 3.

Execution of the test channel instruction does not interfere with the current channel operation or cause the channel to switch to the busy state if no operation is in process.

MULTIPLEXOR CHANNEL ROS ROUTINES

- ROS contains 30 multiplexor channel microprogram routines.
- Each routine is a predefined microprogram that renders a different service to the channel.
- A routine can contain more than one path of microinstructions.
- In any path the entire microprogram may not require more than six microinstructions for completion.
- The channel sets I/O stats to provide branch conditions for the microprogram.

The channel assigns a priority and generates a fixed address for most channel microprogram routines. To request these routines, the multiplexor channel sends the priority to the common channel. If the common channel has not received a request of equal or higher priority from another channel, the channel transmits the five bits from the decoder to ROAR positions 6-A when CPU allows the break in. The five bits, different for each routine, complete the ROS address that locates the first microinstruction in the requested routine. Although a routine may contain more than one path, the first microinstruction in the routine is common to all paths. A branch in the microprogram can occur only after ROS performs the first microinstruction.

While ROS is executing a microprogram to service the channel, hardware facilities in the channel function to accept the service. At various points in the execution of the routine, ROS transmits a DTC signal to the channel to synchronize the microprogram and channel hardware actions.

Because some microprograms are common to more than one channel operation, the microprogram may be shown on several different flow charts. For example, routine B0, the sequence controls fetch routine, is the first routine shown on the flow charts for start I/O initial selection, test I/O, and halt I/O.

Figures 29 and 30 are examples of the sequence of multiplexor channel routines that are used to execute byte-mode and burst-mode read operations.

Initial Selection

- Made up of B0, B1, and B2.
- Start I/O from CPU causes the multiplexor channel to execute the initial selection sequence.
- Channel must execute six microprograms contiguously to complete the initial selection sequence.
- The microprogram routines:
  1. Assemble the control data provided by CPU in designated channel storage areas to enable the channel to sustain an I/O operation without CPU intervention.
  2. Select and transmit a command byte to the I/O unit that indicates the type of operation to be performed.
  3. Analyze the unit status byte.
  4. Have a priority of three.
- Channel sets condition codes and CPU stats that indicate the state of the channel, subchannel, and I/O unit.

Buffer out bus bit 7 from CPU to the common channel causes the common channel to issue start I/O to the channel. If the multiplexor channel is in poll status, it requests the first of six microprogram routines that are chained to execute the initial selection sequence. These routines continue the start I/O operation in I/O mode after CPU has entered the countdown loop.

B0--Sequence Control Fetch (3): The initial selection routines enable the multiplexor channel to transfer the unit address and CCW-1 from the CPU L register and R register, respectively. The data address contained in CCW-1 is stored in the data address word section of local storage that is reserved for the multiplexor channel. The unit address is used to address bump storage to obtain the subchannel sequence controls for the unit specified by the CPU start I/O instruction. The contents of the sequence controls enable the multiplexor channel to determine if the subchannel is busy or not busy. If the subchannel is busy, the channel terminates the start I/O operation; common channel sets condition code and a CPU stat that signals CPU the specified start I/O operation was not initiated. If the subchannel is not busy, the channel tests for interrupt conditions. If the channel is holding an interrupt, the interrupt condition is cleared and priority is given to the start I/O operation.

Each subchannel has six operating states as indicated by the sequence controls maintained in the control word in BUMP. These controls are modified during I/O operations to provide the current state of the subchannel. The current state of the sequence controls is used to direct I/O operations. The states are:

Idle (0000) -- The addressed subchannel, control unit, and device are not busy and do not contain any pending interrupt conditions.

Busy (0001) -- A command has been initiated, but end status has not been received.

Channel End During Command Chaining (0011) -- Channel end, but not device end, has been received during command chaining.

End Status Received (0101) -- The channel end status has been stored in the interrupt buffer.

End Status Queued (0111) -- The channel end status could not be stored because the interrupt buffer was full. The status is stacked at the CU.

Device End or Attention in the Interrupt Buffer (0110) -- The I/O device has terminated the operation and the end status is in the interrupt buffer.

The six states for various I/O operations are shown in Figure 28.

B1--Start I/O Unit Select (3): If the sequence controls indicate the device is either idle or has device end/attention in the interrupt buffer, CCW-2 is fetched and initial selection of the I/O device begins. Device end/attention in the interrupt buffer is cleared. This status information will be received from the CU and will be stored in the CSW. The unit address is sent out to the device. If no response is received from the device, condition code 3 is prepared to indicate the unit is not operational.

The four other possible states for the sequence controls, busy, channel end received for command chaining, end status in the interrupt buffer, or end statusqueued indicate the channel is busy and condition code 2 is prepared.

Both CCW-1 and CCW-2 are tested for program errors; if either CCW contains a program error, the channel terminates the start I/O operation. The channel: (1) sets the program check latch, (2) breaks the initial selection sequence chain, and (3) requests the load 64 preparation routine.

B2--Start I/O Unit Address Compare (3): The I/O device that is selected returns its address. The address returned is compared to the address transmitted. If the addresses are the same, the command byte is sent to the device.

B3--Start I/O Status Analysis (3): The address of the next CCW is stored in BUMP. The status from the CU is checked and, if all zeros, condition code 0 is prepared to indicate acceptance of the command. If channel end, with or without device end, is returned and command chaining is specified, condition code 0 is prepared. Any other status bits being on cause the CSW to be stored and condition code 1 to be prepared.

If the unit does not request service immediately following the initial selection sequence and the operation code is input, the I/O stats are changed to the output operation code. The output operation code prevents the routines that store the control information in bump storage from executing an erroneous data address update function.

B5--Count Store (3): The byte count from the multiplex area of LS is stored in BUMP.

B6--Data Address Store (3): The data address from LS is stored in BUMP. The channel returns to poll status after ROS executes routines that copy the local storage control information into the bump storage area designated for the subchannel. If the local storage control information is not stored in bump storage, subsequent I/O operations would alter the local storage control data and render the information useless.

Data Handling

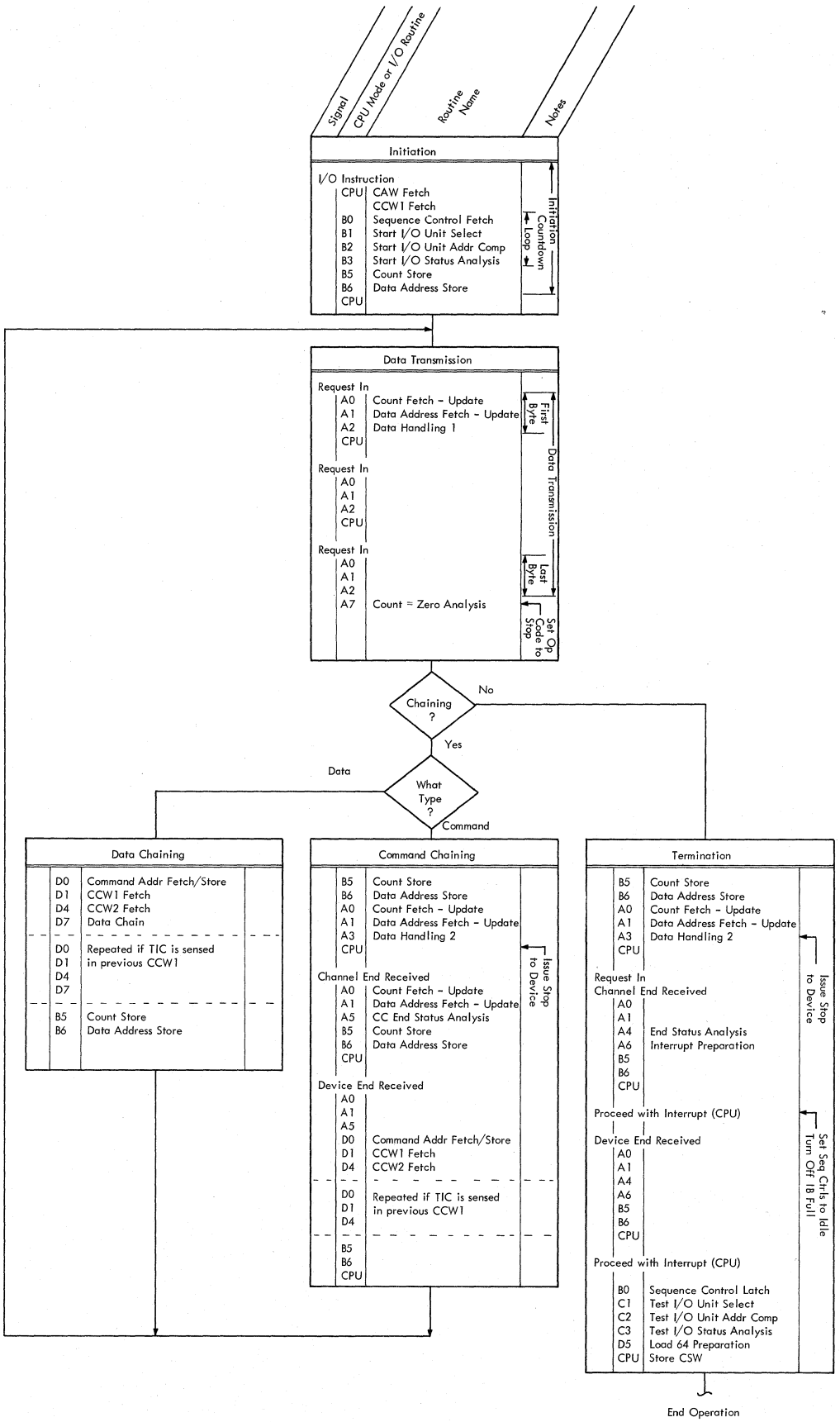
- Input and output data transfers are controlled by channel routines.
- The channel requests data handling routines when a unit requests data service from the channel, or when initial selection is complete and the specified operation code is output mode.
- The routines maintain an updated data count and data address.

The channel routines provided for data operations can be divided into three groups, according to the function they perform.

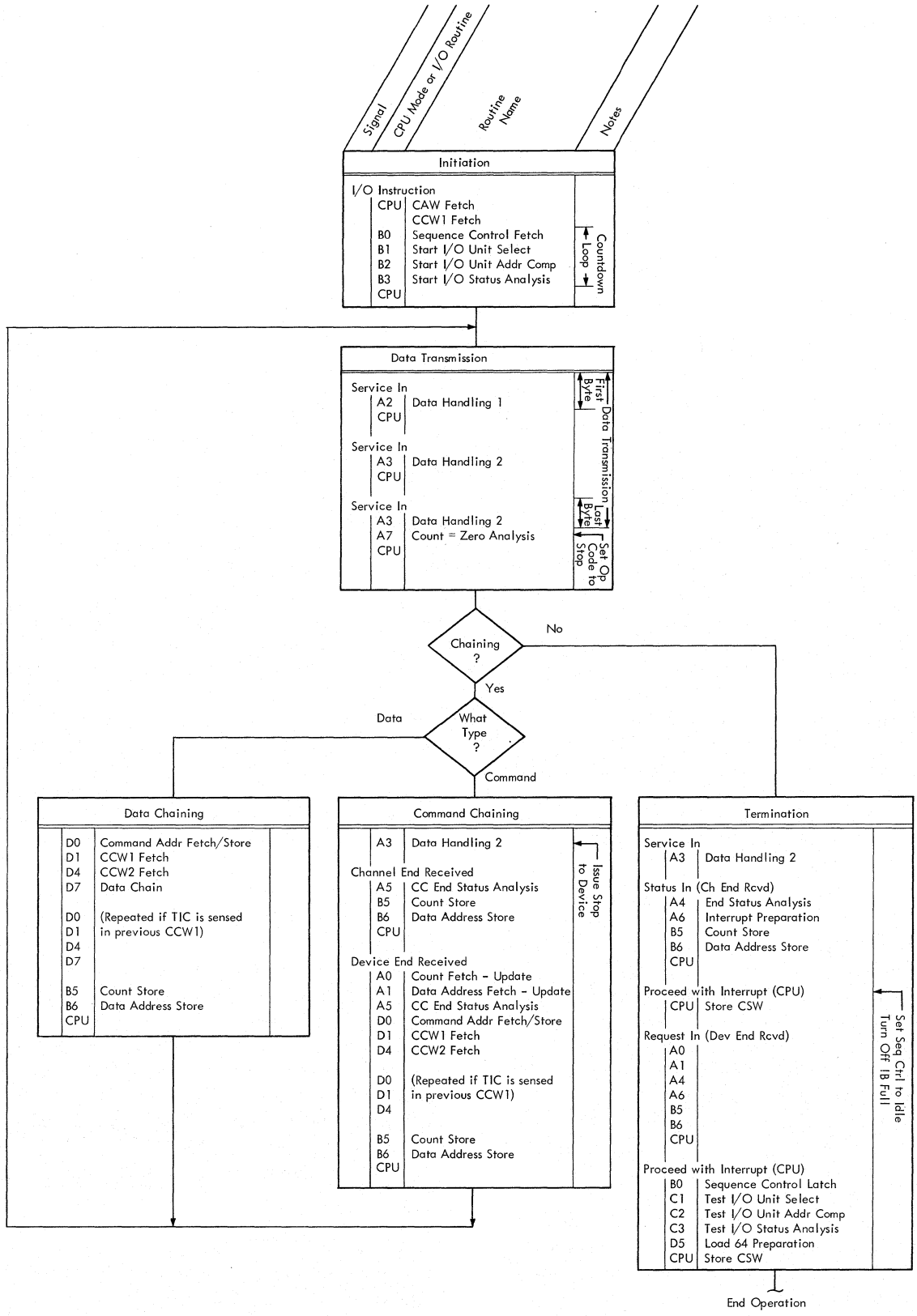
1. Routines A0 and A1 comprise the first group; these routines fetch the control information from bump storage and move the control information to local storage.
2. Routines A2 and A3 comprise the second group; these routines transfer data from either the interface bus in to main storage or from main storage to the interface bus out. They maintain an updated local storage data count and data address.
3. Routines B5, B6, and B7 comprise the third group; these routines return the updated local storage data count and data address to bump storage.

Channel in Poll Status

In poll status, the channel is not busy. The channel drops poll status and becomes busy if either the CPU or the interface issues signals to the channel. CPU requests for channel action are excluded in the following channel operations because the examples are limited to data operations. Receipt of address in causes ROS to address bump storage with the unit address received on the interface bus in. Therefore, address in causes the channel to make sequential requests for routines A0 and A1. ROS executes the routines for the following purposes:



● FIGURE 29. EXAMPLE OF A READ OPERATION (BYTE MODE)



● FIGURE 30. EXAMPLE OF A READ OPERATION (BURST MODE)

A0--Count Fetch and Update (3): A request for service from an I/O device causes the A0 routine to be requested. The count is fetched, decremented by one, and stored again in BUMP. The undecremented count is placed in LS where it can be accessed quickly to be decremented in case of burst mode operation. The I/O device forces burst mode for two or more bytes by making repeated service in requests while connected to the channel.

A1--Data Address Fetch and Update (3): The data address is fetched, updated by one, and stored again in BUMP. The unchanged address is stored in LS for use in case of burst mode.

Continued channel operation depends on the next interface reply. If the reply is status in, the channel requests an end status analysis routine; concepts of end status analysis are presented in the End Status Analysis and Command Chaining sections of this manual. If the reply is service in, the channel requests routine A2.

Input Forward and Backward

A2--Data Handling 1 (3): ROS executes data handling 1 for the following purposes:

1. To fetch the required data byte from main storage and to transfer the byte to the requesting unit.
2. To update the local storage data address and data count.
3. To test for data count equal to zero.

If the data count equals zero, the channel terminates the data handling operation and sequentially requests the routines that prepare the subchannel for subsequent end status analysis operation. If the data count is not equal to zero, the next interface reply determines the next channel routine request. If the reply is status in, the channel requests the appropriate end status analysis routine. If the reply is not service in and operational in drops, indicating that the unit requires no additional data service, the channel returns to poll status. The data count and data address are not returned to bump storage because both the bump and local storage data counts and data addresses are correct after a data byte is transferred by routine A2.

If the unit operates in burst mode, the unit does not drop operational in; instead, the unit replies with another service in. The channel requests routine A3, for burst mode operations.

A3--Data Handling 2 (2): ROS executes routine A3 once for each byte of input data transferred. There is one main difference between data handling routines A2 and A3; routine A3 sets the UCW store trigger in addition to performing the three data handling 1 functions.

If the count equals one, the channel:

1. Requests the count zero analysis routine to prepare the subchannel for end status analysis operation.
2. Executes routines B5 and B6; these routines store the local storage data count and data address in bump storage.
3. Returns to poll status.

Output

If the interface reply is not status in, the channel requests routine A2 without waiting for service in.

A2--Data Handling 1 (3): Routine A2, as used for output mode, differs from the routine A2 of input mode. Routine A2 fetches the required data byte and transfers the data to the interface. Output mode data handling of routine A2 does not modify the data count or data address; therefore, after the first pass through the routine, the local storage count is one less than the number of data bytes transferred. Because of this one count difference, the channel terminates output data transfers when the count equals one.

If the unit replies with service in, indicating that the first byte of data was stored by the unit, the channel requests routine A3.

A3--Data Handling 2 (2): ROS executes routine A3 without waiting for the next service in reply. The data byte fetched by either the data handling 1 or 2 routine prior to receipt of service in is called a prefetched byte. Prefetching permits the channel to have one data byte ready for output mode transfer prior to the request for data. Faster channel operation is possible when data is prefetched. ROS executes routine A3 to:

1. Fetch a data byte from main storage and transfer the byte to the interface.
2. Update the local storage data address and data count.
3. Test for data count equal to one.
4. Set the UCW store latch.

The channel executes routine A3 once for each byte of data transferred if the unit continues operating in burst mode. When the unit no longer requires data, service in and operational in are dropped. Because of the channel output mode prefetching, the channel terminates output data transfers with an unused data byte in the interface buffer out latches.

Input Skip

If the reply is service in and the operation code is input skip, the channel requests routine A3. Because the input skip operation is not used to store data in main storage, routine A3 does not address main storage for input skip operation. When ROS executes routine A3 for input skip operation, the routine:

1. Does not transfer data.
2. Decrements the local storage data count.
3. Is executed once for each service in received.

The channel terminates the input skip operation with the same sequence previously described for input forward data handling operations.

UCW Store Operation

When the unit no longer requires service, service in and operation in are dropped. The UCW store latch indicates that the bump and local storage data counts and data addresses are not equal; the latch also causes the channel to request routine B5. ROS executes these routines to store in bump storage (1) the updated local storage data count word (routine B5), and (2) the updated local storage data address word (routine B6). If any data errors occur during the data handling operations, routines B5 and B6 are preceded by check handling routine B7. After the ROS executes routines B5 and B6, the channel returns to poll status.

B5--Count Store and B6 Data Address Store: All requests from an I/O device begin with routines A0 and A1 which modify the count and data address stored in BUMP. If no data is transferred following the request, this count and address will be incorrect. Routines B5 and B6 are used to fetch the correct count and address from LS and place them in BUMP.

B7--Check Handling (3): Channel checks are stored in check latches. Routine B7 stores the resulting check bits in LS.

Variations in Data Handling

PCI (Program Controlled Interrupt)

When ROS executes routines A2 or A3, a test is made to determine if the PCI flag is set. If the flag is set, ROS sets the PCI request latch. The PCI request latch causes common channel to request routine C0; ROS executes the routine C0 after the current data handling routine. The PCI routine signals CPU that the first byte of input data has been transferred.

C0--PCI (3): Routine C0 performs the following functions:

1. Sets the PCI code in the interrupt buffer.
2. Sets the unit address in the interrupt buffer.
3. Resets the PCI flag in the data count word.
4. Sets the interrupt buffer full latch.

Execution of the PCI request routine does not change the channel data handling sequence. After the PCI routine is executed, ROS executes the next requested channel routine.

Channel Not in Poll Status

When the channel is in an initial selection sequence, two conditions cause the channel to request data handling routines; receipt of service in, and an output mode operation code.

When service in and operational in prevent the channel from returning to poll status, the channel requests a data handling routine. For channel input operations, the channel requests routine A2. Routines A0 and A1 are bypassed because the initial selection sequence has the subchannel control information assembled in local storage at the end of the initial selection sequence. ROS executes routine A2 to transfer the first byte of data. Preceding input and output mode data handling sequences describes the routines by which the channel transfers data, and stores or updates the control information in bump storage.

For channel output operations, the channel requests routine A2 without waiting for service in. Again, routines A0 and A1 are bypassed because the initial selection sequence has the subchannel control information assembled in local storage. ROS executes routine A2; a byte is prefetched. If the unit does not request the prefetched data byte, the channel requests routines B5 and B6. These routines store the local storage control information in bump storage. After ROS completes the execution of routines B5 and B6, the channel returns to poll status.

If the channel requests the prefetched data byte, refer to the preceding output mode data handling sequence for a description of how the channel transfers data, and stores or updates the control information in bump storage.

Stop Data Handling

The stop operation code is not issued to a subchannel as the result of start I/O and initial selection operation, but it is developed by channel routine A7 after the channel has transferred the last byte of data to the unit. When the data count equals zero during input or one during output mode data handling, the channel requests the count zero analysis routine A7.

A7--Count Equal Zero Analysis (3): Routine A7 performs the following functions:

1. For an output operation code, reduces the count from one to zero.
2. Changes the local storage operation code to stop.
3. Sets the UCW store latch.

The following interface reply determines the next channel routine request. If the unit does not reply with service in or status in, the channel requests routines B5 and B6. If the unit replies with status in, the channel requests the appropriate end status analysis routine. If the reply is service in, the channel requests routine A3. Routine A3 issues command out to the unit; command out causes the unit to stop sending service in. The following interface reply again determines the next channel routine request; for status in, the channel requests the appropriate end status analysis routine.

If there is no reply and operational in drops, the channel returns to poll status after ROS executes routines B5 and B6.

Chaining

- A single start I/O instruction initiates chaining.
- Chaining permits the channel to fetch a new CCW after the current CCW is exhausted.
- Two types of chaining are provided: chaining of data, and chaining of commands.

When the channel completes the transfer of data specified by a CCW, channel operation continues if one of the chain flags is set. The end of channel data operations includes a test for the presence of either chain flag; the presence of a chain flag causes the channel to request routines that fetch a new CCW. The new CCW permits the channel to resume operation with the unit specified by the start I/O instruction. ROS executes the channel chaining routines:

1. To fetch the command address.
2. To fetch CCW-1 and CCW-2.

These functions are common to both command and data chaining.

The command address fetch routine D0 fetches the command address from bump storage and stores it in local storage. The local storage command address is required to fetch the new CCW. CCW-1 fetch routine D1 and CCW-2 fetch routine D4 each fetch their respective halves of the CCW from main storage. The common functions of chaining end when both halves of the new CCW are stored in local storage. The contents of the new CCW determine the remaining channel chaining operations.

Data Chaining

- The new CCW specifies a new data address for the original I/O operation.
- The new CCW specifies a new data count for the original I/O operation.
- Successive CCWs must be in adjacent double-word main storage locations unless the new CCW contains a transfer in channel (TIC) command.

Chain data operations begin when the data count in the current CCW equals zero. The count equals zero condition causes the channel to request the count equal zero analysis routine A7. Routine A7 tests for the chain data (CD) flag. If the chain data flag is set, the channel requests the command address fetch routine D0. Routine D0 adds eight bytes to the previous command address. After ROS executes routine D0, the channel requests CCW-1 fetch routine D1. Routine D1 uses the updated command address to fetch a new CCW-1 from main storage; the new CCW-1 is stored in local storage. Next, ROS executes CCW-2 fetch routine D4.

Routine D4 uses the updated command address plus four bytes to fetch a new CCW-2. If CCW-1 contains a TIC command, the channel again requests the command address fetch routine. The command address fetch routine detects the TIC entry code in the I/O stats and again adds eight bytes to the command address. The channel again requests the CCW fetch routines; these routines fetch another new CCW. If the new CCW also contains a TIC command, the program check latch is set. A program check causes the channel to request data chaining routine D7. Routine D7 sets the operation code to stop in the local storage. If the unit replies with service in, the channel requests data handling routine A3; routine A3 issues command out to terminate the unit operation. If the unit replies with status in, the channel sequentially requests the end routines A4 and A6; the routines process status, load the interrupt buffer, and request an interrupt.

If a new CCW is available in local storage and a program check has not been detected, the channel requests data chaining routine D7. The new flags, old operation code, old channel status, and new data count are formed in the local storage data count word location. After ROS completes the execution of routine D7, the chaining operation is complete.

Following is a brief description of the routines following A0, A1, and A3 when the last byte is detected.

A7--Count Equals Zero Analysis (3): Although this routine is requested because the data chaining flag is set, the only action taken is to request the next routine.

D0--Channel Address Fetch and Store (3): To obtain the address of the next CCW, the CAW is fetched from BUMP and placed in LS. The address is updated for the next CCW and replaced in BUMP.

D1--CCW-1 Fetch (3): The first half of the CCW is fetched from the SE and stored in LS.

D4--CCW-2 Fetch (3): The second half of the CCW is fetched from the SE and placed in the L register.

If the CCW-1 just fetched specifies a TIC, the A7, count zero analysis, and the D0, channel address fetch and store, routines are requested. The address for the D0 routine is fetched from LS (the CCW-1 just stored). These routines are followed by D1, CCW-1 fetch, and D4, CCW-2 fetch, routines.

D7--Chain Data Address (3): Housekeeping is performed to establish control conditions for the old and new operations.

B5--Count Store (3): The new byte count is stored in BUMP.

B6--Data Address Store (3): The new data address is stored in BUMP.

Command Chaining

- The new CCW initiates a new operation with the I/O unit specified by the start I/O instruction.
- Command chaining initiates a new operation only if the current operation ends without error.
- Completion of the current command chaining operation without errors does not cause an interrupt until chaining is completed.
- Successive CCWs must be located in adjacent double-word main locations unless the unit presents a status modifier bit with the device end status.
- Command chaining is not possible if the CCW contains both command chain and data chain bits.

The command chain operation begins when an I/O unit presents status to the channel with the chain command latch set. Status in and the set chain command latch cause the channel to request the command chain end status analysis routine A5.

A5--CC End Status Analysis (3):

1. Detects error conditions and terminates the command chaining operation if any errors are detected.
2. In the absence of errors, processes status without requesting an interrupt.
3. Upon receipt of device end, requests the command address fetch routine D0.

Routine A5 resets the command chaining flag and terminates the command chaining operation if any of the following conditions are detected:

1. Count equals zero and no suppress incorrect length indication.
2. Attention, busy, unit check, or unit exception.
3. Status modifier bit and not the device end.
4. Device end without a previous or accompanying channel end.

Routine A5 codes the sequence controls equal to command chaining end received when the unit status contains channel end. Command chaining end received does not cause an interrupt: instead the channel sequentially requests the count and data address store routines. After ROS executes the store routines, the channel returns to poll status. Receipt of device end with an accompanying channel end or a previous command chaining end causes the channel to begin command chaining by requesting command address fetch and store routine D0. Routine D0 adds eight bytes to the original command address or sixteen bytes to the original command address if the device end status contains a status modifier bit. Next, the channel requests CCW-1 fetch routine D1; routine D1 fetches CCW-1 with the updated command address. The channel requests CCW-2 fetch routine D4; routine D4 fetches CCW-2.

If the CCW does not contain a program error, the channel requests the command chaining initial selection routine D2 to:

1. Select the unit.
2. Compare the interface unit address with the start I/O unit address.
3. Issue the new command byte to the unit.
4. Request command chaining initial status analysis routine D3.

If the routines detect a program error in the CCW, the channel executes the command chaining initial selection routine to:

1. Stop the unit operation with command out.
2. Terminate the command chaining by resetting the command chain flag and setting the operation code to stop.
3. Request command chaining initial status analysis routine D3.

Receipt of status in causes ROS to execute the command chaining initial status analysis routine D3. If the channel has detected a previous error, or if routine D3 detects that the unit status byte is not

equal to zero, the channel sequentially requests the interrupt preparation, count store, and data address store routines. After these routines are completed, the channel returns to poll status. If the channel has not detected a previous error and the unit status byte equals zero, the channel command chaining is completed. The subsequent channel routine requests depend on the new operation code and the next interface reply.

Following is a brief description of the routines required for command chaining.

Last Byte: The last byte of data is transferred. The routines executed are:

A0--Count fetch and update.

A1--Data address fetch and update.

A3--Data handling (last byte).

A7--Count equals zero analysis. Routine A7 is requested because routine A3 detects last byte. Because this is a chained command, this routine sets the operation code to stop in LS.

B5--Count store.

B6--Data address store. The count and data address have been modified in BUMP and are incorrect. B5 and B6 fetch the correct count and address from LS and place them in BUMP.

Stop: The next request from the I/O device results in the normal data handling routines.

A0--Count fetch and update. The count in BUMP is modified normally.

A1--Data address fetch and update. The data address in BUMP is not modified in A2 because the operation code is set to stop.

A3--Data handling. Because the operation code is set to stop, no data is handled. The stop command is sent to the device. The correct count from LS is stored in BUMP. Routines B5 and B6 are not required to correct the count and address.

Channel End Status: When the request is made to present channel end status, the data handling routines A0 and A1 are executed.

A0--Count fetch and update.

A1--Data address fetch and update.

A5--Command chaining end status analysis. Routine A5 is requested to perform the end status analysis because channel end status has been presented.

B5--Count store.

B6--Data address store. The correct count and data address are fetched from LS and stored in BUMP.

Device End Status: When the I/O device makes a request to present device end status, routines A0, A1, and A5 are executed as for channel end.

A0--Count fetch and update.

A1--Data address fetch and update.

A5--Command chaining end status analysis. Routine A5 is requested to perform the end status analysis. A branch to D0 is performed to fetch the next command address.

D0--Command address fetch and store. The address of the next CCW is fetched from BUMP and stored in LS. The address is updated and replaced in BUMP.

D1--CCW-1 fetch. The first half of the CCW is fetched from storage and stored in LS.

D4--CCW-2 fetch. The second half of the CCW is fetched from storage and stored in the L register. Address out is set and the device is selected.

D2--Command chaining unit address compare. The address received from the device selected is compared to the address transmitted. If the addresses are equal, the command is sent to the device.

D3--Command chaining initial status analysis.

B5--Count store.

B6--Data address store. The count and data address from LS are stored in BUMP.

If a TIC was detected in the CCW-1 fetched in D1, routines D0, D1, and D4 are repeated to fetch the new CCW.

The groups of routines executed for command chaining, last byte, stop, channel end status, and device end status can occur at widely separated times. Last byte, stop, and channel end status usually occur within the same millisecond but depend mainly on other multiplexor channel activity. Device end may occur several seconds after channel end if mechanical motion is involved. Some devices present device end and channel end at the same time, and the channel end sequence of routines is not required. If the I/O device terminates the command before the count equals zero, the stop command is not required.

Termination

An I/O operation is normally terminated after one command or after a series of chained commands. A stop command must be issued to the I/O device for terminations that occur as a result of the byte count reaching zero, an error condition at the channel, or a halt I/O instruction. Terminations can also occur as a result of either the end-of-record or error conditions detected at the device or CU. (The CU may recognize end-of-record as a control line, a special bit, a special byte, or a series of bytes.)

If the count goes to zero, the last byte and the stop sequences are the same as for command chaining. If the count does not go to zero, the data-handling routines service the last byte. When channel end status is presented, the sequence deviates from the command chaining channel end. Data-handling

routines A0 and A1 are the same but A4 instead of A5 is used for end status analysis. Routine A6, interrupt preparation, follows. The routines listed below occur when channel end and/or device end is presented. The sequence occurs twice if channel end and device end are presented separately.

A0--Count fetch and update.

A1--Data address fetch and update.

A4--End status analysis. Routine A4 fetches, examines, and prepares the sequence controls (SC) for updating. The SC are examined to determine if the subchannel is active or not and if the interrupt buffer (IB) is full or not. The four possible states and the action taken are listed below:

Sequence Controls Active and Interrupt Buffer Not Full: The routine sets the sequence controls equal to channel end in the interrupt buffer. Service out is issued to the unit indicating that the status byte was accepted. The channel requests the interrupt preparation routine.

Sequence Controls Active and Interrupt Buffer Full: The routine sets the sequence controls equal to channel end queued. Command out is issued to the unit, indicating that the unit must stack the status. The channel requests the interrupt preparation routine.

Sequence Controls Not Active and Interrupt Buffer Not Full: The routine sets the sequence controls equal to device end in the interrupt buffer. The channel issues command out, indicating that the unit must stack the status. The channel requests the interrupt preparation routine.

Sequence Controls Not Active and Interrupt Buffer Full: The routine sets the sequence controls equal to idle. The channel issues command out to the I/O unit, indicating that the unit must stack the status. The channel requests routine B5, the count store routine; after ROS executes the count and data address store routines, the channel returns to poll status.

ROS executes the interrupt preparation routine to load the interrupt buffer and to request an interrupt. Unless the sequence controls are active with the interrupt buffer full, the routine loads the interrupt buffer and requests an interrupt. When the sequence controls are active and the interrupt buffer is full, the routine compares the interrupt buffer and local storage unit addresses. If the addresses are an unequal compare, the channel issues command out to the unit indicating another unit has status information in the interrupt buffer. If the addresses are an equal compare and the interrupt buffer interrupt code equals PCI, the channel end code is added to the current contents of the interrupt buffer. The channel: (1) sets service out, indicating the status byte was accepted and (2) requests an interrupt.

A6--Interrupt Preparation: Routine A6 branches on the SC prepared in A4. If channel end in IB or device end in IB has been prepared (items 1 and 3 under A4), the end status and the interrupt code are stored in the IB. If channel queued has been prepared (item 2 under A4), the IB and LS addresses are compared. If the addresses are not equal, the IB contains status for another unit. Command out is issued to the device. The unit stacks the status. If the addresses are equal and the IB interrupt code indicates a PCI, channel end is added to the current contents of the IB. Service out is issued to the unit to indicate the status was accepted. Because IB is full, and if channel 0 is not masked, common channel conditions branch on channel interrupt (BCHI) so that CPU can recognize the pending interrupt at the beginning of the next I-Fetch.

B5--Count store.

B6--Data address store. The count and data address from LS are stored in BUMP.

Interrupt

If the interrupt buffer is full and channel 0 is not masked, an exception branch to the interrupt ROS routine will occur in CPU at the beginning of the next I-Fetch. This interrupt routine, which is executed in CPU mode, issues a "Proceed on Interrupt" (E→CH/E = 1001) request to the channel. Multiplex channel responds with an immediate stat 3 reply. The interrupt routine then fetches the interrupt buffer from local storage (Sector 10 Word 1110) to determine the type of interrupt that is pending.

Device End in the IB

If the interrupt code is device end only, an "Interrupt Test I/O" (W→CHCTL/W = 80) is issued to clear the device end status that is stacked in the device and form the CSW prior to completing the interrupt. The "Interrupt Test I/O" operation in the multiplex channel is identical to the test I/O operation, described in this manual, with the exception that a condition code is not set at the end of initial selection.

Other Than Device End in the IB

If the interrupt code is other than device end, the CPU interrupt routine fetches the CSW data from the interrupt buffer and the bump storage (UCW) locations associated with the unit address in the IB before completing the interrupt.

Figure 31 shows the interrupt flow for the multiplex channel. The solid blocks denote CPU mode functions while the dotted blocks are I/O mode routines.

Test I/O

- Tests addressed units for interrupt conditions.
- Clears all interrupt conditions on the addressed unit.
- Sets condition code in the program status word.
- Provides CPU with data to update the CSW.

Buffer out bus bit 5 from CPU to the common channel causes the common channel to issue test I/O to the channel. If the multiplexor channel is in poll status, it requests the first of four microprogram routines that execute the test I/O sequence. These routines continue the test I/O operation in I/O mode after CPU has entered the countdown loop.

The routines in the test I/O sequence fetch the sequence controls from bump storage and analyze them. Based on the state of the sequence controls, the channel indicates that the unit is either busy or unavailable. Next the channel may:

1. Execute the load 64 preparation routine, or
2. Select the unit, process the unit status, and execute the load 64 preparation routine.

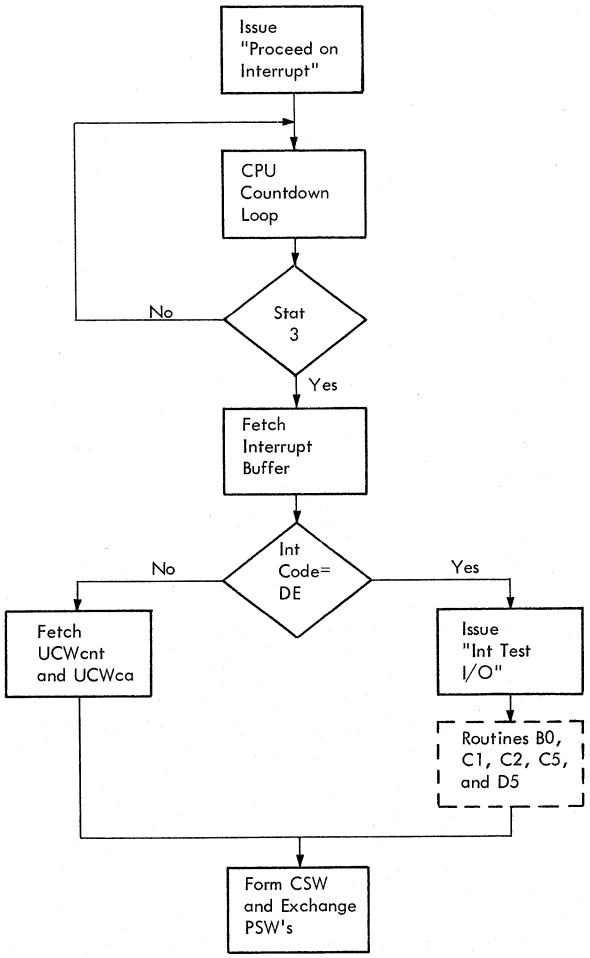
Routine D5:

1. Loads the M register with unit and channel status.
2. Loads the L register with unit address.
3. Loads the R register with zeros.

The channel returns to poll status.

Routine B0 addresses bump storage and fetches the sequence controls; the test I/O instruction unit address and sequence controls are used by the remaining routines to select and to determine the state of the subchannel and unit.

Following routine B0, the channel requests test I/O unit select routine C1. Routine C1 tests the sequence controls and executes one of the following sequences.



● FIGURE 31. PROCEED ON INTERRUPT SEQUENCE

Sequence Controls Idle: The subchannel is not busy; therefore, the channel requests: (1) test I/O unit select routine C1 to select the unit, and (2) test I/O unit address compare routine C2 to compare the interface unit address with the instruction unit address. After the address is verified, command out is issued to the unit. When the unit returns its status byte, the channel requests test I/O status analysis routine C3. Routine C3 stores the unit status byte and issues service out to reset the device end latch. If the unit status byte equals zero, common channel sets a condition code 0 and CPU stat 3, indicating to CPU that the unit is available and not busy. Channel returns to poll status.

If the unit returns a status byte that is not equal to zero, routine C3:

1. Stores the unit status byte.
2. Issues service out to reset the device end latch.
3. Loads the L register with unit status, channel status, and data count.
4. Codes the I/O stats for the proper entry into the load 64 preparation routine D5.

Device End or Attention in the Interrupt Buffer: The beginning of the test I/O sequence is the same as when the sequence controls are idle. Routines B0, C1, and C2:

1. Fetch the sequence controls.
2. Select the unit.
3. Compare the interface unit address with the instruction unit address.
4. Issue command out to the unit.

When the unit returns its status byte, the channel requests routine C3.

Routine C3:

1. Stores the unit status byte.
2. Issues service out to reset the device end latch.
3. Loads the L register with the unit status, channel status, and data count.
4. Codes the I/O stats for the proper entry into the final multiplexor routine (load 64 preparation routine D5).

Routine D5:

1. Loads the M register with unit and channel status.
2. Loads the L register with the unit address.
3. Loads the R register with zeros.

Common channel sets CPU stats 2 and 3 that signal CPU to load the channel status word. Next it sets a condition code 1, and channel returns to poll status.

Channel End Queued: Routine C1 fetches the unit address from bump storage and compares it with test I/O instruction unit address. If the addresses are not an equal compare, common channel sets CPU stat 3 and condition code 2, indicating the unit is not available.

If the addresses compare equally, the channel selects the unit with test I/O unit select routine C7. When the unit returns its address, test I/O unit address compare routine C2 compares the interface unit address with the instruction unit address and issues command out.

When the unit returns its status byte to the multiplexor channel, routine C3:

1. Issues service out to reset the channel end latch.
2. Sets the data address word to zero.
3. Sets the sequence controls equal idle.

Based on the original state of the sequence controls (channel end queued), the multiplexor channel executes load 64 preparation routine D5.

Routine D5 completes the test I/O operation with the following sequence:

1. Unit address is stored in the L register.
2. Unit status, channel status, and data count are stored in the M register.

The channel returns to poll status.

Channel End in the Interrupt Buffer: This condition is processed in much the same manner as the channel end queued condition. Test I/O unit select routine C1 fetches the interrupt buffer contents. The routine compares the instruction unit address with the interrupt buffer unit address. Common channel sets: (1) a CPU stat 3 and (2) condition code 2 if the addresses do not compare equally, indicating the unit is not available.

If the addresses compare equally, routine C7 loads the L register with:

1. Unit status from the interrupt buffer.
2. Channel status from the unit control word.
3. Data count from the unit control word.

The I/O stats are coded for proper entry into the interrupt preparation routine D5. The PCI bit is added to the I/O stat code, if the condition is detected.

Routine D5 loads the L and M registers as follows:

1. If the I/O stats contain a PCI bit, PCI is added to the channel status.
2. The unit status, channel status, and data count are stored in the M register.
3. Unit address is stored in the L register.
4. The data address word in bump storage is set to zero; this action sets the sequence controls idle and in agreement with cleared channel end condition.

The channel returns to poll status.

Test I/O Routines

Following is a brief description of the routines used for test I/O:

B0--Sequence Controls Fetch (3): The unit address of the test I/O instruction is used to form the bump address and the sequence controls are fetched.

C1--Test I/O Unit Select 1 (3): The sequence controls are tested. Further actions are:

Busy or CC End Received -- No further action in routine C1. Condition code 2 and stat 3 are set to indicate that the subchannel is busy. The channel returns to polling.

Channel End in the IB -- The interrupt buffer is fetched from local store and the unit address in the IB is compared with the unit address in the test I/O instruction. If the addresses are equal, IB full is reset and a request is made for routine C7. If the addresses are not equal, condition code 2 and stat 3 are set to indicate that the channel is busy.

Channel End Queued -- The UCW unit address is fetched and a request is made for routine C7.

Device End or Attention in the IB -- Reset IB full and proceed with interface selection sequence as described under "Idle" below.

Idle -- Set address out and select out to start an interface selection sequence. Upon proper interface response a request will be made for routine C2.

C7--Test I/O Unit Select 2 (3): The I/O stats are tested to determine which of the two instances in routine C1 caused the request for C7. Results are as follows:

Channel End in the IB and Test I/O ua Equal to IB ua -- The UCW command address is fetched and placed in the R backup register. Channel and unit status is formed in the L register and a request is made for routine D5 (load 64 prep).

Channel End Queued -- If the test I/O ua is equal to UCW ua, the true command address is fetched and placed in the R backup register. Address out and select out is set to initiate an interface selection sequence. Upon proper interface response a request will be made for routine C2. If the test I/O unit address does not compare with the UCW ua, condition code 2 and stat 3 are set. The channel returns to polling.

Interface Responses: Once initial selection sequence has been started the response can be "Select In," indicating that there is no device available. If the operation is "Interrupt Test I/O," this condition should not occur and a logout will be requested following a time out. If, however, the operation is a test I/O instruction, this conditon will cause conditon code 3 and CPU stat 3 to be set, indicating that the device is not available.

If the device is available, "OP IN" will rise, "Address Out" will be reset, and "Address In" will rise. At this point "Select Out" is reset and a request is made for routine C2.

C2--Test I/O ua Compare (3): The UCW count word is fetched and placed in local storage count word. The address out is compared with the address in and, if unequal, a logout is requested. If the two addresses are equal, "Command Out" is set to cause the interface sequence to proceed. Upon response by "Status In" a request will be made for routine C3.

C3--Test I/O Status Analysis (3): The status byte is accepted from bus in and "Service Out" is set to clear any status that may be stacked in the device. The unit status byte is tested to determine the function of the rest of routine C3.

Status Modifier or Busy Bit Alone -- The I/O stats are coded to allow the device end entry to routine D5 (load 64 prep). The unit status is placed in byte zero of the L register and a request is made for routine D5.

Status Not Zero and Not Channel End Queued -- The sequence controls are set to idle and the I/O stats are coded to allow device end entry to routine D5. The unit status is formed in byte zero of the L register and routine D5 is requested.

Channel End Queued -- The sequence controls are set to idle and the I/O stats are coded to allow channel end entry to routine D5.

Sequence Controls Idle and Status Equal to Zero -- Condition code 0, CPU stat 3 are set.

D5--Load 64 Preparation (3): Routine D5 forms in the L, R, and M registers the CSW data. For entries from test I/O the actions can be placed into two groups:

Channel End Queued or Channel End in IB

1. The unit address is placed in L register byte zero.
2. The M register is set to unit status, channel status, and residual count.
3. The R register is set to the stg. protect key and command address.
4. If the PCI flag is set and entry was from channel end in the IB, the PCI bit is set in the channel status byte.

Other Test I/O Entries

1. The unit address is placed in the L register byte zero.
2. The M register is set to unit and channel status.
3. The R register is cleared.

In both cases an E→CH/E = 0000 micro-order is issued to reset the common channel and CPU stat 3 is set. The channel returns to polling and the CPU forms the CSW and sets condition code 1.

Halt I/O

- Channel routines terminate the unit operation if the channel is in poll status.
- Channel logic circuits terminate unit operation if the channel is not in poll status.

Buffer out bus bit 6 from CPU to the common channel causes common channel to issue halt I/O to the channel. If the channel is in poll status, it requests the first of two routines that execute the halt I/O sequence. These routines are executed while CPU is in the countdown. If the channel is not in poll status, channel logic circuits terminate the current operation.

Channel Not in Poll Status

When the channel cannot request the halt I/O routines, the CPU countdown circuits continue counting. If the channel returns to poll status prior to a countdown time out, the channel requests the routines that execute the halt I/O instruction.

When operation prevents the channel from requesting the halt I/O routines and a time out occurs, the channel is assumed to be in burst mode. CPU issues

time out to set the channel burst mode latch. The countdown is reset, and another countdown is started.

If the channel starts a data handling sequence during the second countdown, a disconnect is issued to stop any unit operation regardless of the instruction unit address. For this situation, common channel sets condition code 2; count store and data address store microprogram routines complete the operation. If the channel returns to poll status during the second countdown, the channel requests the halt I/O routines; the routines execute the halt I/O operation.

If the channel cannot request the halt I/O routines because the channel is busy with the command chaining initial status analysis routine D3, the channel action depends on the unit status. If the device end and channel end are both present in unit status byte, the command chaining flag is reset; common channel sets a condition code; channel returns to poll status. If the status byte equals zero, the count is reset, the channel waits for a return to poll status or device end status with routine D3. If either condition occurs, the halt I/O operation is executed. If a second time out occurs, the channel requests a CPU logout.

Channel in Poll Status

When the channel is in poll status, halt I/O causes the channel to request routine B0. ROS executes sequence controls fetch routine B0. Channel requests halt I/O unit select routine C6 to determine the final course of action based on the contents of the sequence controls.

Not Busy or Not Command Chaining End Received: Common channel sets condition code and stat, indicating to CPU that the unit was not operating; thus, no action is taken.

Busy or Command Chaining End Received: Channel uses routine C6 to select the unit.

When the unit responds with operational in, the disconnect signal is issued to stop the unit operation. The load 64 preparation routine sets the unit control word operation code to stop, resets the chain command flag, and stores the unit address in the L register. The operation is complete when CPU stat 2, CPU stat 3, and condition code 1 are set. If the unit does not select, it is not available.

When a unit responds with status in, it is a busy unit that cannot execute the halt I/O command. The control unit busy routine C5 sets the stop operation code in the M register and the unit status in the L register. The I/O stats are coded equal to control unit busy; the control unit busy code permits the proper entry into the load 64 preparation routine. Load 64 preparation adds unit address to the L register, and sets CPU stats and a condition code.

Test Channel

- Tests the addressed channel for interrupt buffer full, interrupt buffer not full, or burst mode.
- Sets condition code.
- Does not interfere with current channel operation.

Buffer out bus bit 4 causes multiplexor channel to execute the test channel instruction. The test channel instruction is executed by common channel logic circuits. Three condition codes represent the states of the channel.

If the interrupt buffer full latch is set, the common channel replies with stats zero and three, and condition code one.

If the interrupt buffer full latch is off, the channel replies with stats zero and three, and condition code zero.

If the burst mode latch is set, the common channel replies with stats zero and three, and condition code two.

Miscellaneous Channel Routines

The following routines are used in various I/O operations:

C4--Control Check (3): This routine is requested whenever there is a control check, there are no other routines requested, and the mode switch is in the DISABLE position. Its purpose is to reset the common channel, multiplexor channel, and the UCW by setting the sequence controls to IDLE and the op to STOP. A machine check interrupt always follows the use of this routine.

C5--Control Unit Busy (3): If the control unit is busy when selected for a start I/O, halt I/O, or test I/O instruction, unit status will be returned from the control unit. Routine C5 accepts the status and branches to routine D5.

INDEX

CUT ALONG LINE

COMMENT SHEET

SYSTEM/360 MODEL 50 MULTIPLEXOR CHANNEL

FIELD ENGINEERING THEORY OF OPERATION, Y22-2827-0

FROM

NAME _____ OFFICE/DEPT NO. _____

CITY/STATE _____ DATE _____

To make this manual more useful to you, we want your comments: what additional information should be included in the manual; what description or figure could be clarified; what subject requires more explanation; what presentation is particularly helpful to you; and so forth.

FOLD FOLD

FOLD FOLD

How do you rate this manual: Excellent _____ Good _____ Fair _____ Poor _____

Suggestion from IBM Employees giving specific solutions intended for award considerations should be submitted through the IBM Suggestion Plan.

NO POSTAGE NECESSARY IF MAILED IN U.S.A.

FOLD ON TWO LINES, STAPLE, AND MAIL

fold fold

FIRST CLASS
PERMIT NO. 419
POUGHKEEPSIE, N.Y.

BUSINESS REPLY MAIL

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

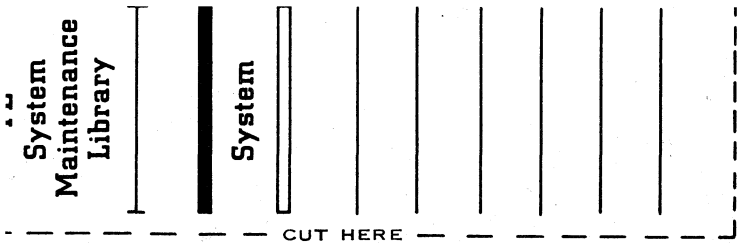
POSTAGE WILL BE PAID BY . . .

IBM CORPORATION

P.O. BOX 390
POUGHKEEPSIE, N.Y. 12602

ATTENTION: FE MANUALS, DEPT. B96

fold fold



Y22-2827-0

Printed in U.S.A. Y22-2827-0

IBM®

International Business Machines Corporation
Field Engineering Division
112 East Post Road, White Plains, N.Y. 10601