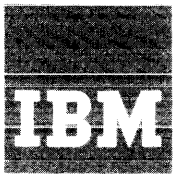
File No. S360-13
Form A22-6900-1



Systems Reference Library

Data Acquisition Special Features for the IBM System/360 Model 44

This publication provides information about the operation, control, and interface of three special features:

Direct Word
Direct Data Channel
Priority Interrupt

These features are particularly suitable for high-speed data acquisition and relatively complex control applications in the scientific fields.

Additional information can be found in *IBM System/360 Model 44 Functional Characteristics*, Form A22-6875, and *IBM System/360 Principles of Operation*, Form A22-6821.

SECOND EDITION

This is a major revision of, and obsoletes, A22-6900-0. Changes to the text are indicated by a vertical line to the left of the change; revised illustrations are denoted by the symbol ● to the left of the caption.

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

Requests for copies of IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

This manual has been prepared by the IBM Systems Development Division, Product Publications, Dept. B98, P.O. Box 390, Poughkeepsie, New York 12602. Send comments concerning the manual to this address.

Contents

Operation

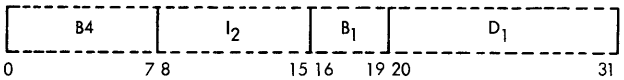
The direct word feature provides for the transfer of a full 32-bit word of information between an external device and the main storage of the system. Its purpose is to permit communication with a variety of non-IBM external devices in the function of data acquisition. It corresponds closely to the direct control feature of System/360, but it has no interrupt lines and also differs significantly in passing a word rather than a byte. Sometimes this feature may be expediently reserved for the passing of control and synchronizing information while the system channels are used for the volume of data, although this is not a restriction.

Information and control signals are exchanged over the direct word interface lines (Figure 1). To effect the transfer of the word of information, two instructions are added to the Model 44 instruction set: WRITE DIRECT WORD to place a word on the direct-out lines, and READ DIRECT WORD to take information from the direct-in lines.

Instructions

Write Direct Word

WRDW $D_1 (B_1), I_2$ *[SI]*



This instruction causes the word at the location designated by the operand address to be fetched from storage and made available as a set of 32 static signals on the direct-out lines. These signals remain available on the direct-out lines until the next WRITE DIRECT WORD is executed. No parity is presented with these 32 data bits.

Also, the eight instruction bits 8-15 are made available as 0.5 to 1 microsecond timing pulses on the signal-out lines. No parity is presented with these eight bits.

Concurrently with the eight signal-out pulses, a 0.5 to 1 microsecond timing pulse is provided on the write-out line. The timing of the signal-out and write-out pulses is such that they overlap the change of the static signals on the direct-out lines.

The operand address must have zeros in its two low-order bit positions, thereby designating that the operand is located on a word boundary; otherwise, a specification exception results in a program interruption.

The eight signal-out lines are utilized also for READ DIRECT WORD.

Time in Microseconds: 3.0 basic, 2.25 with high-speed general registers, using single indexing ($B \neq 0$).

Condition Code: The code remains the same.

Program Interruptions:

Operation (The direct word feature is not installed. The operation is suppressed.)

Privileged Operation (The instruction is encountered with the CPU in the problem state. The operation is suppressed.)

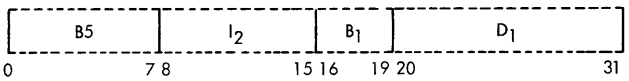
Protection (The operand location is protected for fetching, and the key in storage does not match the protection key in the PSW. The operation is terminated.)

Addressing (The operand location is outside the available main storage of the installation. The operation is terminated.)

Specification (The operand is not located on a word boundary. The operation is suppressed.)

Read Direct Word

RDDW $D_1 (B_1), I_2$ *[SI]*



Unless a signal is present on the hold-in line, this instruction causes a word of data to be accepted from the 32 direct-in lines and placed in the location designated by the operand address. No parity is accepted with the data signals, but four parity bits are generated as the data are placed in storage.

The eight instruction bits 8-15 are made available (unconditionally) as 0.5 to 1 microsecond timing pulses on the signal-out lines. No parity is presented with these eight bits.

Concurrently with the eight signal-out pulses, a 0.5 to 1 microsecond timing pulse is provided on the read-out line.

The hold-in line is tested for the presence of a hold signal after the read-out signal has been completed; if data are to be accepted, the hold signal must be absent for at least 0.5 microsecond. As long as the hold signal is present, the CPU does not complete the operation.

(Excessive duration of the operation may cause timer updating to be omitted.)

The operand address must have zeros in its two low-order bit positions, thereby designating that the operand is located on a word boundary; otherwise, a specification exception results in a program interruption.

The eight signal-out lines are utilized also for WRITE DIRECT WORD.

Time in Microseconds: 4.5 basic, 3.75 with high-speed general registers, plus external delay, using single indexing ($B \neq 0$).

Condition Code: The code remains unchanged.

Program Interruptions:

Operation (The direct word feature is not installed. The operation is suppressed.)

Privileged Operation (The instruction is encountered with the CPU in the problem state. The operation is suppressed.)

Protection (The operand location is protected for storing, and the key in storage does not match the protection key in the PSW. The operation is terminated.)

Addressing (The operand location is outside the available main storage of the installation. The operation is terminated.)

Specification (The operand is not located on a word boundary. The operation is suppressed.)

Programming Note

In the case of a single data source (for example, 32 contact sense points), the hold-in line may be left permanently down. If external logic is provided to switch one of several data sources onto the direct-in lines, the hold-in line is raised to prevent the CPU from sampling the 32 lines while the data are changing and therefore invalid.

Interface between CPU and External Devices

Interface Lines and Signal Timings

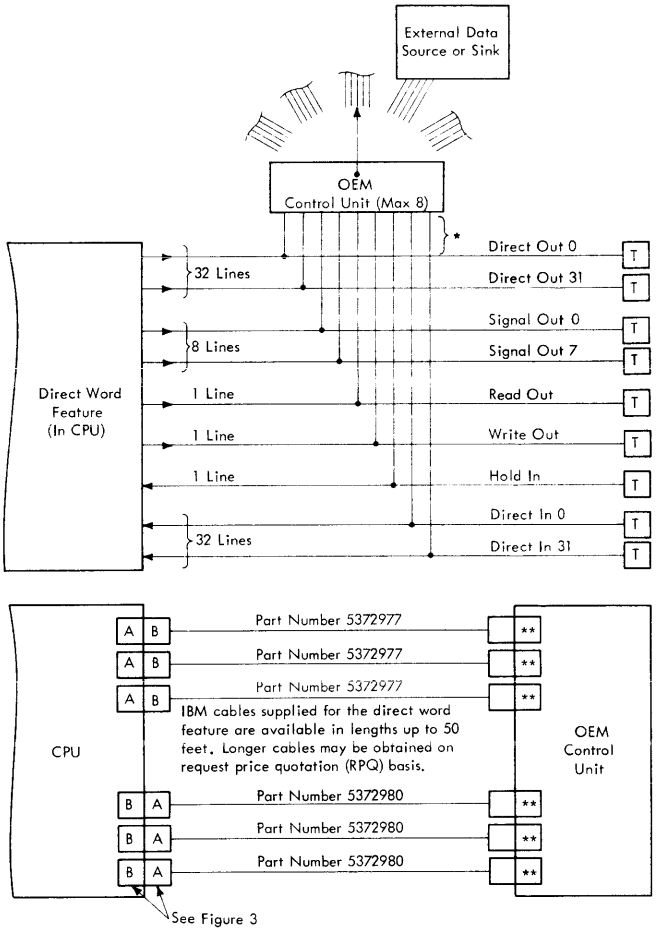
All the interface lines and the cabling necessary to carry them are shown in Figure 1.

The signals on the interface lines, functionally described under "Write Direct Word" and "Read Direct Word," have timings presented in more detail here and in Figure 2. In the following descriptions, the up level is the active level; the down level is the inactive level.

Direct-Out, Write-Out

The down level of the write-out pulse indicates that the static signals on the 32 direct-out lines are no longer changing from old data to new data and are therefore valid for transfer to the external device.

The up level of the write-out pulse overlaps the transition of the signals on the direct-out lines by a minimum of 100 nanoseconds; that is, data already on the direct-out lines are valid for at least 100 nanoseconds after the rise of the write-out pulse to its up level, and new data are valid for at least 100 nanoseconds before the fall of the write-out pulse below its up level.



NOTES:
* Maximum length of stub (connecting line to circuit card) is 6 inches.
** Burndy connector ME23XR-1, mounted on the non-IBM unit, is supplied by the user.
T = Line terminator

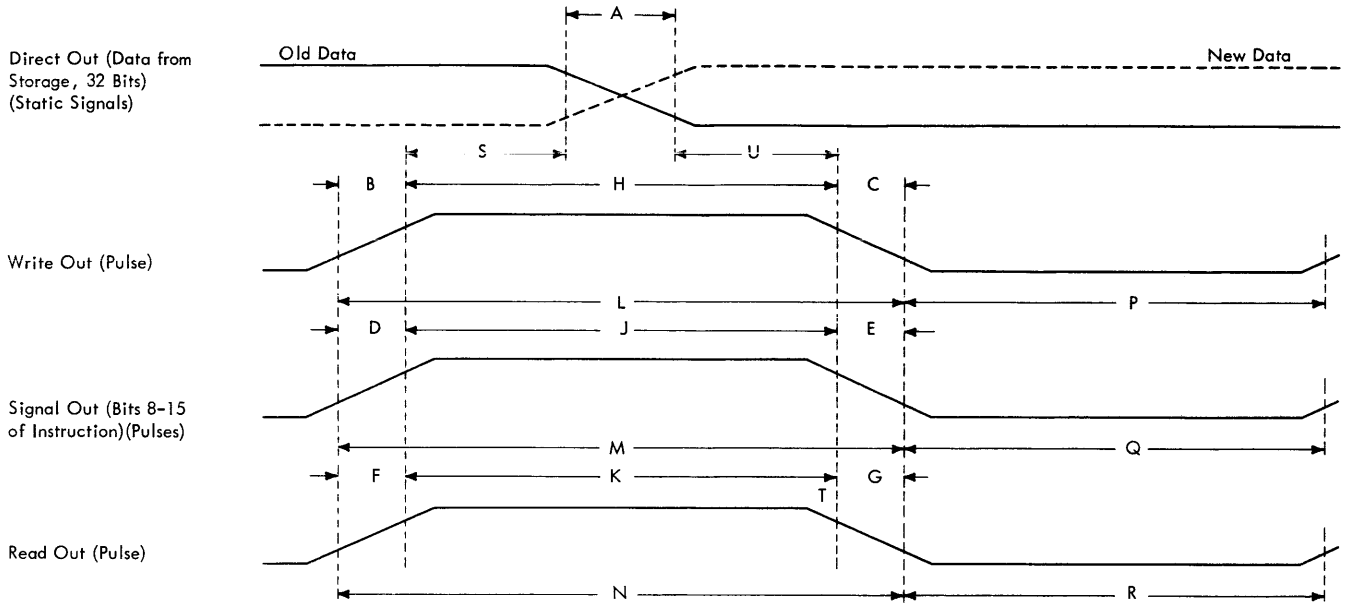
Figure 1. Interface Lines and Cabling for Direct Word Feature

The write-out pulse has a minimum width of 500 nanoseconds and a maximum width (including transitions) of 1,000 nanoseconds. Once the write-out pulse falls, it does not rise for at least 500 nanoseconds.

Direct-In, Hold-In, Read-Out

The down level of the hold-in signal indicates that the static signals on the 32 direct-in lines are no longer changing from old data to new data and are therefore valid for transfer to main storage.

The up level of the hold-in signal must overlap the transition of the signals on the direct-in lines by a minimum of 100 nanoseconds; that is, data already on the direct-in lines must be valid for at least 100 nanoseconds after the rise of the hold-in signal to its up level, and new data must be valid for at least 100



NOTE: Signals shown as seen at output of receiver, that is, inverse to interface signals.

A, B, C, D, E, F, G — Maximum transition time is 200 nanoseconds.

H, J, K — Minimum duration is 500 nanoseconds.

L, M, N — Maximum, including transition, 1,000 nanoseconds.

B, D — Leading edges coincidental within skew tolerances.

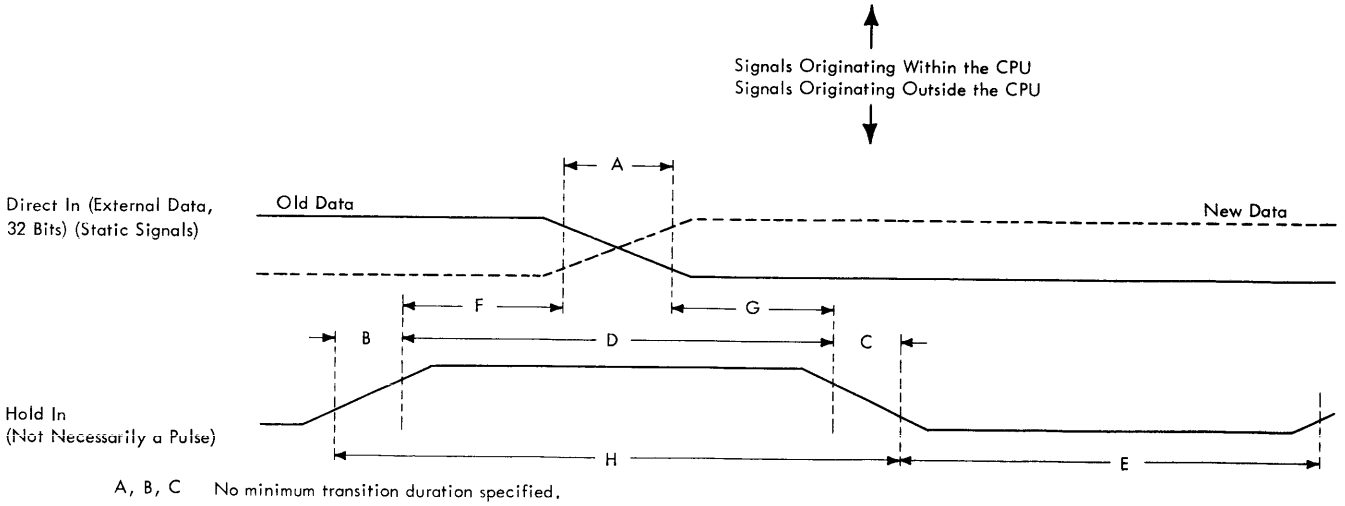
F, D — Leading edges coincidental within skew tolerances.

S — Overlap of write out to start of change A, 100 nanoseconds (min).

U — Overlap of write out to end of change A, 100 nanoseconds (min).

T — Earliest time to sample hold line during read direct word.

P, Q, R — Minimum down time between pulses is 500 nanoseconds.



A, B, C — No minimum transition duration specified.

D, E — Minimum is 500 nanoseconds, no maximum specified (see H).

F — Overlap of hold in to start of change A, 100 nanoseconds (min). No maximum specified.

G — Overlap of hold in to end of change A, 100 nanoseconds (min). No maximum specified.

H — Maximum, including transition, 1,000 nanoseconds.

Figure 2. Direct Word Signal Timing

nanoseconds before the fall of the hold-in signal below its up level.

The read-out pulse has a minimum width of 500 nanoseconds and a maximum width (including transitions) of 1,000 nanoseconds. Once the read-out pulse falls, it does not rise for at least 500 nanoseconds. The sampling of the hold-in line to determine whether the data are valid does not start until the read-out pulse has fallen below its up level (Figure 2). The hold-in signal must have a minimum up-time of 500 nanoseconds and, when at the down level, must remain down for at least 500 nanoseconds. Data transfer from the direct-in lines is completed within 100 nanoseconds after the time that the hold-in signal is determined to be absent.

Signal-Out

During either a WRITE DIRECT or a READ DIRECT WORD operation, the timing pulses on the eight signal-out lines have a minimum width of 500 nanoseconds and a maximum width (including transitions) of 1,000 nanoseconds. The leading edges of these timing pulses coincide (within skew limitations) with the leading edge of either the write-out pulse or the read-out pulse.

Connectors and Pin Assignments

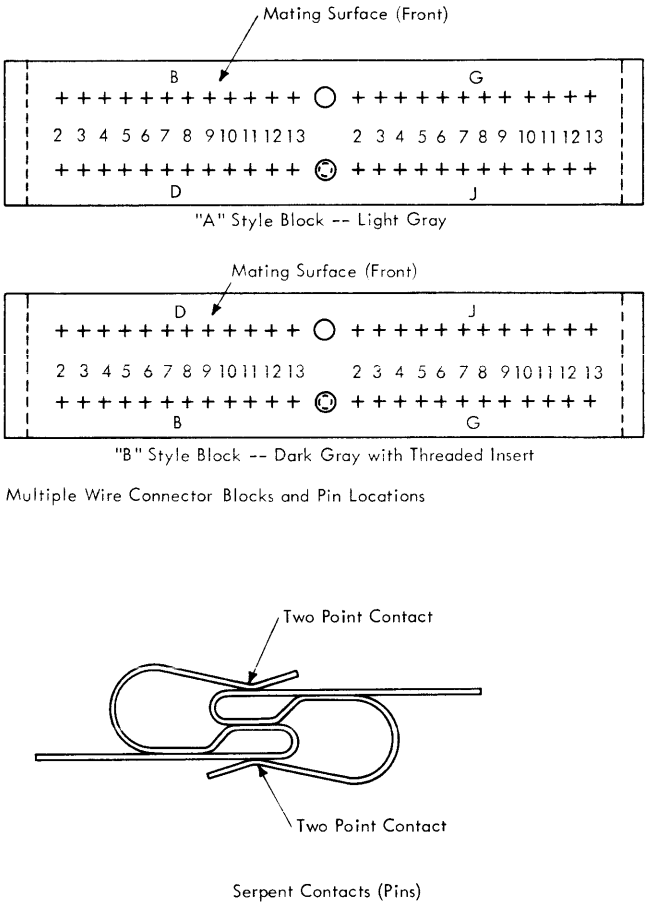
The six cable connectors for the direct word feature, located on the tailgate of the CPU, are of the multiple wire type with serpent contacts. To prevent mismating of connectors, they are color-coded and keyed. Connectors of the same color block must not be mated; the light gray "A" style must mate with the dark gray "B" style. The connectors are keyed to prevent mating of the wrong A and B blocks.

Each connector has 48 serpent contacts (Figure 3); these have two mating surfaces and are gold-plated over phosphor bronze. The connector has a voltage rating of 24 volts AC or DC; each contact has a current rating of 6 amperes, and the contacts are not intended for interrupting current.

The interface lines are assigned to specific connector pins as shown in Figure 4. Each diagram shows two connectors, with pin numbering for both in the left three columns. Thus, direct out 0 has its signal line on pin D4 and its ground shield on pin B4 of the first B style connector, and direct in 0 connects to pins B4 and D4 of connector 2.

Using Direct Word With External Interrupt

Direct word can be used to interconnect two Model 44's; when it is so applied, the external interrupt feature should be used to enable one Model 44 to alert the other for exchange of information. The usual interconnection is completely reciprocal, requiring two direct word features and two external interrupt features.



Multiple Wire Connector Blocks and Pin Locations

Serpent Contacts (Pins)

Figure 3. Connector Blocks and Contacts

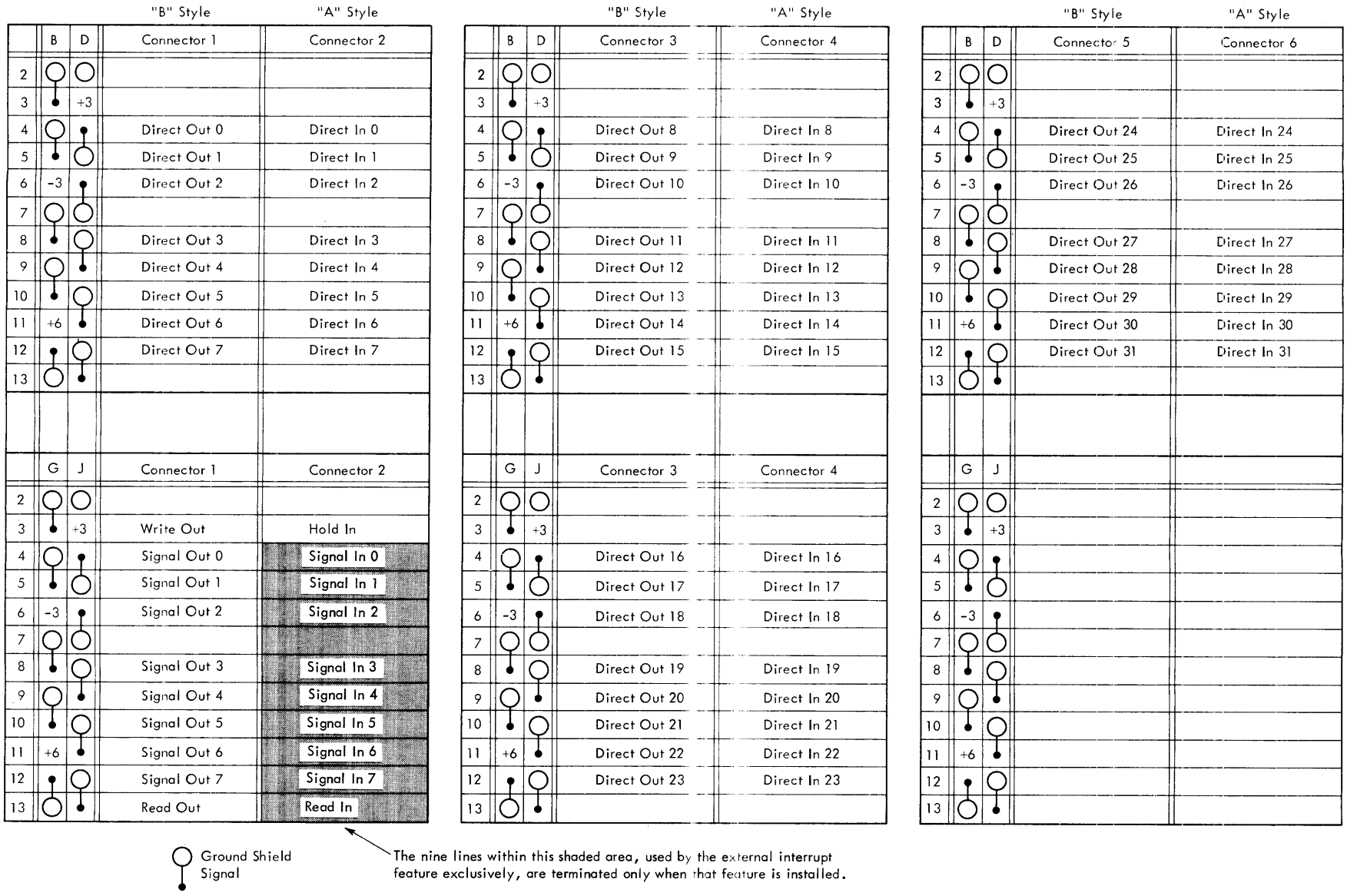
Whenever direct word and external interrupt are installed together, they share connector 2 of the direct word feature. The direct word feature terminates the direct-in and hold-in lines, and the external interrupt feature terminates the signal-in lines.

If external interrupt is not installed on one Model 44, the direct word signal-out lines of the other Model 44 may be disconnected at the output side of the drivers, or electrical termination for the lines must be additionally provided. Any other unusual interconnection requires similar special action.

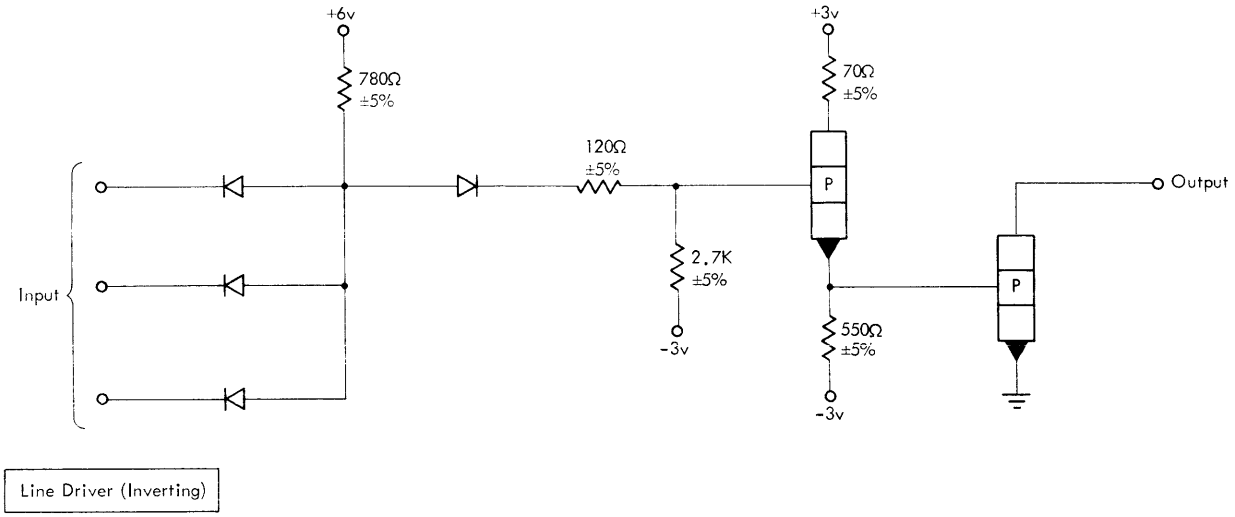
The external interrupt feature is described with the direct control feature; see *IBM System/360 Direct Control and External Interrupt Features OEMI*, Form A22-6845.

Electrical Specifications

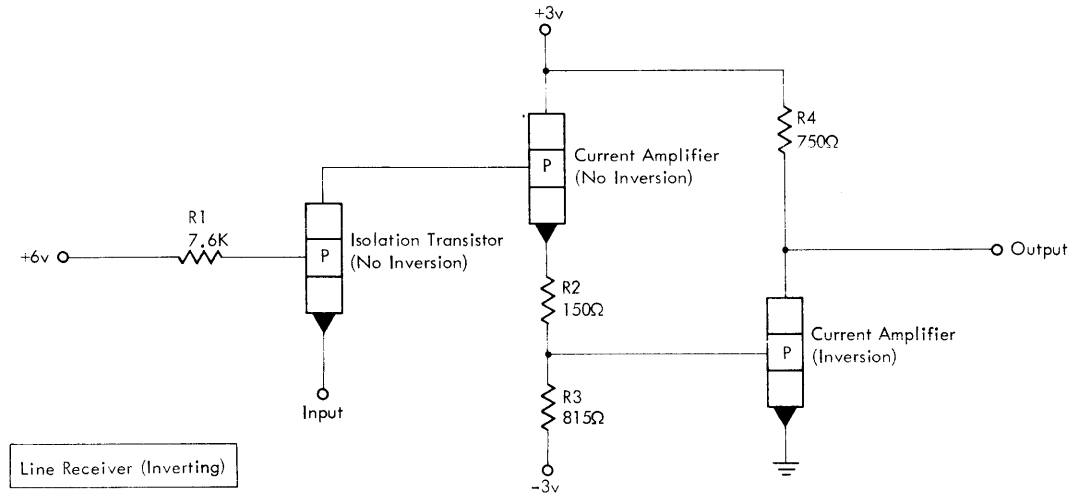
Figure 5 shows the circuits used to drive, receive, and terminate the lines between the CPU and an external device. The direction of current flow (conventional) is plus if it is flowing into a component or minus if it is flowing out of a component. Of the two positive levels on a line, the more positive level denotes a logical 0 and the more negative denotes a logical 1.



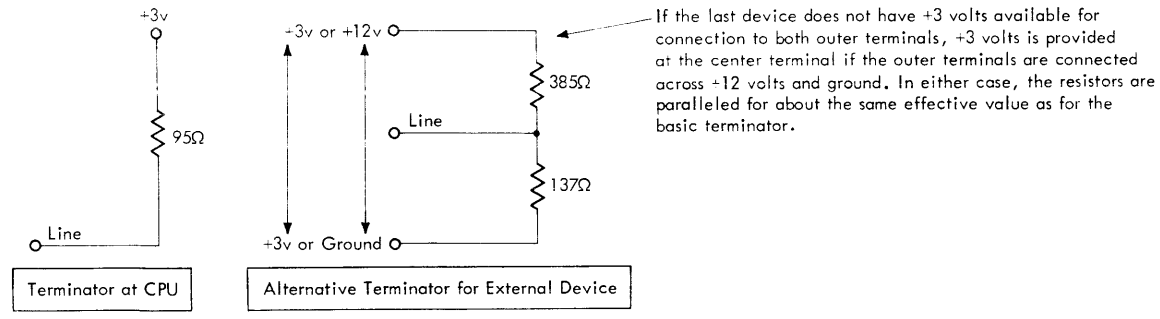
● Figure 4. Pin Assignments for Direct Word Feature



Line Driver (Inverting)



Line Receiver (Inverting)



Terminator at CPU

Alternative Terminator for External Device

Figure 5. Drivers, Receivers, and Terminators for Interface Lines

Line Driver

To transmit a logical 1, the driver conducts; while conducting, it must be capable of accepting +56 milliamperes from a 0.33-volt source.

To transmit a logical 0, the driver must draw less than 100 microamperes from the line.

If the driver is open-circuited while conducting, voltage must not drop below 0 volt.

Line Receiver

The output must be interpreted as a logical 1 for the more negative line signal and as a logical 0 for either a plus line signal or an open input.

Receivers must not require a switching level more positive than +2.58 volts for interpretation as a logical 0, nor more negative than +1.49 volts for a logical 1.

Receivers must not be damaged by a DC up level of 3.4 volts nor by a DC down level of 0 volt.

Receiver input must not require a positive current greater than 0.35 milliampere at the most positive up level of 3.4 volts. Negative current required must not be greater than that taken by an 8.14-kilohm resistor network connected to a 6.24-volt supply.

Input impedance of each receiver should be made as high as possible, and never lower than 4 kilohms.

Line Terminators

The line terminator at the CPU is a two-terminal network consisting of resistors and a power supply. One terminal must be connected to the signal line and the other must have a zero signal impedance to ground. The terminal connected to the signal line must present an open-circuit voltage between +2.88 and +3.4 volts. Impedance between the terminals must not be less than 90 ohms nor greater than 105 ohms. The current is measured at the terminal connected to the cable, and must not be greater than 21.6 milliamperes flowing out of the component at +1.24 volts.

Figure 5 also shows an alternative and equivalent line terminator for the external device.

Multiple Drivers and Receivers

As many as eight receivers may be driven by one driver; the driver must be located at one of the extreme ends of the line between them. As many as nine drivers may be dot-OR'ed to drive one receiver; the receiver must be located at one of the extreme ends of the line between them.

Receivers must not be less than 3 feet apart. No minimum requirement is set for the spacing between drivers. No minimum requirement is set for the spacing between a terminator and driver or receiver if the terminator is placed on the outermost end of the line.

NOTE: An end-of-line driver or receiver may be placed beyond the terminator. In that case, the distance between the end-of-line driver or receiver and the terminator must be less than 6 inches.

Single Driver and Receiver

The driver and receiver must be located at the extreme ends of the line between them. The line need be terminated in its characteristic impedance (see "Line Terminators") only at the receiver end. The distance between the end-of-line receiver and the terminator must be less than 6 inches.

Characteristic Impedance

Lines must have a characteristic impedance of 92 ± 10 ohms. Except as noted under "Multiple Drivers and Receivers" and "Single Driver and Receiver," the lines must be terminated at each extreme end in their characteristic impedance with a network called the terminator. Because of transmission line characteristics, maximum stub length (line to circuit card) is 6 inches.

Fault Conditions

The signal line may be grounded with no damage to the drivers, receivers, or terminators. Loss of power at either end causes no damage.

Loss of power at any terminator may cause random errors in information transmission. Power off at either extreme end of the interface causes unpredictable signal levels which may result in CPU hangup or improper operation of external devices. Where power is off in any intermediate device, line operation is unaffected.

Skew

Skew between any pair of pulses is the time between their leading edges. The allowable skew between any pair of the eight signal-out pulses is less than 200 ns.

Cable Length

Cable length is limited by a maximum cable resistance for any line of 29 ohms between the output pin of a line driver at the CPU and the input pin of the line receiver at any external device, or vice versa.

Contact Resistance: A maximum of 0.22-ohm contact resistance per control unit is permitted. This includes connections to the pins to and from the external cables.

CPU Resistance: The maximum allowable internal cable resistance between the external connector pin and the input pin to the driver or receiver card is 4.5 ohms.

Control Unit Resistance: The maximum allowable internal cable resistance between the external connector pin and the input pin to the driver or receiver card is 1.5 ohms.

The cable may consist of any combination of flat cable, coaxial cable, and printed wire, within the limitation of 1.5 ohms of flat cable and printed wire.

Noise

The maximum noise coupled onto any signal line within a cable, caused by any combination of changes external to that line, must not exceed 250 millivolts.

Operation

The direct data channel provides a very fast but simple data transfer capability for the Model 44. It is a special feature, controlled by standard IBM System/360 input/output instructions and commands, which allows the connection of external devices that perform word-by-word data transfers with the Model 44 CPU at transfer rates up to and including the full processor storage speed of a million words per second (4 million bytes per second).

The direct data channel (DDC) presents a demand-response interface to the external device(s) that allows the half-duplex transfer of parallel 36-bit words (32 data, 4 parity) between the CPU and an external device. The DDC controls this interface and the transfer of words to and from full word boundaries in storage. (Full word addressing is achieved by ignoring the two low-order bits of the data address in the channel command.)

Parity checking is not carried out within the DDC, but the DDC is informed if parity errors occur during the transfer of data into storage (that is, during a read operation).

The first high-speed multiplexor channel (HSMPX) is a prerequisite for the direct data channel. When these two channels are present, the only other channel that can be added is the multiplexor (MPX) channel.

The line characteristics of the interface permit the attachment of as many as eight external devices. Because the interface has no device addressing capability, the attachment of more than one device requires that an acceptable addressing technique be devised. Possible alternatives include:

Manual switching
First word addressing
Model 44 direct word feature

If it is necessary for an attached device to cause an attention or device-end interruption, or alternatively to indicate an overrun condition, this can be achieved, without action by the DDC, by using one or more lines provided by the external interrupt or priority interrupt feature to cause specific action by the program.

I/O Addressing

The method of addressing the direct data channel is as detailed in the "Input/Output Device Addressing" section of *IBM System/360 Principles of Operation*, Form A22-6821. Of the 16 bits (00000 XXX xxxx xxxx) that form the complete address:

1. The first five bit-positions must contain zeros.
2. The next three bits (XXX) are assigned to specify the channel in the normal way:

010 – direct data channel

3. The eight low-order bit positions must contain zeros. When an I/O interrupt occurs, logical 0 bits are stored in positions 24-31 of the old PSW.

Instructions

Start I/O

Commands

The following table specifies which command codes are valid for START I/O instructions addressing the direct data channel. It also indicates those flags that are valid for each command. Where flags are not defined, the corresponding bits in the command are ignored.

COMMAND	CODE	FLAGS
Write	0000 0001	CD CC PCI
Read	0000 0010	CD CC PCI
Read Backward	Invalid	. . . . .
Control*	0000 0011	CD CC PCI
Sense	Invalid	. . . . .
Transfer in channel	XXXX 1000	. . . . .

NOTES

CD Chain data
CC Chain command
PCI Program-controlled interruption
X Bit position reserved—must be 0.

*As indicated by the zeros in bits 2-5 in the command code, the control command is always treated as a no-operation control order.

Resulting Condition Code:

0 I/O operation initiated and channel proceeding with its execution
1 CSW stored
2 Channel busy
3 Channel not operational

Data Addressing

The data address and count fields in the CCW must refer to and generate addresses referring to full word boundaries in storage. Failure to do this will cause erroneous data transfers because the two low-order bits in both of these fields in the CCW are ignored. The consequences differ between read and write operations: Primarily, an incorrect specification in the data

address causes a displacement error, and an incorrect specification in the count field causes a truncation error. A CCW count field of less than four bytes is invalid.

Test I/O

This instruction is valid for only reference to the direct data channel, with the results indicated in the condition code; but otherwise it functions as described in *IBM System/360 Principles of Operation*, Form A22-6821.

Resulting Condition Code:

0 Channel available
1 CSW stored
2 Channel busy
3 Channel not operational

Halt I/O

Execution of this instruction terminates the transfer of data to or from the external device. (See "Command" under "Interface between CPU and External Devices.") If the direct data channel was engaged in a data-transferring operation, it is immediately placed in the interruption-pending state. This condition is not distinguished from the case where the channel operation was ended (by the device or by word count = 0) just prior to the HALT I/O instruction.

Resulting Condition Code:

0 Channel available
1 ————
2 Channel operation terminated; interruption pending in channel
3 Channel not operational

Test Channel

This instruction tests the state of the the DDC and sets an appropriate condition code.

Resulting Condition Code:

0 Channel available
1 Interruption pending in channel
2 Channel working
3 Channel not operational

Channel Status Word

Not all of the 16 possible status bits are presented in a CSW associated with a direct data channel operation. The following are those status bits generated by the DDC to indicate conditions implied by the sequence of events during an operation. The notes column gives any significant indication that is not defined in *IBM System/360 Principles of Operation*, Form A22-6821.

BIT	CONDITION	NOTES
36	Channel end	Indicates the end of an operation.
37	Device end	Indicates the end of an operation.
40	Program-controlled interruption	
42	Program check	
43	Protection check	
44	Channel data check	
45	Channel control check	
46	Interface control check	Indicates a time-out condition following the use of priority in. (See "Priority In" under "Interface between CPU and External Devices.")

Status bits 32-35, 38, 39, 41, and 47 are not used. (These indicate, respectively, attention, status modifier, control-unit end, busy, unit check, unit exception, incorrect length, and chaining check.) To check for a correct length record, the program should inspect the count portion of the CSW after the operation. In command chaining, however, only the final CSW is stored and there is no way of telling whether previous records were of expected length.

Interface between CPU and External Devices

Interface Lines and Signal Timings

Operation between the direct data channel and an external device is effected through the DDC's interface. The interface consists of a set of lines that provide the control signals and data path shown in Figure 6. The signals and data have the sequence shown in Figure 7.

Output Data

The 36 output data lines present, to the external device, data words received from full word boundaries in storage via a 36-bit channel data register. Thirty-two lines carry data bits and four lines carry the odd parity bits (one per byte) as received from storage.

The lines are valid for sampling by the external device only when a write command is indicated and the command and sync-out lines are up, and remain valid until the sync-in signal is received.

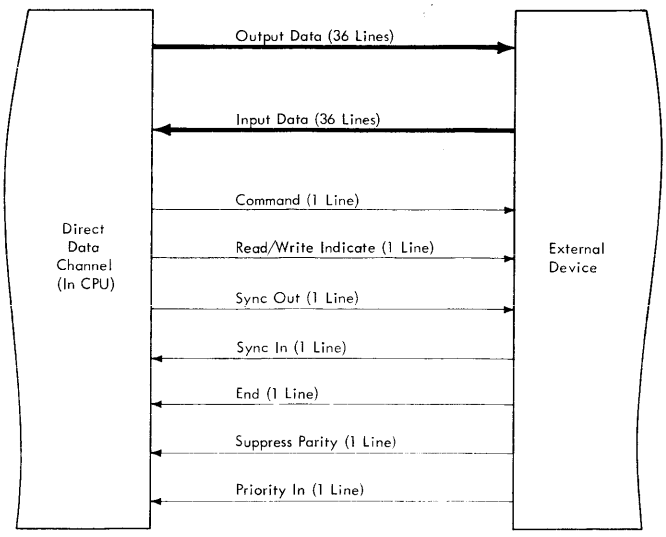


Figure 6. Interface Lines for Direct Data Channel

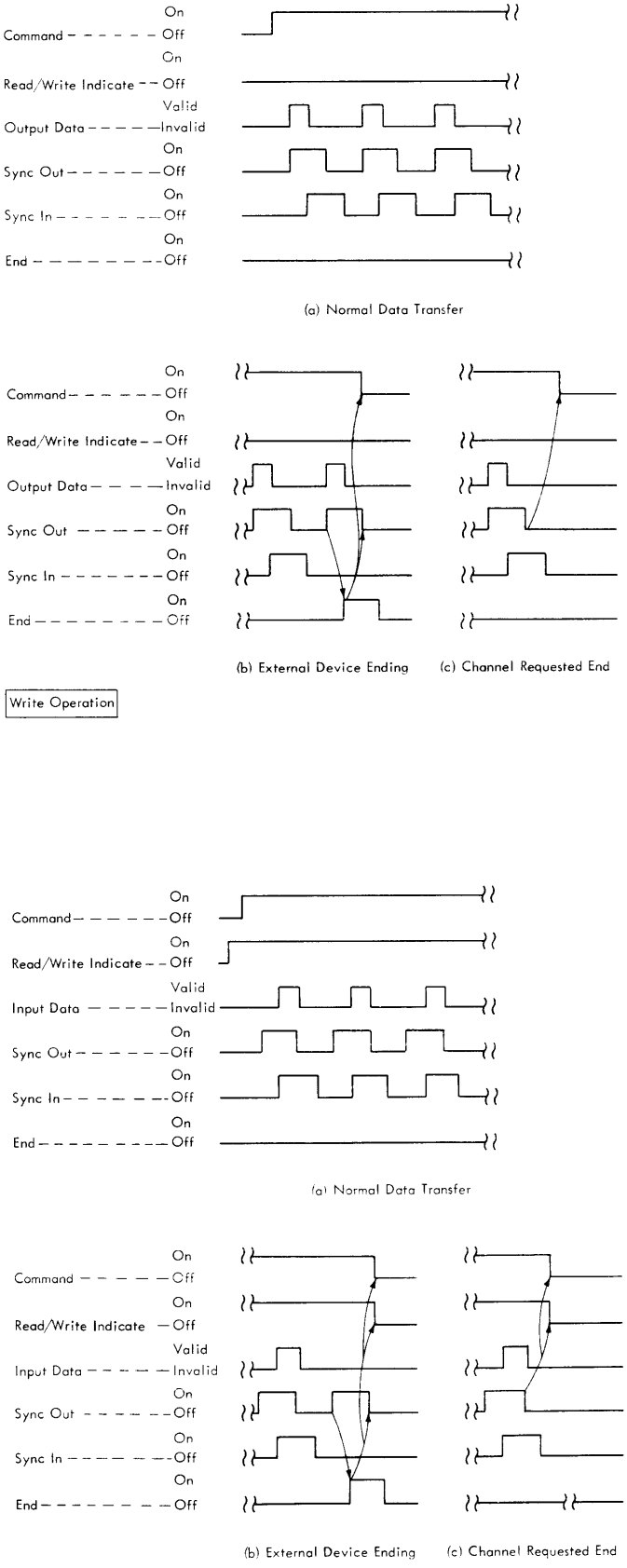


Figure 7. DDC Signal Sequence

Input Data

The 36 input data lines present information from the external device to storage at full word boundaries, via a 36-bit channel data register. Thirty-two lines are used for input data. Four lines are available to transmit odd parity bits (one per byte) when the external device has this capability. A channel data check is caused by the detection of incorrect parity during the transfer into storage unless the suppress parity line is up.

The input data lines are sampled by the DDC when the read/write line indicates a read command and the command, sync-in, and sync-out lines are all up. At this time, the signals on the input data lines remain valid and deskewed at the DDC until the drop of sync-out.

Command

This line rises to indicate to the external device that the channel has decoded a read or write command; it remains up until the end of the operation. The command line drops to inform the external device that the DDC has no more data to transmit or receive.

A channel-end interrupt is generated when the command line drops. When the interrupt is accepted, the associated CSW is stored and the direct data channel is no longer busy.

A HALT I/O instruction can end a transfer at any time by causing the command line to drop.

A time-out, occurring in the operation of sync-out and previously enabled by a signal on the priority-in line, causes the command line to drop immediately after the condition has been detected. In this case, bit 46 (interface control check) will be set in the CSW when the interrupt is accepted.

The condition of word count = 0 ends a transfer by causing the command line to drop at the time that sync-out would have risen again after the last data word was transferred.

A program check or protection check detected during data transfer, or during CCW fetching for data chaining, causes the command line to drop at once.

After a channel data check (bad parity on input data when the suppress parity line is down), good parity is forced in storage but the command line remains up to permit reading to continue.

The external device can end a transfer by raising the end line, but only when sync-out is up. The command line will then drop when sync-out drops.

If command chaining is specified, the down state of the command line (between the end of one operation and the beginning of the next) may last for as little as 2 microseconds, during which the read/write indicate line may change state. External devices must be capable of recognizing this minimum down time.

NOTE: If necessary, external devices can use the lines of a separate interrupt feature to indicate unusual ending conditions. The program must then take specific action.

Read/Write Indicate

The signal level of this line indicates to the external device whether the operation being selected is a read or a write. The line is down for a write operation or up for a read operation.

Sync-Out

The significance of the signal level on this line to the external device depends on the command being executed:

Write Command: The sync-out signal rises to notify the external device that a data word is on the output data lines. The data are stabilized and deskewed at the interface connector when this line is raised. A sync-in signal from the external device resets the sync-out signal.

Read Command: The sync-out signal rises to notify the external device that the DDC is ready to accept a word of data over the input data lines. Sync-out falls to acknowledge that data have been transferred from the input data lines to the DDC. The presence of the data will have been indicated by the rise of sync-in.

Signals Controlling Sync-Out: The sync-out line may rise only when the command line is up and sync-in is down. Conversely, sync-out is reset down by the rise of sync-in or the fall of the command line. The fall of command occurs at the end of a data transfer, including, as causes:

1. Word count = 0
2. A HALT I/O instruction
3. A signal on the end line
4. A time-out condition occurring in the operation of sync-out. (This time-out condition occurs during an I/O operation, after the DDC has detected a signal on the priority-in line during START I/O, if the interval between successive word transfers, as indicated by the rise of successive sync-out pulses, exceeds 75 ± 25 milliseconds.)
5. A program check or protection check detected during data transfer, or during CCW fetching for data chaining.

Sync-In

After a normal START I/O (priority-in line down), there is no time-out; the channel must wait for the rise of sync-in from the device: that is, for the device to accept or offer data. Any time-out must be provided by programming.

The significance of the signal level on the sync-in line depends on the command being executed:

Write Command: The sync-in signal rises in response to a sync-out signal. It notifies the DDC that the external device has accepted a word of data over the output data lines.

Read Command: The sync-in signal rises in response to a sync-out signal. It notifies the DDC that the external device has placed data on the input data lines. The data must be stabilized and deskewed at the interface connector when this line is raised.

Signals Controlling Sync-In: The sync-in line may rise only when the command and sync-out lines are both up. Only the fall of sync-out resets sync-in.

End

This line is used by the external device to signal that it has completed its operation and will not generate or accept more data. The end line is raised, to end the transfer, only when the sync-out and command lines are up.

NOTE: If it is necessary to convey an end-of-file condition to the program, this must be done by some means that requires no specific action on the part of the DDC. A specific bit pattern within the input data, or alternatively the use of an interrupt feature, are possible methods.

Suppress Parity

This line is used by the external device to prevent a channel data check from being caused by invalid parity in the incoming data. Consequently, it is effective only during a read operation; if it is used, it must remain up for the duration of the read operation.

Although data words are always written into main storage with the correct parity, a channel data check results, when the external device presents a data word with invalid parity, if the suppress parity line is down. Accordingly, this line is used on devices that do not generate valid parity; it may be strapped on by the customer if he never uses parity.

Priority-In

The priority-in line from the external device is sampled by the direct data channel during the execution of the START I/O instruction.

If the priority-in line is down, the DDC must share accesses to storage with other channels and the CPU, so that the DDC cannot take consecutive cycles if any other request for service is present.

If the priority-in line is up, the DDC selects and obtains exclusive access to main storage for data transfers after the necessary START I/O functions have been performed. The timer is not updated, and no interrupts are accepted (except machine-check interrupts). If command or data chaining is used, the data transfer rate is as high as 1,300,000 bytes per second; if com-

mand or data chaining is not used, the rate is as high as 1,000,000 words (4,000,000 bytes) per second.

Also, the up level of the priority-in line makes a time-out function effective; namely, if the interval between the rise of successive sync-out pulses exceeds 75 ± 25 milliseconds, the command line falls and thus ends the data transferring operation.

NOTE: In reading from a buffered asynchronous external device, an average data transfer of more than 500,000 *words* per second may be achieved, even when the priority-in line is down. However, certain CPU and channel functions require two consecutive storage cycles, so that the instantaneous rate of the DDC, when priority in is down, cannot be guaranteed above:

250,000 *words* per second if the HSMPX channel is not working.

125,000 *words* per second if the HSMPX channel is working.

Connectors and Pin Assignments

The four cable connectors for the direct data channel feature, located on the tailgate of the CPU, are as described for the direct word feature. The interface lines are assigned to specific connector pins as shown in Figure 8.

Electrical Specifications

The electrical requirements for the direct data channel interface are as described for the direct word feature.

	B	D	"A" Style Connector 1	"A" Style Connector 2
2	○	○		
3	●	+3	Data Out ØP	Data Out 2P
4	○	●	Data Out 0	Data Out 16
5	●	○	Data Out 1	Data Out 17
6	-3	●	Data Out 2	Data Out 18
7	○	○		
8	●	○	Data Out 3	Data Out 19
9	○	●	Data Out 4	Data Out 20
10	●	○	Data Out 5	Data Out 21
11	+6	●	Data Out 6	Data Out 22
12	●	○	Data Out 7	Data Out 23
13	○	●	Command	Sync Out

	G	J	Connector 1	Connector 2
2	○	○		
3	●	-3	Data Out 1P	Data Out 3P
4	○	●	Data Out 8	Data Out 24
5	●	○	Data Out 9	Data Out 25
6	-3	●	Data Out 10	Data Out 26
7	○	○		
8	●	○	Data Out 11	Data Out 27
9	○	●	Data Out 12	Data Out 28
10	●	○	Data Out 13	Data Out 29
11	+6	●	Data Out 14	Data Out 30
12	●	○	Data Out 15	Data Out 31
13	○	●	Read/Write Indicate	

	B	D	"A" Style Connector 3	"A" Style Connector 4
2	○	○		
3	●	+3	Data In ØP	Data In 2P
4	○	●	Data In 0	Data In 16
5	●	○	Data In 1	Data In 17
6	-3	●	Data In 2	Data In 18
7	○	○		
8	●	○	Data In 3	Data In 19
9	○	●	Data In 4	Data In 20
10	●	○	Data In 5	Data In 21
11	+6	●	Data In 6	Data In 22
12	●	○	Data In 7	Data In 23
13	○	●	End	Suppress Parity

	G	J	Connector 3	Connector 4
2	○	○		
3	●	-3	Data In 1P	Data In 3P
4	○	●	Data In 8	Data In 24
5	●	○	Data In 9	Data In 25
6	-3	●	Data In 10	Data In 26
7	○	○		
8	●	○	Data In 11	Data In 27
9	○	●	Data In 12	Data In 28
10	●	○	Data In 13	Data In 29
11	-6	●	Data In 14	Data In 30
12	●	○	Data In 15	Data In 31
13	○	●	Sync In	Priority In

○ Ground Shield
● Signal

● Figure 8. Pin Assignments for Direct Data Channel Feature

Operation

The Model 44 is highly efficient in real-time applications wherein time measured in microseconds is of special significance. Such applications include both high-speed data acquisition and the control of experimental systems; in the second of these, the computing power of the CPU needs to be coupled with the ability to respond quickly to external events, and to distinguish between many different external situations with a minimum of programming. These abilities can be provided with the special feature called priority interrupt.

The 32 levels of the priority interrupt feature are supplemental to the single-level, six-line external interrupt feature. Either feature may be installed alone, or the two features may be installed together; they are logically and physically independent.

Every function provided by the priority interrupt feature can be achieved by programming with the external interrupt feature, but with slower response time. The priority interrupt feature is faster because of:

1. Including more interrupt levels and, therefore, the feasibility of nesting response routines one within another. This provides for the interruption of one routine by another of higher priority.

2. Providing an immediate physical branching to a very large number of response routine programs. At each of the 32 levels, as many as 256 external conditions can be identified, with direct branching to the distinct storage addresses.

Advanced System Requirements

Priority interrupt systems for some applications can become highly complex. It may be necessary to distinguish among, and provide distinct responses to, an unusually large number of external events. The priority may need to be changed from time to time; moreover, the assignment of priority may be made time-dependent (for example, the longer the delay before a certain event occurs, the higher the priority assigned to the response). All these functions can be achieved either by programming or, more rapidly, by the priority interrupt feature, which has enough external connections to enable the advanced user to add more sophisticated controls outside the CPU.

Compatibility with Other System/360 Models

The priority interrupt mechanism can be fully disabled and enabled under program control. It is also disabled as part of the system reset procedure; therefore, real-time programs calling for priority interruptions must specifically enable them. This is normally done as part of the post-IPL initialization.

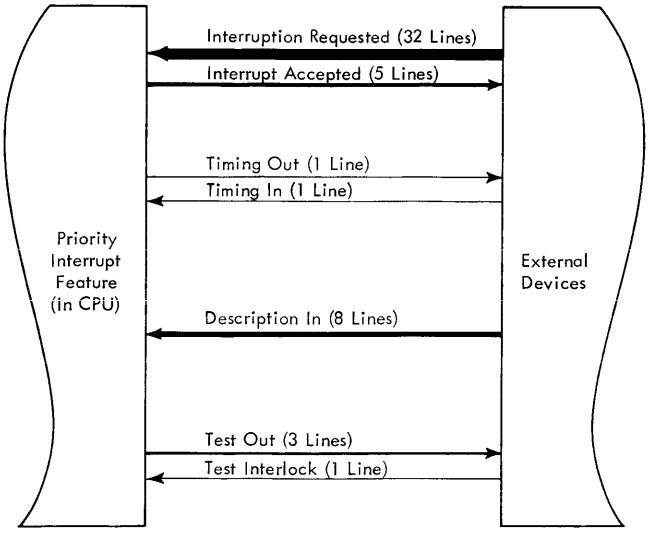
When the priority interrupt mechanism is fully disabled, the Model 44 is architecturally compatible (in Model 44 instruction subset terms) with other models of the System/360.

The instruction to enable one or more priority interrupt levels is a privileged instruction normally invalid for System/360; consequently, if a supervisor program which does not support priority interrupts is loaded into the Model 44 by means of the IPL procedure, that supervisor is effectively protected against interference from priority interrupt signals.

Signal Exchanges

The external interface of the priority interrupt feature consists of the 51 lines shown in Figure 9.

A signal on any of the interruption-requested lines is taken as a request for a CPU interruption at the indicated level. Several requests may be pending at one time. When the CPU takes an interruption (after first satisfying priority and masking conditions), it identifies, at the external interface, which level it is taking, by placing the binary coded value of the in-process



● Figure 9. Interface Lines for Priority Interrupt Feature

interruption on the interrupt-accepted lines and giving the timing-out signal.

The user may supply eight bits of data, for use (1) as data, or (2) to describe the reason for the interrupt, or (3) to call for sublevel programming routines. In any case, this is done by placing a byte on the eight description-in lines and giving the timing-in signal. If such description is not needed, the timing-in signal is still required by the CPU; at its simplest, the timing-in signal may be created by a jumper from timing-out back to timing-in.

In summary, if the user wants to supply data with the interrupt or to extend the interrupt system beyond the 32 unique interrupts, he responds to timing-out by first decoding the interrupt-accepted lines, next placing a byte on the description-in lines, and then raising the timing-in line.

Whenever the CPU detects the rise of timing-in, it samples the eight description-in lines, stores any byte so obtained in the old PSW associated with the interruption request, and may (if so programmed) use the byte to modify the instruction address of the new PSW.

Interrupt Levels

Thirty-two interrupt levels are defined, in addition to the five basic interrupt levels of the System/360 (external, supervisor call, program, machine-check, and input/output). The old and new PSW's for each level are interleaved in a 512-byte block of main storage; for a priority level of n, the old PSW is assigned to storage location (2048 + 16n) and the new PSW is assigned to (2056 + 16n):

DECIMAL ADDRESS	HEXADECIMAL ADDRESS	PURPOSE
2048	800	Level 0, old PSW
2056	808	Level 0, new PSW
2064	810	Level 1, old PSW
2072	818	Level 1, new PSW
2080	820	Level 2, old PSW
↓	↓	↓
2544	9F0	Level 31, old PSW
2552	9F8	Level 31,new PSW

Each level is permanently associated with one of the interruption-requested lines. The requested interruption is taken, if the priority can be honored and if enabled by the program, either at the end of the current CPU instruction or (if the CPU is in the wait state) immediately.

The format of the PSW is described in *IBM System/360 Principles of Operation*, Form A22-6821. The interruption code of the old PSW, stored in its bit positions 16-31 as a result of a priority interrupt, is:

```
                 01234567   bit positions on description-in lines
 00000000        bbbbbbbb   signals on description-in lines
16       23 24         31
```

Modifying the Instruction Address

Bits 16-23 of the new PSW constitute a mask for controlling a change in the instruction-address bits 54-61 fetched *from* the new PSW. (The contents of the new PSW in storage are not changed.) Note that bits 54-61 specify a word rather than a byte. The mask controls the change in the instruction address as follows:

Zero: If a bit of the mask is 0, the corresponding bit from the instruction address is used unchanged.

One: If a bit of the mask is 1, the presence of a 1 on the associated description-in line causes the corresponding instruction-address bit to be forced to 0.

EXAMPLE		FOR CORRESPONDING BITS:
Mask	00110011	← If this bit is 1
Description-in bits	00001111	← And this bit is 1
I-address fetched	01010101xx	← Then this bit, if 1
Address from which next instruction is fetched, per change in effective I-address	01010100xx	← Is forced to 0

Roles of the Description-In Bus

The following cases show how data on the description-in lines can be used to perform a variety of distinct functions; they also demonstrate the practical function of the address modification mask.

Example 1: Four distinct external conditions occur in such a way that the simultaneity of any two or more of them has special significance.

Let the occurrence of any condition cause an interruption request at level 10. When the CPU signals that level 10 has been accepted for interruption, let the conditions that have occurred be presented to the CPU as bits 2-5 of description-in, with bits 18-21 of the level 10 new PSW (the mask) set to 1 and bits 56-59 of the level 10 new PSW (part of the instruction address) set to 1.

Then, the first instruction of the response routine will be fetched from one of 15 different addresses, according to the 15 different combinations of four events (excluding 0) which may have occurred. By choosing bits 2-5 of description-in, corresponding to bits 56-59 of the new PSW instruction address, the first instructions of the response routines will be located at 16-byte intervals in storage.

Example 2: Eight distinct external events occur that are logically unrelated; the system response to each event is unaffected by the occurrence of any of the other seven events.

Let the occurrence of any one of the events cause an interruption request at level 12. Let some program housekeeping be required that is common to all eight response routines; consequently, let bits 16-23 of the level 12 new PSW (the mask) be all zero bits, so the instruction address of the new PSW is not modified.

Then, occurrence of any of the eight events, alone or in combination, leads to a single common housekeeping program from which, by programmed testing of bits 24-31 of the level 12 old PSW, the system may call, serially, the separate response routines that may be requested.

Example 3: The system is required to count events that are signalled by pulses on a single line; the interval between successive pulses may be only a few microseconds. Because the Model 44 may take several microseconds to respond to a priority interrupt signal, it is necessary to provide external circuits to safeguard against the loss of a pulse.

Let the occurrence of a pulse cause an interruption request at level 4. Let an external electronic counter add one for each successive pulse, to a maximum of 16 pulses. On acceptance of the level 4 interrupt by the CPU, let the contents of the counter be placed on description-in, and let the counter be subsequently reset.

Then, if the program has set bits 16-23 of the level 4 new PSW (the mask) to 0, the first instruction of the response routine will be fetched without address modification, and the response routine can add the four-bit value of the external count as stored in the level 4 old PSW.

Priority and Masking of Interrupts

Priority interrupts are accepted, and the interruptions taken, under the control of a priority scheme that is fixed in circuitry and supplemented by a program control over the enabling or disabling of specific interrupts. This enabling or disabling of the interrupts is termed program-controlled masking, but has an entirely separate function from that of the address modification masking just described.

Whenever a priority interruption is taken (that is, when old and new PSW's are exchanged in response to an outstanding request), an in-process latch is set for that level. This latch is reset only by the execution of a variant of the LOAD PSW instruction, LOAD PSW SPECIAL, which must be used to signal that a particular response routine has been completed.

When interruptions at two or more levels are requested at the same time, the lowest enabled value is taken; thus, level 0 has the highest priority. When an in-process latch is set for any level, no new priority interrupt is accepted at that or any lower priority level. Higher priority interrupts may be accepted, and the interruptions taken; moreover, the five basic System/360 interruptions may be taken, if they are not masked in the current PSW. (For clashes between priority interrupts and the five basic interrupts, see "Conflicts between Priority Interrupts and Basic System Interrupts.")

When the response routine for a given level is interrupted by a higher priority level from the priority interrupt feature, the in-process latch for that higher level is also set. At the conclusion of the highest priority response routine, its in-process latch is reset and the next CPU activity (if no higher priority interrupts intervene) is the next lower priority routine that was interrupted, at the end of which its in-process latch is also reset. This sequence continues until all interruptions are cleared. When all in-process latches are off, LOAD PSW SPECIAL has the same effect as LOAD PSW.

The program-controlled masking is effected by a 32-bit priority mask register whose contents can be changed by a privileged instruction, CHANGE PRIORITY MASK. Each priority level has an associated bit; if the bit is 1, the interrupt is enabled, and if the bit is 0, the interruption request is kept pending. The same instruction may be used to selectively cancel any interruption requests waiting at the levels, or to enable any priority levels while cancelling previous requests at the same levels. Thus an interruption request may be:

Accepted in its order of priority
Cancelled
Held off and later accepted (in priority)
Held off and later cancelled

An interruption request is honored when all three of the following conditions exist (without all three, the interruption request is kept waiting until specifically cancelled or a system reset occurs):

1. The mask bit for that level is set to a 1.
2. No request of higher priority is also enabled.
3. No interruption of the same or any higher level is in process.

The contents of the priority mask register are made 0 by a system reset (and thus by IPL). The CHANGE PRIORITY MASK instruction fetches a 32-bit word from storage and uses it to perform either an OR or an AND function on the contents of the register, thus providing new contents. The OR function permits selective enabling of any one or more interrupt levels, whereas the AND function provides the selective disabling. (See "Instructions.")

Programming Notes

The fixed priority scheme and the programmed masking are mutually independent. The priority mask register need not be changed as part of any response routine; the fixed priority scheme protects the program against re-entry at the same interrupt level — a risk that would be present if only a programmed mask were provided.

The programmed mask permits a slight reassignment of priorities, either:

1. By changing the assignment of an external condition from its assigned priority to no priority at all, or

2. By reassigning (switching) an external condition between two fixed priority levels, if the condition is connected to two interruption-requested lines.

The use of logical OR/AND functions to enable/disable interrupt levels means that the programmer who wants to define the masked state of any level does not need to know the previous enabled/disabled state of that or any other level.

Conflicts between Priority Interrupts and Basic System Interrupts

When a priority interruption is requested and enabled at the same time that one of the five basic System/360 interruptions is requested and enabled, the resolution is:

Concurrence with Machine-Check, Program, or Supervisor-Call Interrupt: The current PSW is stored in the old PSW location for the machine-check, program, or supervisor-call interruption. The new PSW for the machine-check, program, or supervisor-call interruption is fetched and immediately stored in the old PSW for the priority interruption, which is then taken. In effect, the 4-microsecond delay to set up for the machine-check, program, or supervisor-call interruption is added to the preceding instruction time before the priority interruption can be taken.

Concurrence with External or I/O Interrupt: The current PSW is stored in the old PSW location for the priority interruption, which is then taken. The external or I/O interruption will follow, if enabled by the new PSW loaded from the priority interrupt location.

Programming Notes

The action of storing an old PSW and loading a new PSW may change the enabled/disabled state of the system for machine-check, external, and I/O interrupts, but this action does not change the enabled/disabled state for priority interrupts.

If the high-resolution timer is installed, all priority and other interruptions (except machine-check and program interruptions) — requested during the course of an instruction that addresses the timer word — are delayed until the end of the next following instruction. This delay does not occur if only the standard timer is installed.

Instructions

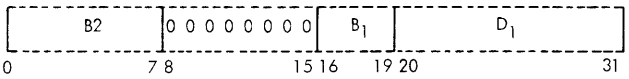
For programming control of the priority interrupt feature, two instructions are added to the Model 44 instruction repertoire:

LOAD PSW SPECIAL to signal the end of a response routine, or for that routine to yield control to a priority interrupt control program, and

CHANGE PRIORITY MASK to enable or disable interruption requests at any one or more priority levels, and/or cancel previous interruption requests at one or more levels.

Load PSW Special

LPSX $D_1(B_1), I_2$ ***[SI]***



This instruction has two effects: the double word at the location designated by the operand address replaces the PSW, and the highest priority in-process latch that is currently on is reset.

The operand address must have zeros in its three low-order bit positions (thereby designating that the operand is located on a double-word boundary) and bit positions 8-15 of the instruction must contain zeros; otherwise, a specification exception results in a program interruption.

The double word that is loaded becomes the new PSW for the next sequence of instructions. Bits 8-11 of the double word become the new protection key, and bits 40-63 become the new instruction address. The new instruction address is not checked for available storage nor for an even byte address during the load PSW special operation; similarly, the protection key is not checked for zero when the protection feature is not installed. These checks occur as part of the execution of the next instruction.

One in-process latch is associated with each interrupt level of the priority interrupt feature. Level 0 has the highest priority, with priorities descending to level 31. An in-process latch is turned on whenever the interruption at the corresponding level occurs. When the latch is on, further interruptions at the same or any lower priority levels are suspended. When all in-process latches are off, LOAD PSW SPECIAL has the same effect as LOAD PSW.

The interruption code in bit positions 16-31 of the new PSW is not retained as the PSW is loaded. When the PSW is subsequently stored because of an interruption, these bit positions contain a new code. Similarly, bits 32 and 33 of the PSW are not retained upon loading. They will contain the instruction length code for the last interpreted instruction when the PSW is stored during a branch-and-link operation or during a program or supervisor-call interruption.

Time in Microseconds: 4.5 basic, 3.75 with high-speed general registers, using single indexing ($B \neq 0$).

Condition Code: The code is set according to bits 34 and 35 of the new PSW loaded.

Program Interruptions:

Operation (The priority interrupt feature is not installed. The operation is suppressed.)

Privileged Operation (The instruction is encountered with the CPU in the problem state. The operation is suppressed.)

Protection (The operand location is protected for fetching, and the key in storage does not match the protection key in the PSW. The operation is terminated.)

Addressing (The operand is outside the available main storage of the installation. The operation is terminated.)

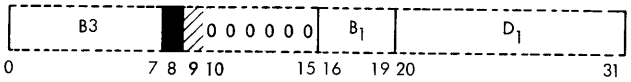
Specification (The operand is not located on a double-word boundary, or bit positions 8-15 in the instruction do not contain zeros. The operation is suppressed.)

Programming Note

It is important for each interrupt response routine to end with this instruction, which is the only way of resetting the latch. Indiscriminate use of LOAD PSW and LOAD PSW SPECIAL could cause the programmer to lose control of the enabled/disabled state of the system.

Change Priority Mask

CHPM $D_1(B_1), I_2$ ***[SI]***



Summary of Functions:

BIT 8 (mask bit)	BIT 9 (cancel bit)	FUNCTION
1	0	Enable levels tagged by 1 bits
0	0	Disable levels tagged by 0 bits
0	1	Cancel levels tagged by 1 bits
1	1	Cancel and enable levels tagged by 1 bits

Bits 8 and 9 determine whether this instruction enables or disables priority levels; and whether previous interruption requests, waiting to be enabled and/or accepted, are selectively cancelled. A word is fetched from storage; bits 0-31 of the word correspond to priority levels 0-31; each level is tagged for the operation by the logical state of its bit in the word.

When bit 8 is a 1, the contents of the word at the location designated by the operand address are OR'ed with the contents of the priority mask register and the result (logical sum) replaces the previous contents of the priority mask register. In this way, any one or up to the natural maximum of 32 interrupt levels can be selectively enabled. The interrupt levels to be enabled are tagged by logical 1 bits in the fetched word; a word of all 1's would enable all levels.

When bits 8 and 9 are 0, the contents of the word at the location designated by the operand address are AND'ed with the contents of the priority mask register and the result (logical product) replaces the previous contents of the priority mask register. In this way, any one or up to the natural maximum of 32 interrupt levels can be selectively disabled. The interrupt levels to be disabled are tagged by 0 bits in the fetched word; a word of all 0's would disable all levels.

When bit 9 is a 1, old interruption requests waiting at the levels tagged by 1 bits in the fetched word are cancelled. New interruption requests are, as before, either accepted in the order of priority for levels enabled or held pending for levels not yet enabled.

The operand address must have zeros in its two low-order bit positions, thereby designating that the operand is located on a word boundary; otherwise, a specification exception results in a program interruption.

Bits 10-15 of the instruction are ignored.

Condition Code: The code remains unchanged.

Time in Microseconds: 3.25 basic, 2.50 with high-speed general registers, using single indexing ($B \neq 0$).

Program Interruptions:

Operation (The priority interrupt feature is not installed. The operation is suppressed.)

Privileged Operation (The instruction is encountered with the CPU in the problem state. The operation is suppressed.)

Protection (The operand location is protected for fetching, and the key in storage does not match the protection key in the PSW. The operation is terminated.)

Addressing (The operand address is outside the available main storage of the installation. The operation is terminated.)

Specification (The operand is not located on a word boundary. The operation is suppressed.)

Programming Notes

Although bit positions 10-15 of the instruction are ignored during the execution, they should contain zeros. These bit positions may be used in the future to specify other functions, in which case the presently assigned functions determined by bits 8 and 9 are assured only when bits 10-15 are 0.

A variant of the DIAGNOSE instruction, intended for use in diagnostic checkout, can cause interruption request latches to be selectively set. If spurious priority interruptions occur during program debugging, the possibility of an erroneous DIAGNOSE instruction should be considered.

Systems Programming With Priority Interrupts

The Model 44 priority interrupt feature requires a resident control program that is structured differently from conventional nonreal-time control programs.

In essence, a control program may be defined as the code that permits two or more tasks, running asynchronously in the system, to obtain access to common system facilities. A Model 44 has, of course, a certain complement of channels and input/output devices, a finite core storage size, and a single timer. Because many priority programs will require the use of these facilities, a control program is necessary to resolve contention for them.

In nonreal-time systems, it is common practice for the control program to protect its facilities from multiple use by disabling all interrupt sources whenever the control program is active. In a true priority-interrupt system, however, the disabling of interrupts for reasons only of internal housekeeping is inconsistent with the system objective of fast response. The priority mask register of Model 44 is provided to allow the applications programmer to ignore external signals when it suits the application; it is not intended for, nor is it capable of, disabling priority interrupts for the convenience of the control program.

It is fundamental to the design of the priority interrupt feature that priority interrupts are not disabled except at the express wish of the applications programmer. It follows that the response routine initiated by a priority interrupt is in full command of the system, not subordinate to a master control program.

Logically, a Model 44 with the priority interrupt feature contains as many as 33 "master" programs in coexistence, one for each priority level plus one background supervisor. (See Figure 10.) Although the background supervisor may control background batch processing, its chief task is to control the programs that are called in real time by priority interrupts, but whose execution is delayed by reason of waiting for a busy facility to become free.

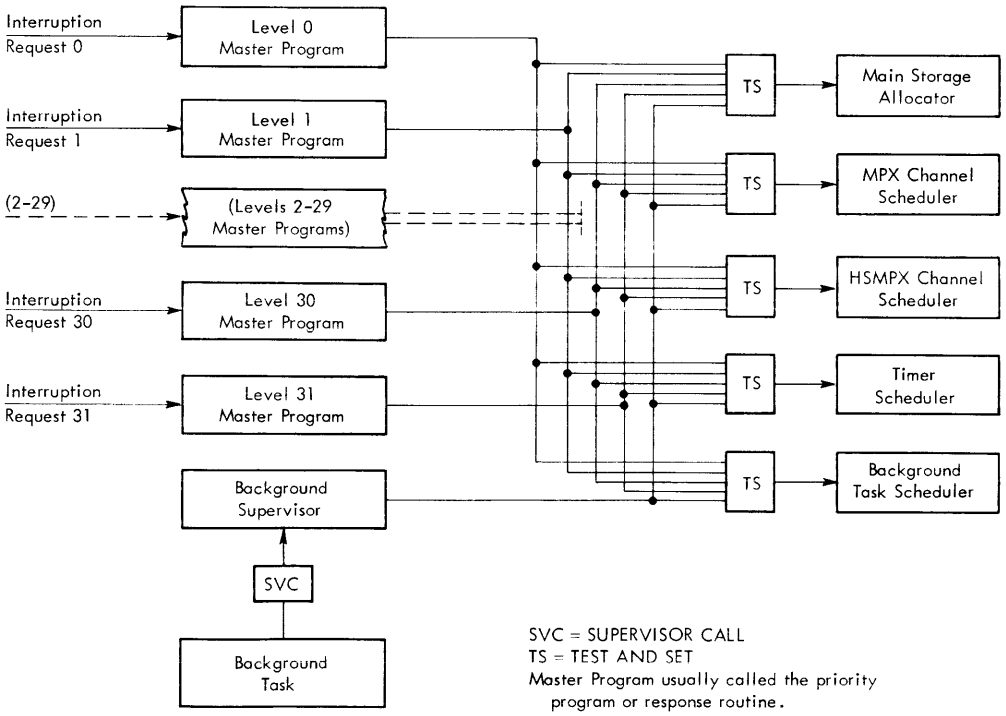
With the possibility of 33 independent master programs residing in the system, it is essential that each such program observe conventions when using common facilities. The machine-language code that enforces these conventions is, in effect, the *priority interrupt control program* for which the following considerations are recommended.

Facility Status Preservation

Each priority program preserves the status of those system facilities intended to be used by it (including one or more general registers). A common re-enterable subroutine may be provided for this purpose.

Facility Testing; Request Queuing

A priority program needing access to a common facility is normally required to test whether that facility is in use at the time, with the TEST AND SET instruction. If the facility is available, the priority program may exercise that facility. If the facility is busy, the priority program may make an entry in a queue associated with the facility, and must yield control with the LPSX instruction.



● Figure 10. Example Structure of a Priority Interrupt Control Program

If a common subroutine exercises that facility, and if this subroutine is of a kind that is serially reusable rather than re-enterable, it must not only be protected by TEST AND SET; on completion it must execute all queued requests that were recorded by higher priority programs while it was in use.

In this environment, the priority level at which access to a shared facility is obtained is either the calling program's level if the facility is free, or the using program's level if the facility is in use.

Example: A level 20 program has found that the multiplexor channel control subprogram is free and has called it at level 20 to execute an I/O operation. This subprogram is briefly interrupted by a level 10 program that needs access to the same subchannel (or same device, or section of nonre-enterable code). The level 10 program makes an entry in a queue — finding and reserving a position in the queue by means of TEST AND SET — and yields control with the LPSX instruction. When level 20 regains control, the multiplexor channel control subprogram is resumed and completes execution of the request made by level 20; then, this subprogram checks its queue, and proceeds to execute the queued request from level 10 before linking back to the level 20 program that called it.

Queuing of requests is necessary only where two or more priority programs call for a single facility within the period of time required to exercise the facility. Priority programs that do not use the same facilities conflict only wherever the application system designer, in assigning the external signals to priority levels, has necessarily given processing priority to one program over another.

Scheduling with I/O and Timer Interrupts

Input/output and high-resolution timer interrupts occur at the priority of the background supervisor. (See note below.) Therefore, a priority program that initiates an I/O operation (and is to yield control until the I/O operation is complete) should schedule the resumption of its processing under the background supervisor. The same is true for a priority program that sets up a timer interrupt. The background supervisor may be designed with an internal priority structure for performing these deferred tasks.

NOTE: The background supervisor has no corresponding in-process latch and is interruptible by all 32 priority programs.

Access to Time

To provide priority programs with access to the time of day with a resolution commensurate with that of the high-resolution timer (13 microseconds), provision can be made for a fully re-entrant subprogram to read the real time without the need for queuing.

Assume that the timer is to be used as a source of timed interrupts, and that the common scheduler of timed interrupts maintains:

1. In location 84 the time at which the next interrupt is to occur.
2. In location 80 (the timer word) the decrementing interval before which the interrupt is to occur.

Under these assumptions, real time is equal to the contents of location 84 less the contents of location 80.

If general register 1 contains the time at which a new interrupt is wanted and general register 0 is available for use, the common scheduler of timer interrupts (which cannot contain fully re-enterable code) can change the values of locations 80 and 84 as follows:

INSTRUCTION		EXAMPLE VALUES (rough)
(LOAD)	LR 0, 1	To be 10:57
Copy new interrupt time into reg. 0		
(SUB. LOG.)	SL 0, 84	Was to be 11:00
(ADD LOG.)	AL 0, 80	:05 in timer
Compute interval between current real time and the time new interrupt is wanted. (Example is 10:57 −11:00 +:05)		
(STORE)	ST 0, 80	:02 in timer
Store new interval in timer		
(STORE)	ST 1, 84	To be 10:57
Store new interrupt time in 84		

Time of day is 10:55

Because priority interrupts are automatically delayed by a reference to location 80 when the high-resolution timer is installed, any priority program can read the real time without interference by using the following instruction sequence:

```
L   R, 80      Load interval into R
SL  R, 84      Subtract time of next interrupt
LCR R, R       Load complement of R
```

(R is any general register made available for use by the calling program.) Because the reference to location 80 precedes the reference to location 84, this sequence of instructions reads real time even in an interruptible environment. A priority program cannot interpose between the last two instructions in the former sequence nor between the first two instructions in the latter sequence.

Response Times

A pulse detected on an interruption-requested line sets an interruption-requested latch in the CPU. This latch remains set until:

The interrupt is accepted,

Or a system reset occurs,

Or a CHANGE PRIORITY MASK instruction is given to reset the latch and thus cancel any pending interruption request at that level.

The output of the 32 interruption-requested latches, gated by the 32 bits of the priority mask register and degated by the output of in-process latches for equal and higher priorities, is tested at the end of each instruction. If any priority interruption is requested and enabled, instruction sequencing stops.

As soon as the highest enabled priority level has been determined, its binary coded value is placed on the interrupt-accepted lines and the timing-out signal is raised. Simultaneously, the CPU begins fetching the new PSW and storing the old PSW.

If the timing-in signal rises within 1 microsecond at the CPU, the first instruction of the interrupt response routine is fetched 5 microseconds after the end of the current instruction. If the rise of the timing-in signal is delayed, so is the interruption setup; the CPU needs 3 microseconds after the rise of timing-in before it can fetch the next instruction.

Factors Affecting Response Times

Fast responsiveness is hindered when external signals are liable to be held pending while a supervisor performs housekeeping functions. In the following, however, it is assumed that the Model 44 is operated with a control program structured in the way previously described: i.e. with priority interrupts disabled only at the express wish of the applications programmer. Delay between an external signal and its programmed response will thus depend on:

1. *The priority level assigned to the signal.* Because an interruption request is held pending by, and a priority program can be interrupted by, a higher-level request, the system designer should consider the frequency of each type of external signal, and the duration of its programmed response — as well as its implied urgency — before assigning priority levels to signals.

2. *The priority mask register.* Note that disabling an interrupt level with the CHANGE PRIORITY MASK instruction is considered a function for the applications programmer and not for the system programmer.

3. *The duration of the current instruction.* Because Model 44 floating-point instructions and the four I/O instructions may take as long as approximately 125 microseconds to execute, this potential delay exists for all response routines.

4. *The extent to which the response requires access to common facilities that may be busy.*

NOTES

a. General registers and floating-point registers need never be busy in the logical sense, although the priority program may take time to preserve their prior contents.

b. Real time, the priority mask register, and the direct word interface need never be unavailable.

c. Common data sets, channels, I/O devices, the timer as a source of timed interrupts, and transient areas of main storage may be logically busy.

In each case, the access time to a common facility depends on the detailed structure of the control program routines that service these facilities.

Interface between CPU and External Devices

Interface Lines and Signal Timings

Interruption-Requested

These signals from the external device are permitted to rise at any time within the limits described in the following text and in Figure 11.

A signal of 0.5-microsecond minimum duration picks the associated latch; the latch will be reset at the time, during the setting up of the requested interruption, when timing-out and timing-in are both up.

The user must choose *one* of the following rules to determine the maximum duration of interruption-requested signals:

1. They must have a maximum duration of 1 microsecond.

2. They must fall before the rise of the associated timing-in.

A second signal on the line has one of the following results:

1. If it occurs before the rise of timing-in, it is ignored.

2. If it occurs after the fall of timing-out, it causes a second interruption (with the first being taken and the second held pending).

3. If it occurs in the interval between the rise of timing-in and the fall of timing-out (2.5 to 3 microseconds), it may or may not pick the latch; the occurrence of a repeat interruption is therefore indeterminate.

When the user is responding to the interrupt-accepted code by placing a byte on the description-in bus, he should not allow the causative interruption-requested line to rise a second time while timing-in and timing-out are both up (this pertains to the third condition just described).

Interrupt-Accepted, Timing-Out, Timing-In

The interrupt-accepted lines to the external device carry the binary coded value of the interrupt level selected for setup. Five lines are required to code for

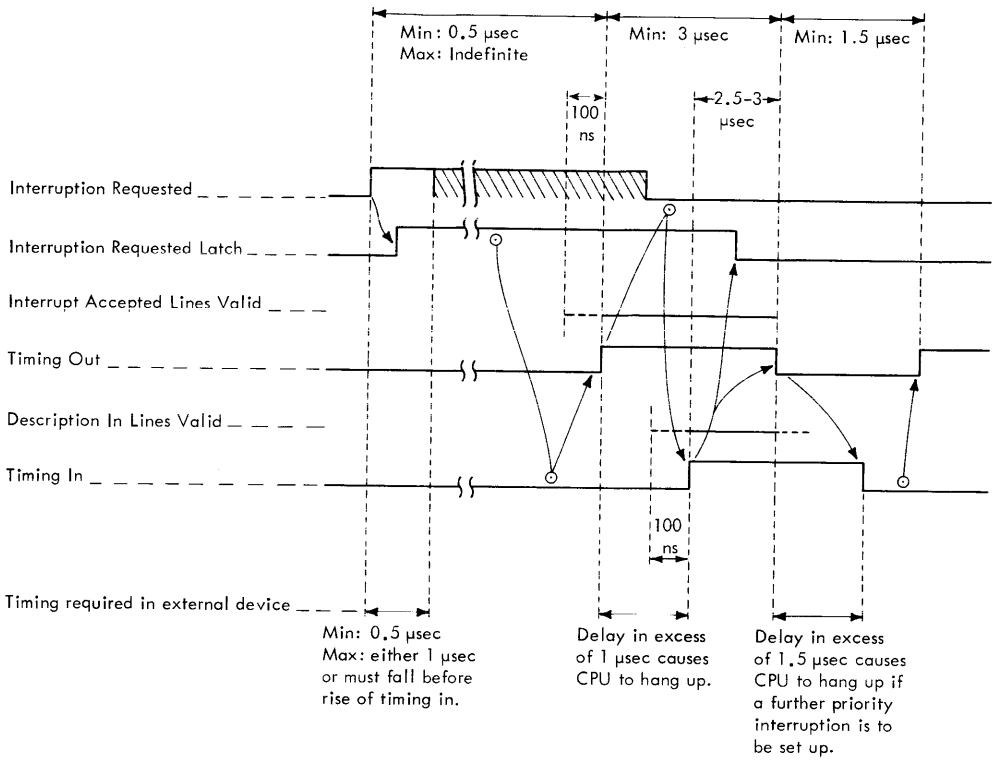


Figure 11. Priority Interrupt Signal Timing

32 levels. Signals on these lines are valid whenever timing-out is up.

The timing-out line to the external device rises after the interrupt-accepted lines are up and falls before they fall. The timing-out signal indicates that a priority interruption is about to be set up in the CPU and the interrupt-accepted lines are all valid. Timing-out rises at the output of the line driver not less than 100 nanoseconds after the selected interrupt level has been signalled on the interrupt-accepted lines; any further deskewing must be done at the external device.

The timing-out line can rise only when the timing-in line from the external device is down, and can fall only when timing-in is up. Timing-in can rise only when timing-out is up and can fall only when timing-out is down.

Timing-in must not rise until 100 nanoseconds after the description-in lines (if used) are valid, but should rise less than 1 microsecond after timing-out rises; any delay beyond 1 microsecond causes CPU processing to hang up during the setup for the interruption. Also, timing-in should fall within 1.5 microseconds after the fall of timing-out; any additional delay causes the CPU to hang up if a higher priority interruption is to be set up, because timing-out cannot rise for the new interruption request until timing-in has fallen for the old one.

The timing-out pulse falls 2.5 to 3 microseconds after the rise of timing-in.

Description-In

These lines from the external device can be used to provide additional information regarding an interruption request. They must be valid at the driver of the external device not less than 100 nanoseconds before the timing-in line is raised, and must remain valid until the fall of timing-out. Lines 0-7 of the description-in bus may be used to modify the instruction address as described in "Signal Exchanges" and "Interrupt Levels."

Test-Out, Test-Interlock

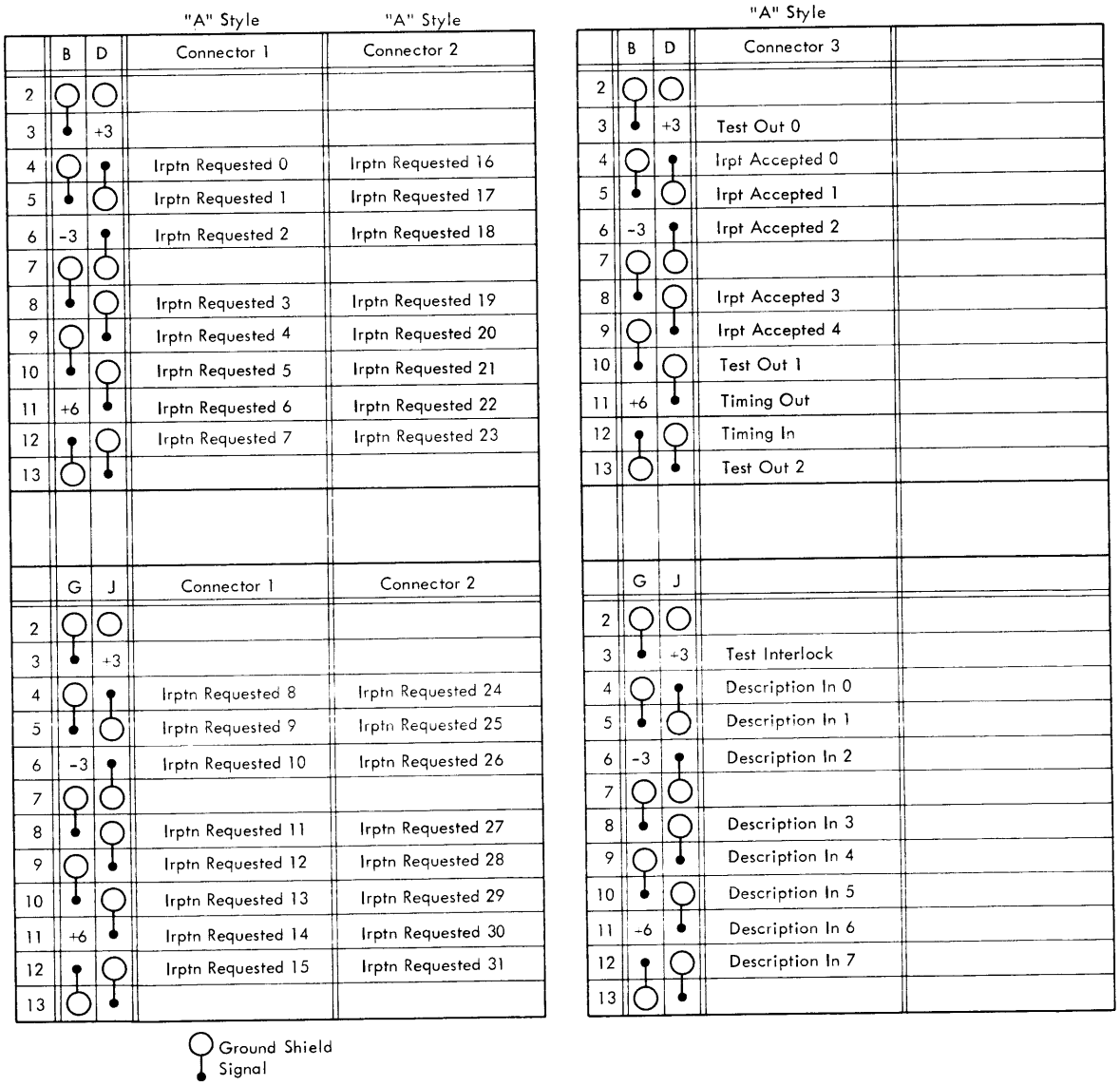
These lines are a maintenance testing facility for field engineering personnel. The three test-out lines are activated only when the test-interlock pin is grounded, whereupon they carry the inverse of the interruption-accepted lines 0, 1, and 2.

Connectors and Pin Assignments

The three cable connectors for the priority interrupt feature, located on the tailgate of the CPU, are as described for the direct word feature. The interface lines are assigned to specific connector pins as shown in Figure 12.

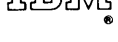
Electrical Specifications

The electrical requirements for the priority interrupt feature are as described for the direct word feature.



• Figure 12. Pin Assignments for Priority Interrupt Feature

Printed in U.S.A. A22-6900-1



International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, N.Y. 10601
[USA Only]

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
[International]