NEC Electronics Inc.

μPD71059
Interrupt Control Unit

Description

The μPD71059 is a low-power CMOS programmable interrupt control unit for microcomputer systems. It can process eight interrupt request inputs, allocating a priority level to each one. It transfers the interrupt with the highest priority to the CPU, along with interrupt address information. By cascading up to eight slave μPD71059s to a master μPD71059, a system can process up to 64 interrupt requests. System scale, interrupt routine address, interrupt request priority, and masking are all under complete program control.

Features

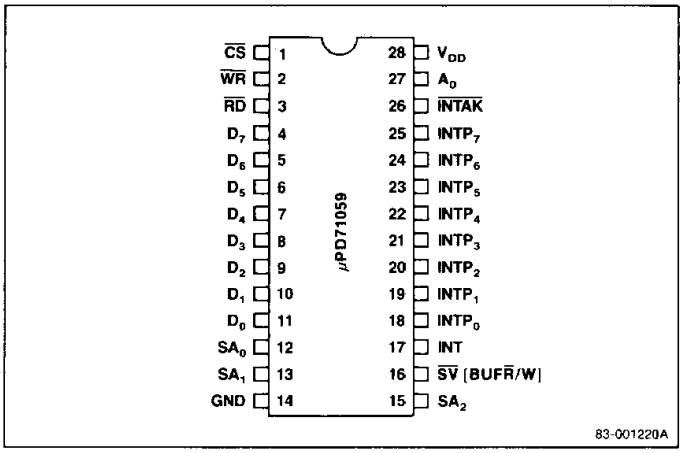
- ☐ μPD8085A compatible (CALL mode)
- ☐ μPD70108/70116 compatible (vector mode)
- ☐ Eight interrupt request inputs per chip
- ☐ Up to 64 interrupt request inputs per system (extended mode)
- ☐ Edge- or level-triggered interrupt request inputs
- ☐ Each interrupt maskable
- ☐ Programmable priority level
- ☐ Polling operation
- ☐ Single +5 V ±10% power supply
- ☐ Industrial temperature range: −40 to +85 °C
- ☐ CMOS technology
- ☐ 8 MHz and 10 MHz

Ordering Information

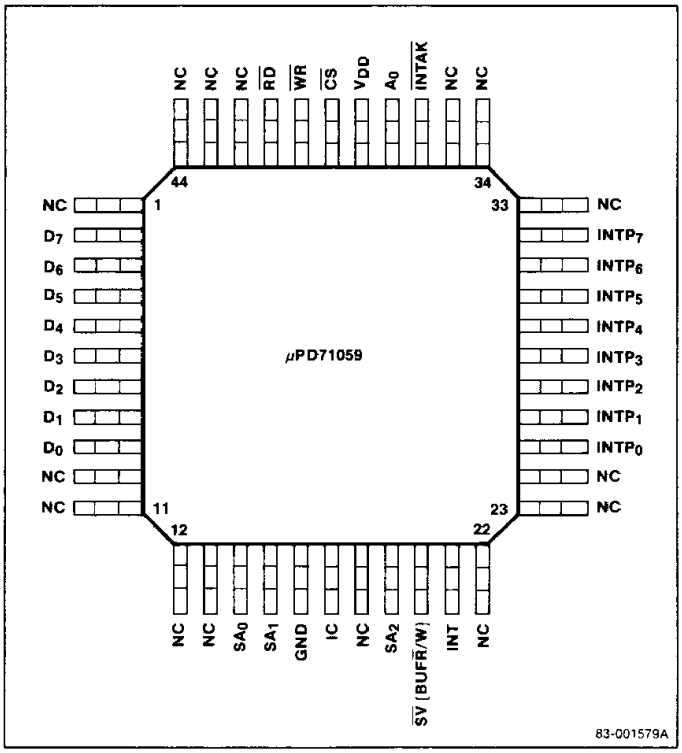
Part Number	Package
μPD71059C-8	28-pin plastic DIP (600 mil)
C-10	
G-8	44-pin plastic QFP (P44G-80-22)
GB-8	44-pin plastic QFP (P44GB-80-3B4)
GB-10	
L-8	28-pin PLCC
L-10	

Pin Configurations

28-Pin Plastic DIP



44-Pin Plastic QFP

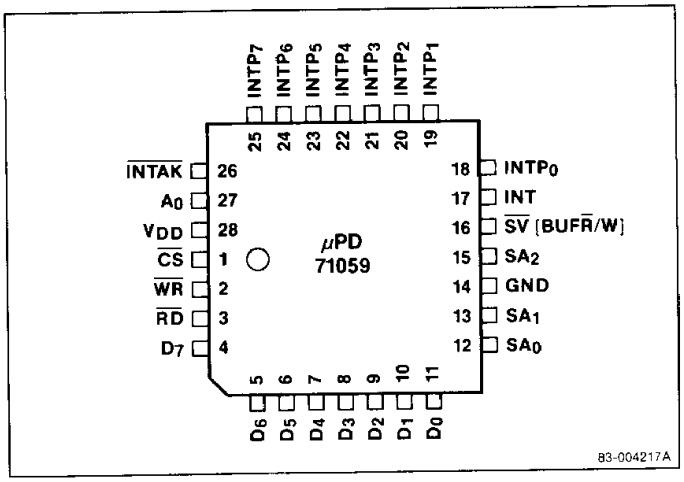


5f

50013-1 (NECEL-160)

Pin Configurations (cont)

28-Pin Plastic Leaded Chip Carrier (PLCC)



Pin Identification

Symbol	Function
D_7-D_0	Data bus I/O
SA_2-SA_0	Slave address I/O, bits 2, 1, 0
GND	Ground potential
IC	Internally connected
$\overline{SV}$ (BUF$\overline{R}$/W)	Slave (Buffer read write) I/O
INT	Interrupt output
$INTP_0$-$INTP_7$	Interrupt inputs
$\overline{INTAK}$	Interrupt acknowledge input
A_0	Address input
V_{DD}	Power supply
$\overline{CS}$	Chip select input
$\overline{WR}$	Write strobe input
$\overline{RD}$	Read strobe input
NC	Not connected

Pin Functions

D_7-D_0 [Data Bus]

The 8-bit 3-state bidirectional bus transfers data to and from the CPU through the system bus. The data bus becomes active when data is sent to the CPU in the $\overline{INTAK}$ sequence. Otherwise, the data bus is high impedance.

$\overline{CS}$ [Chip Select]

The CPU uses the μPD71059's $\overline{CS}$ input to select a μPD71059 to read from (IN instructions) or write to (OUT instructions). The $\overline{RD}$ and $\overline{WR}$ signals to the μPD71059 are enabled when $\overline{CS}$ is low. $\overline{CS}$ is not used for the INTAK sequence.

$\overline{RD}$ [Read Strobe]

The CPU sets the $\overline{RD}$ input to 0 when reading the internal registers IMR, IRR and ISR, and during polling operations to read polling data.

$\overline{WR}$ [Write Strobe]

The CPU sets the $\overline{WR}$ input to 0 when writing initializing words IW1-IW4 and command words IMW, PFCW and MCW.

A_0 [Address]

The A_0 input is used with $\overline{CS}$, $\overline{RD}$, and $\overline{WR}$ to read or write to the μPD71059. Normally, A_0 is connected to A_0 of the address bus. Table 1 shows the relationship between read/write operations and the control signals ($\overline{CS}$, $\overline{WR}$, $\overline{RD}$, and A_0).

$INTP_7$-$INTP_0$ [Interrupt Request from Peripheral]

$INTP_7$-$INTP_0$ are eight asynchronous interrupt request inputs. They can be set to be either edge-or level-triggered. These pins are pulled up by an internal resistance. Their power consumption is lower at high-level input than at low-level input.

INT [Interrupt]

INT is the interrupt request output from a μPD71059 to the CPU or master μPD71059. When an interrupt from a peripheral is input to an INTP pin and acknowledged, the μPD71059 asserts INT high to generate an interrupt request at the CPU or master μPD71059.

$\overline{INTAK}$ [Interrupt Acknowledge]

The $\overline{INTAK}$ input from the CPU acknowledges an interrupt from the μPD71059. After acknowledging the interrupt request, the CPU returns three low-level pulses (μPD8085) or two low-level pulses (μPD70108/70116). Synchronizing to these pulses, the μPD71059 sends a CALL instruction in three bytes, or an interrupt vector number in one byte through the data bus.

$\overline{SV}$ [BUF$\overline{R}$/W] [Slave, Buffer Read/Write]

This pin has two functions. When no external buffer is used in the data bus, it is the $\overline{SV}$ input. When $\overline{SV}$ is low, the μPD71059 acts as a slave. It operates as a master when $\overline{SV}$ is high. $\overline{SV}$ has no master/slave meaning when the μPD71059 is set to single mode.

As the BUF$\overline{R}$/W output, this pin can allow a bus transceiver to be controlled by the μPD71059, if one is required. When the μPD71059 changes its data bus to output, it sets BUF$\overline{R}$/W low. It sets BUF$\overline{R}$/W high when the data bus changes to input.

SA_2-SA_0 [Slave Address]

These pins are only used in systems with cascaded μPD71059s. The master μPD71059 uses these pins to address up to eight slave μPD71059s. These pins are output pins for masters, and input pins for slaves.

Note: In the single mode, SA_2-SA_0 are output pins, but the output data has no meaning.

V_{DD} [Power Supply]

This is the positive power supply.

GND [Ground]

This is the ground potential.

IC [Internally Connected]

This pin must be left unconnected.

Table 1. Read/Write Operations

$\overline{CS}$	$\overline{RD}$	$\overline{WR}$	A_0	Other Conditions	μPD71059 Operation	CPU Operation
0	0	1	0	IRR set by MCW	IRR to Data bus	IRR read
				ISR set by MCW	ISR to Data bus	ISR read
				Polling phase (Note 1)	Polling data to Data bus	Polling
0	0	1	1		IMR to Data bus	IMR read
0	1	0	0	$D_4 = 1$	Data bus to IW1 register	IW1 write
				D_4, $D_3 = 0$	Data bus to PFCW register	PFCW write
				$D_4 = 0$, $D_3 = 1$	Data bus to MCW register	MCW write
0	1	0	1	(Note 2)	Data bus to IW2 register	IW2 write
					Data bus to IW3 register	IW3 write
					Data bus to IW4 register	IW4 write
				After initializing	Data bus to IMR	IMW write
0	1	1	x		Data bus high impedance	
1	x	x	x			
0	0	0	x		Illegal	

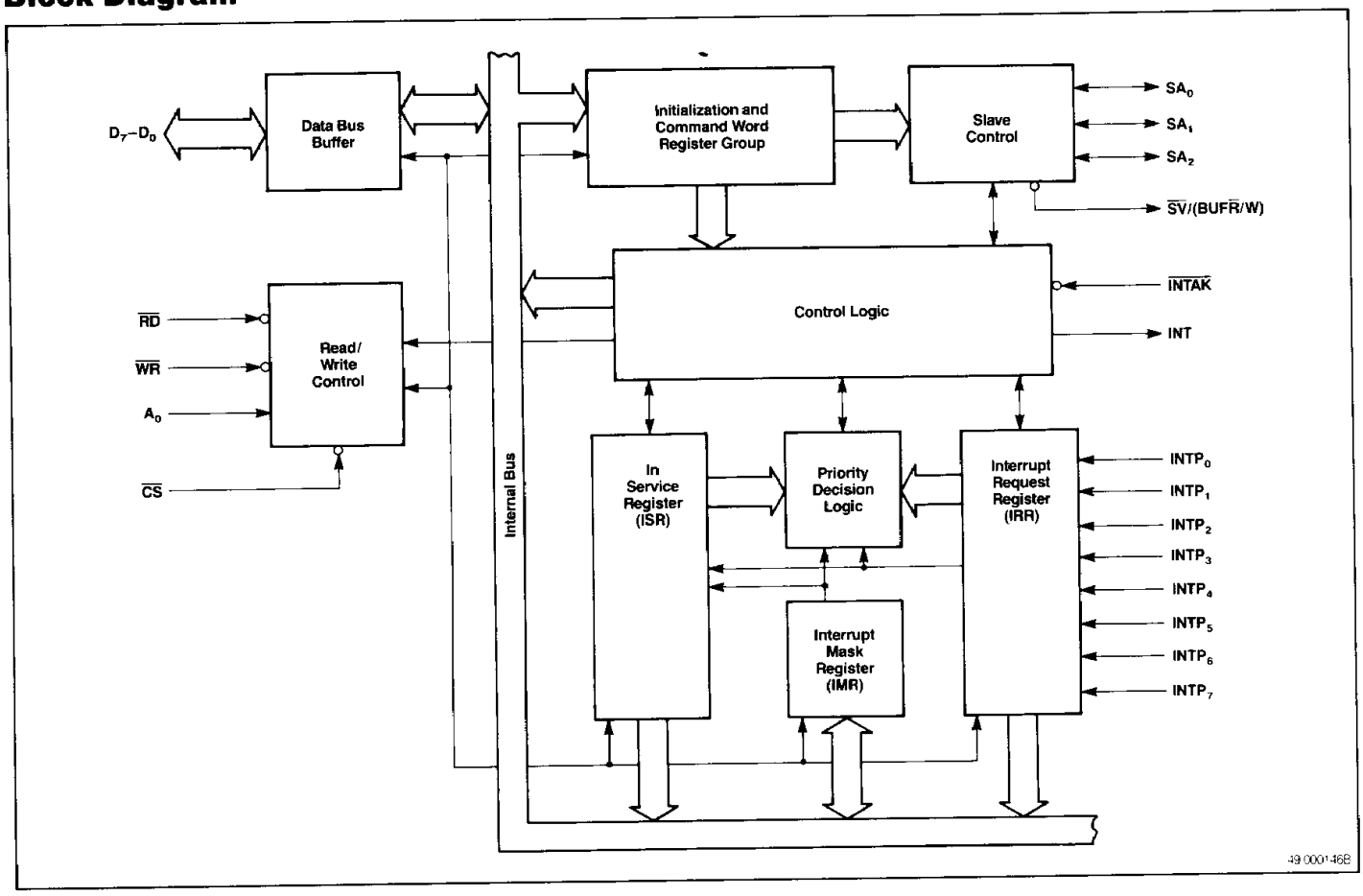
Note:

(1) In the polling phase, polling data is read instead of IRR and ISR.

(2) Refer to Control Words section for IW2-IW4 writing procedure.

5f

Block Diagram



Block Diagram Functions

Data Bus Buffer

The data bus buffer is a buffer between D_7-D_0 and the μPD71059's internal bus.

Read/Write Control

The read/write control controls the CPU's reading and writing to and from the μPD71059 registers.

Initialization and Command Word Registers

These registers store initializing words IW1-IW4 and command words PFCW (priority and finish control word) and MCW (mode control word). The CPU cannot read these registers.

Interrupt Mask Register [IMR]

The interrupt mask register stores the interrupt mask word (IMW) command word. Each bit masks an interrupt. If bit n of this register is 1, the interrupt request $INTP_n$ is masked and cannot be accepted by the μPD71059. The CPU can read this register by performing an IN instruction with $A_0 = 1$.

Interrupt Request Register [IRR]

The interrupt request register shows which interrupt levels are currently being requested. If bit n of the IRR is 1, $INTP_n$ is requesting an interrupt. The CPU can read this register.

In-Service Register [ISR]

The in-service register shows all interrupt levels currently in service. If bit n of this register is 1, the interrupt routine corresponding to $INTP_n$ is currently being executed. The CPU can read this register.

Slave Control

Slave control is used in systems with cascaded μPD71059s. A master μPD71059 uses it to control slave μPD71059s, and a slave uses it to interface with the master μPD71059.

Control Logic

The control logic receives and generates the signals that control the sequence of events in an interrupt.

Priority Decision Logic

The priority decision logic determines which interrupt request from the IRR will be serviced next. The decision is made based upon the current interrupt mask, interrupt service status, mode status, and current priority.

Absolute Maximum Ratings

$T_A = 25°C$

Power supply voltage, V_{DD}	−0.5 to +7.0 V
Input voltage, V_I	−0.5 to V_{DD} + 0.3 V
Output voltage, V_O	−0.5 to V_{DD} + 0.3 V
Power dissipation, PD_{MAX}	500 mW
Operating temperature, T_{opt}	−40 to +85°C
Storage temperature, T_{stg}	−65 to +150°C

Comment: Exposing the device to stresses above those listed in the absolute maximum ratings could cause permanent damage. The device is not meant to be operated under conditions outside the limits described in the operational section of this specification. Exposure to absolute maximum ratings for extended periods may affect device reliability.

Capacitance

$T_A = 25°C$; $V_{DD} = GND = 0$ V

		Limits				Test
Parameter	Symbol	Min	Typ	Max	Unit	Conditions
Input capacitance	C_I			10	pF	$f_c = 1$ MHz
I/O capacitance	C_{IO}			20	pF	Unmeasured pins returned to 0 V

DC Characteristics

$T_A = -40$ to $+85°C$; $V_{DD} = 5$ V ±10%

		Limits				
Parameter	Symbol	Min	Typ	Max	Unit	Test Conditions
Input voltage, high	V_{IH}	2.2		V_{DD} + 0.3	V	
Input voltage, low	V_{IL}	−0.5		0.8	V	
Output voltage, high	V_{OH}	0.7 x V_{DD}			V	$I_{OH} = -400$ μA
Output voltage, low	V_{OL}			0.4	V	$I_{OL} = 2.5$ mA
Input leakage current, high	I_{LIH}			10	μA	$V_I = V_{DD}$
Input leakage current, low	I_{LIL}			−10	μA	$V_I = 0$ V
Output leakage current, high	I_{LOH}			10	μA	$V_O = V_{DD}$
Output leakage current, low	I_{LOL}			−10	μA	$V_O = 0$ V
INTP input leakage current, high	I_{LIPH}			10	μA	$V_I = V_{DD}$
INTP input leakage current, low	I_{LIPL}			−300	μA	$V_I = 0$ V
Supply current (dynamic)						
μPD71059	I_{DD1}		3.5	9	mA	
μPD71059-10	I_{DD1}		4	9	mA	
Supply current (power down mode)	I_{DD2}		2	50	μA	Input pins: $V_{IH} = V_{DD} - 0.1$ V, $V_{IL} = 0.1$ V; Output pins: open (Note 1)

Notes:

(1) In power down mode, $INTP_0$-$INTP_7$, $\overline{INTAK}$, and $\overline{CS}$ must be at high level ($V_{IH} = V_{DD} - 0.1$ V).

5f

AC Characteristics

$T_A = -40$ to $+85°C$; $V_{DD} \pm 5\ V + 10\%$

Parameter	Symbol	8 MHz Limits Min	8 MHz Limits Max	10 MHz Limits Min	10 MHz Limits Max	Unit	Test Conditions
Read Timing							
A_0, $\overline{CS}$ setup to $\overline{RD}$ ↓	t_{SAR}	0		0		ns	
A_0, $\overline{CS}$ hold from $\overline{RD}$ ↑	t_{HRA}	0		0		ns	
$\overline{RD}$ pulse width low	t_{RRL}	160		120		ns	
$\overline{RD}$ pulse width high	t_{RRH}	120		90		ns	
Data delay from $\overline{RD}$ ↓	t_{DRD}		120		95	ns	$C_L = 150$ pF
Data float from $\overline{RD}$ ↑	t_{FRD}	10	85	10	60	ns	$C_L = 100$ pF
Data delay from A_0, $\overline{CS}$	t_{DAD}		200		120	ns	$C_L = 150$ pF
BUFR/$\overline{W}$ delay from $\overline{RD}$ ↓	t_{DRBL}		100		80	ns	
BUFR/$\overline{W}$ delay from $\overline{RD}$ ↑	t_{DRBH}		150		100	ns	
Write Timing							
A_0, $\overline{CS}$ setup to $\overline{WR}$ ↓	t_{SAW}	0		0		ns	
A_0, $\overline{CS}$ hold from $\overline{WR}$ ↑	t_{HWA}	0		0		ns	
$\overline{WR}$ pulse width low	t_{WWL}	120		100		ns	
$\overline{WR}$ pulse width high	t_{WWH}	120		90		ns	
Data setup from $\overline{WR}$ ↑	t_{SDW}	120		100		ns	
Data hold from $\overline{WR}$ ↑	t_{HWD}	0		0		ns	
Interrupt Timing							
INTP pulse width	t_{IPIPL}	100		80		ns	(Note 1)
SA setup to second, third $\overline{INTAK}$ ↓	t_{SSIA}	40		40		ns	Slave
$\overline{INTAK}$ pulse width low	t_{IAIAL}	160		120		ns	
$\overline{INTAK}$ pulse width high	t_{IAIAH}	120		90		ns	$\overline{INTAK}$ Sequence
INT delay from INTP ↑	t_{DIPI}		300		200	ns	$C_L = 150$ pF
SA delay from first $\overline{INTAK}$ ↓	t_{DIAS}		360		250	ns	Master, $C_L = 150$ pF
Data delay from $\overline{INTAK}$ ↓	t_{DIAD}		120		95	ns	$C_L = 150$ pF
Data float from $\overline{INTAK}$ ↑	t_{FIAD}	10	85	10	60	ns	
Data delay from SA	t_{DSD}		200		150	ns	Slave, $C_L = 150$ pF
BUFR/$\overline{W}$ delay from $\overline{INTAK}$ ↓	t_{DIABL}		100		80	ns	$C_L = 150$ pF
BUFR/$\overline{W}$ delay from $\overline{INTAK}$ ↑	t_{DIABH}		150		100	ns	
Other Timing							
Command recovery time	t_{RV1}	120		90		ns	(Note 2)
$\overline{INTAK}$ recovery time	t_{RV2}	250		90		ns	(Note 3)
$\overline{INTAK}$/command recovery time	t_{RV3}	250		90		ns	(Note 4)

Notes:

(1) The time to clear the input latch in edge-trigger mode.

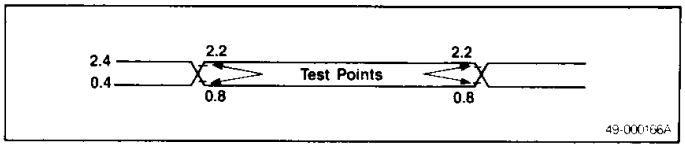
(2) The time to move from read to write operation.

(3) The time to move to the next $\overline{INTAK}$ operation.

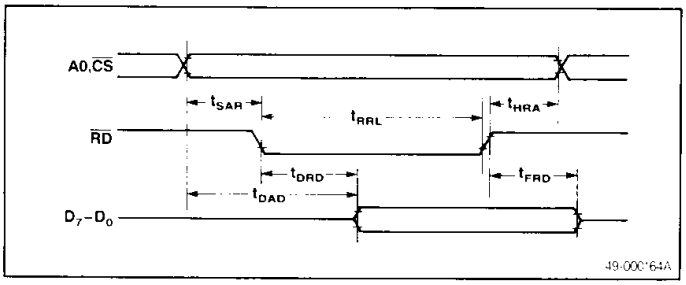
(4) The time to move $\overline{INTAK}$ to/from command (read/write).

Timing Waveforms

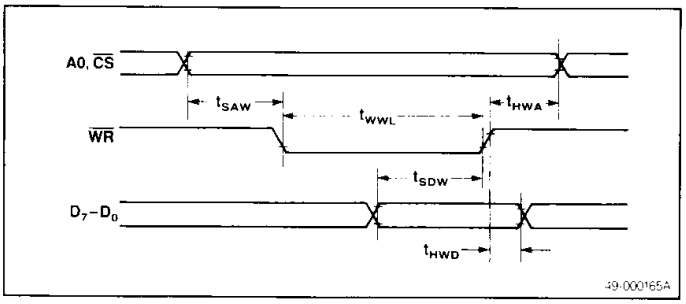
AC Test Input/Output Waveform



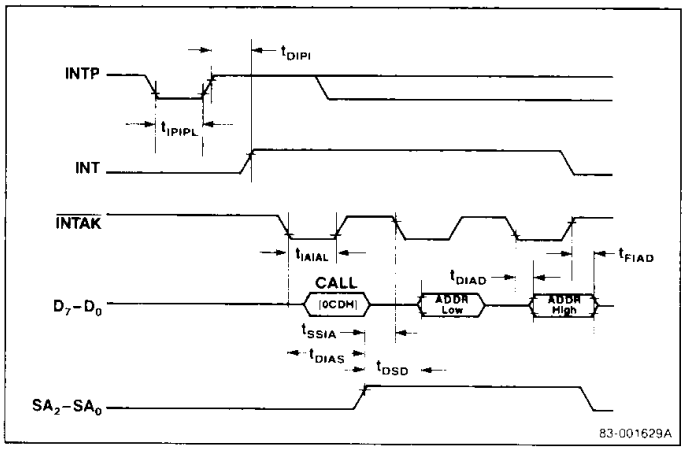
Read Cycle



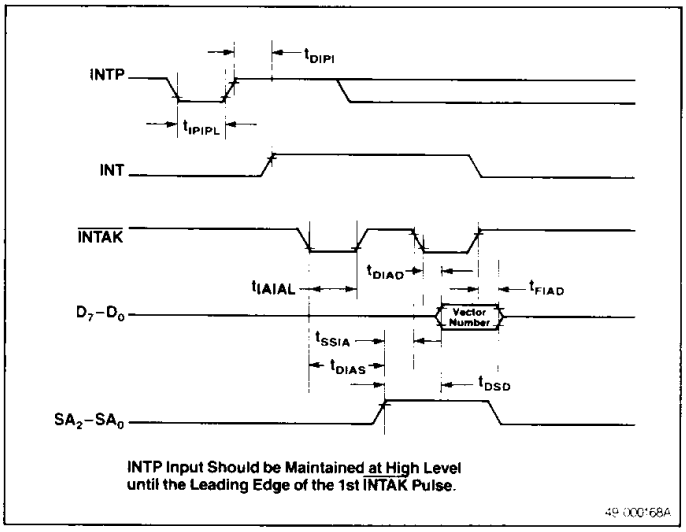
Write Cycle



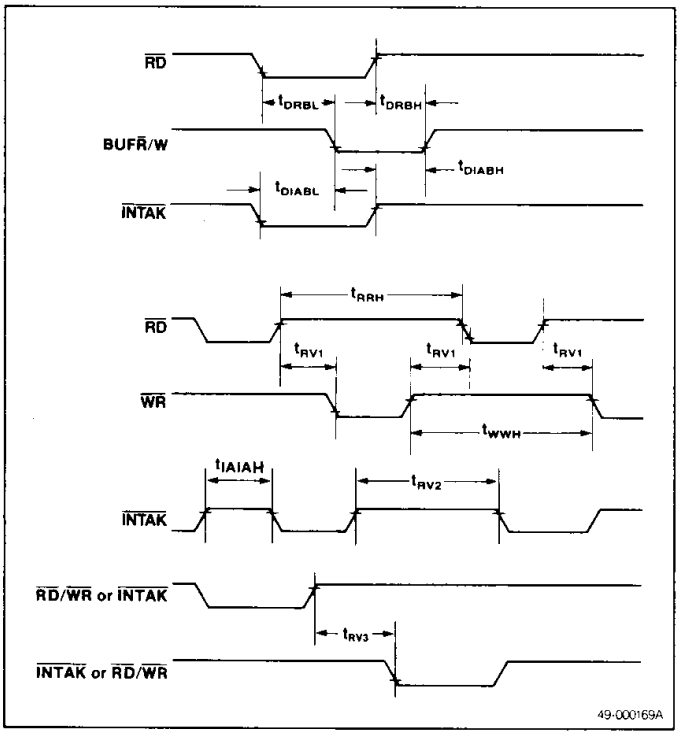
INTAK Sequence (CALL Mode)



INTAK Sequence (Vector Mode)



Other Timing



5f

Interrupt Operation

Almost all microcomputer systems use interrupts to reduce software overhead when controlling peripherals. However, the number of interrupt pins on a CPU is limited. When the number of interrupt lines increases beyond that limit, external circuits like the μPD71059 become necessary.

The μPD71059 can process eight interrupt request according to an allocated priority order and transmit the signal with the highest priority to the CPU. It also supplies the CPU with information to ascertain the interrupt routine start address. Cascading μPD71059s by connecting up to eight "slave" μPD71059s to a single "master" μPD71059 permits expansion to up to a maximum of 64 interrupt request signals.

Interrupt system scale (master/slave), interrupt routine addresses, interrupt request priority, and interrupt request masking are all programmable, and can be set by the CPU.

Normal interrupt operation for a single μPD71059 is as follows. First, the initialization registers are set with a sequence of initialization words. When the μPD71059 detects an interrupt request from a peripheral to an INTP pin it sets the corresponding bit of the interrupt request register (IRR). The interrupt is checked against the interrupt mask register (IMR) and the interrupt service register (ISR). If the interrupt is not masked and there is no other interrupt with a higher priority in service or requesting service, it generates an INT signal to the CPU.

The CPU acknowledges the interrupt by bringing the $\overline{\text{INTAK}}$ line low. The μPD71059 then outputs interrupt CALL or vector data onto the data bus in response to $\overline{\text{INTAK}}$ pulses. During the last $\overline{\text{INTAK}}$ pulse, the μPD71059 sets the corresponding bit in its ISR to indicate that this interrupt is in service and to disable interrupts with lower priority. It resets the bit in the IRR at this point. When the CPU has finished processing the interrupt, it will inform the μPD71059 by sending a finish interrupt (FI) command. This resets the bit in the ISR and allows the μPD71059 to accept interrupts with lower priorities. If the μPD71059 is in the self-FI mode, the ISR bit is reset automatically and this step is not necessary.

Software Features

The μPD71059 has the following software features:

- Interrupt types: CALL/vector
- Interrupt masking: Normal/extended nesting
- End of interrupt: Self-FI/normal FI/specific FI
- Priority rotation: Normal nested/extended nested/exceptional nested Automatic priority rotation Rotate to specific priority
- Polled mode
- CPU-readable registers

Hardware Configurations

The μPD71059 has the following hardware configurations:

- Interrupt input: Edge/level sensitive
- Cascading μPD71059s: Single/extended (master/slave)
- Output driver control: Buffered/non-buffered

Mode Control

These features and configurations are selected and controlled by the four initialization words (IW1-IW4) and the three command words (IMW, PFCW, and MCW). The format of these words are shown in figures 2 and 3, respectively.

Control Words

There are two types of μPD71059 control words: initialization words and command words.

There are four initialization words: IW1-IW4. These words must be written to the μPD71059 at least once to initialize it. They must be written in sequence.

There are three types of command words: interrupt mask word (IMW), priority and finish control word (PFCW), and the mode control word (MCW). These words can be written freely after initialization.

Initialization Words

Initialization sequence. When data is written to a μPD71059 after setting $A_0 = 0$ and $D_4 = 1$, data is always accepted as IW1. This results in a default initialization as shown below. See figure 1.

(1) The edge-trigger circuit of the INTP input is reset. IRR is cleared in the edge-trigger mode.

(2) ISR and IMR are cleared.

(3) $INTP_7$ receives the lowest priority; $INTP_0$ receives the highest.

(4) The exceptional nesting mode is released. IRR is set as the register to be read.

(5) Register IW4 is cleared. The normal nesting mode, non-buffer mode, FI command mode, and CALL mode are set.

Initialization Words. The initialization words are written consecutively, and in order. The first two, IW1 and IW2, set the interrupt address or vector. IW3 specifies which interrupts are slaves for master systems, and defines the slave number of a slave system. Therefore, IW3 is only required in extended systems. The μPD71059 will only expect it if bit D_1 of IW1, SNGL = 0. IW4 is only written if bit D_0 of IW1, I4 = 1. See figure 2 for the format of the initialization words.

Command Words

The command words give various commands to a μPD71059 during its operation to change interrupt masks and priorities, to end interrupt processing, etc. See figure 3.

IMW [Interrupt Mask Word]. This word masks the IRR and disables the corresponding INTP interrupt requests. It also masks the ISR in the exceptional nesting mode. Bits M_7-M_0 correspond to the interrupt levels of $INTP_7$-$INTP_0$, respectively.

In the exceptional nesting mode, interrupts corresponding to the bits of IRR and ISR are masked if the M_n bit is set to 1.

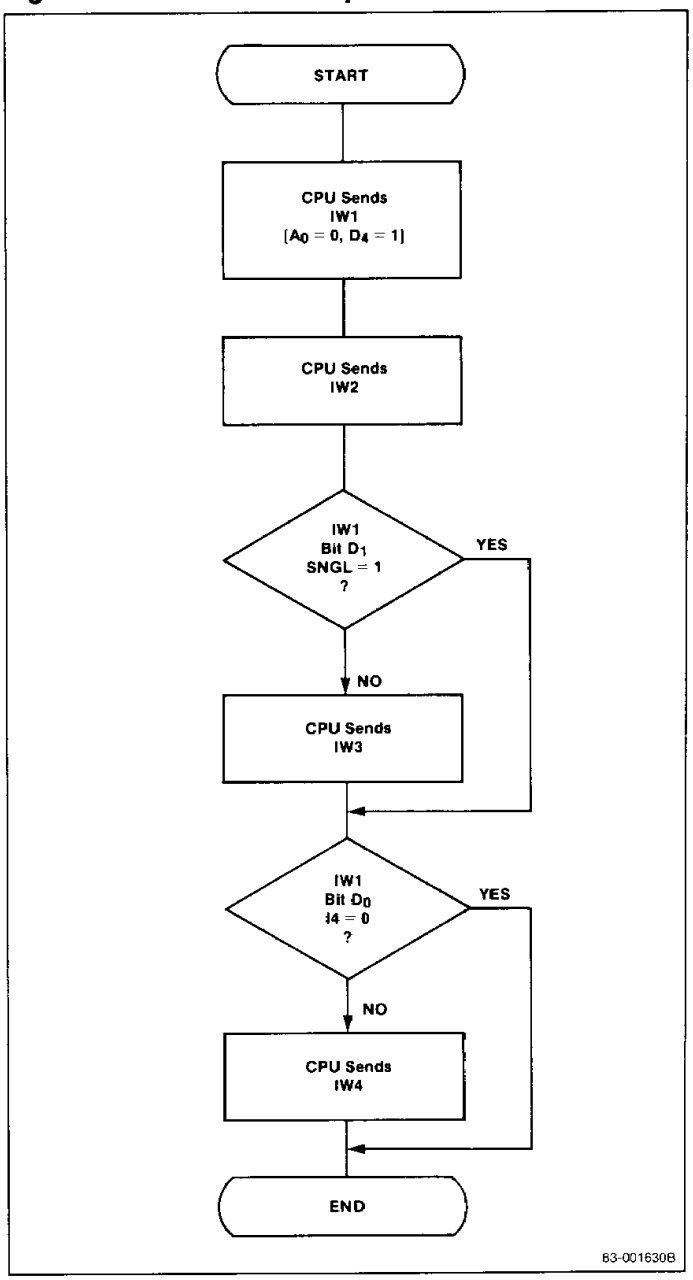
PFCW [Priority and Finish Control Word]. This word sets the FI (finish interrupt) command that defines the way that interrupts are ended, and the commands that change interrupt request priorities.

When RP (rotate priority) is set to 1, the priorities of the interrupt requests change (rotate). The priority order of the 8 INTP pins is as shown in figure 4. Setting a level as the lowest priority sets all the other levels correspondingly. For example, if $INTP_3$ is the lowest priority, $INTP_4$ will be the highest. ($INTP_7$ has the lowest priority after initialization).

SIL (specify interrupt level) is set to 1 to change the priority order or designate an interrupt level. It is used with the RP and FI bits (bits D_7 and D_5). When SIL = 1 and RP or FI = 1, the level identified by IL_2-IL_0 is designated as the lowest priority level. The other priorities will be set correspondingly. When used with FI = 1, it resets the ISR bit corresponding to the interrupt level IL_2-IL_0.

MCW [Mode Control Word]. This word is used to set the exceptional nesting mode, to poll the μPD71059, and to read the ISR and IRR registers.

Figure 1. Initialization Sequence



5f

Bits SR and IS/$\overline{IR}$ are used to read the contents of the IRR and ISR registers. When SR = 0, no operation is performed. To read IRR or ISR, set $A_0 = 0$ and select the IRR or ISR register by writing to MCW. To select the IRR register, write MCW with SR = 1 and IS/$\overline{IR}$ = 0. To select the ISR, write MCW with SR = 1 and IS/$\overline{IR}$ = 1. The selection is retained, and MCW does not have to be rewritten to read the same register again. IRR and ISR are not masked by the IMR.

Figure 2. Initialization Word Formats (Sheet 1 of 2)

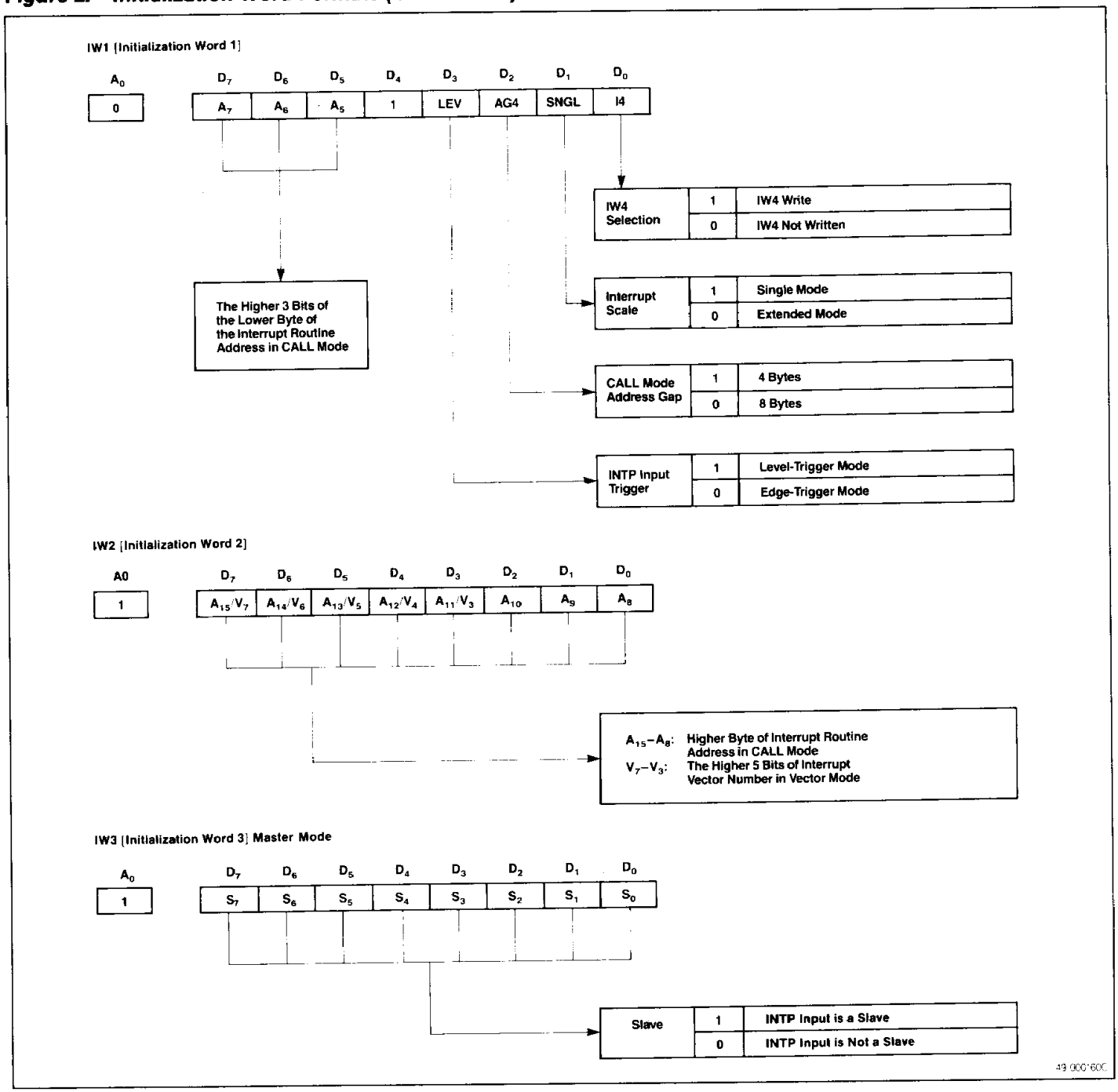
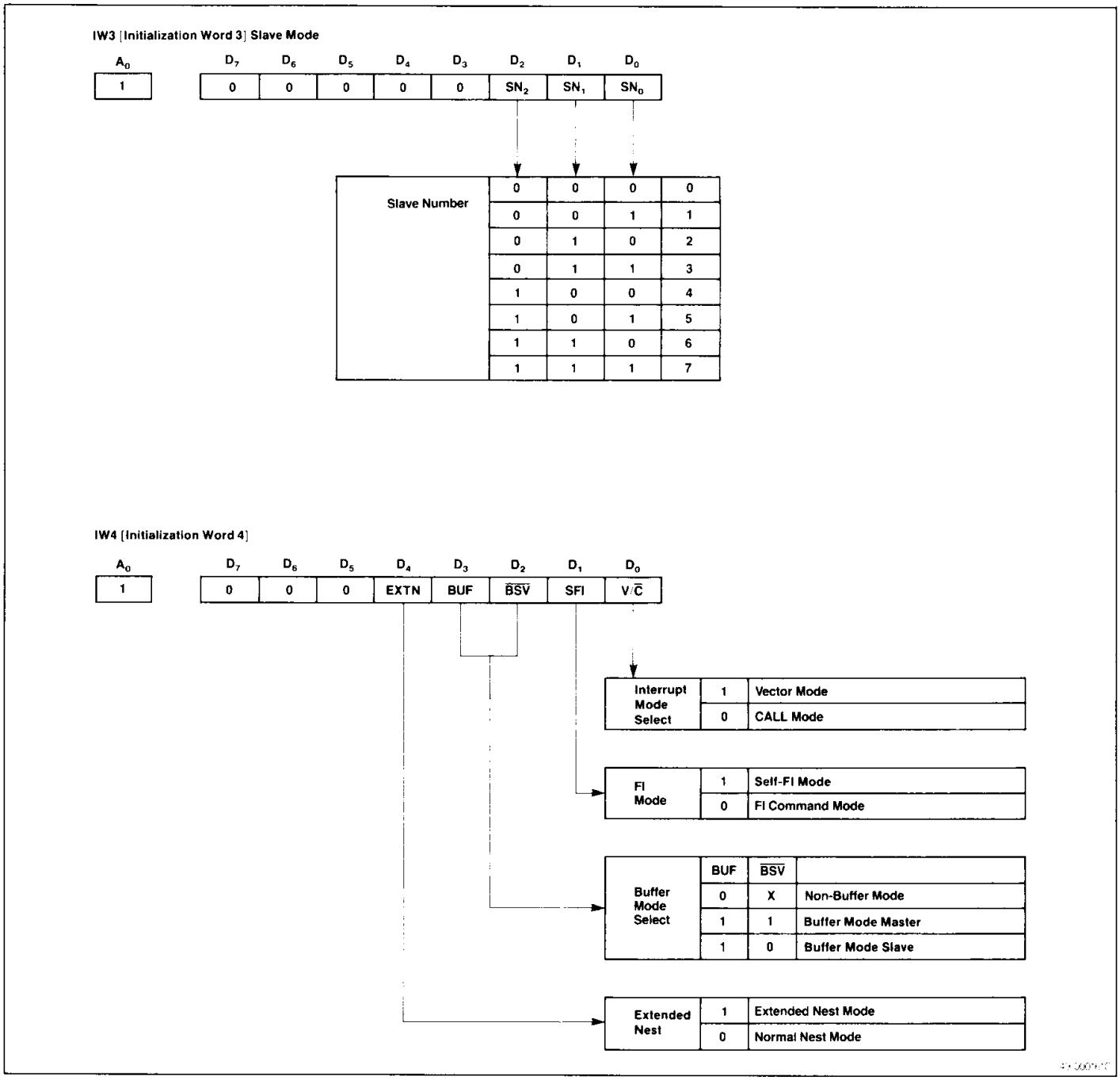


Figure 2. Initialization Word Formats (Sheet 2 of 2)



5f

Figure 3. Command Word Format

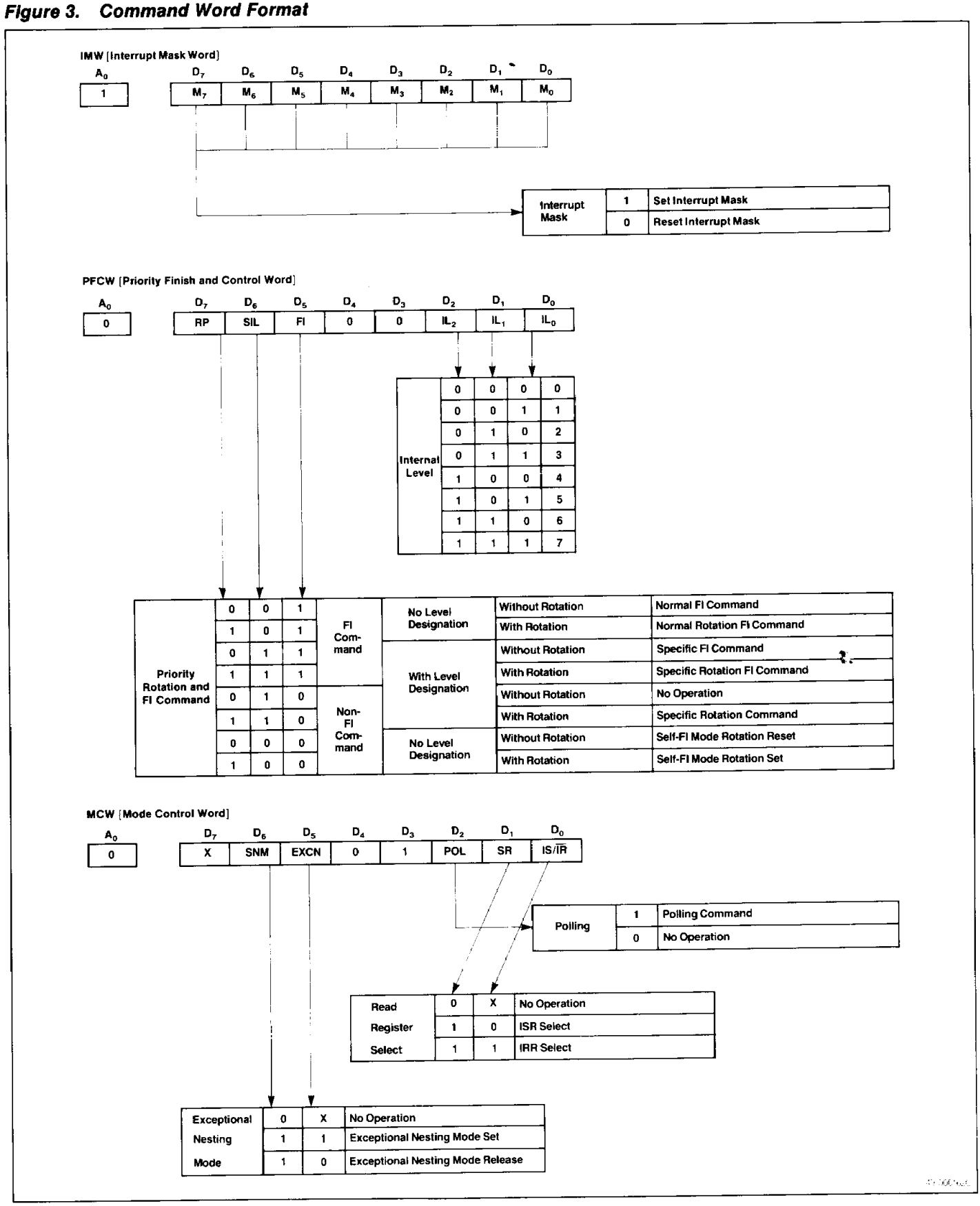
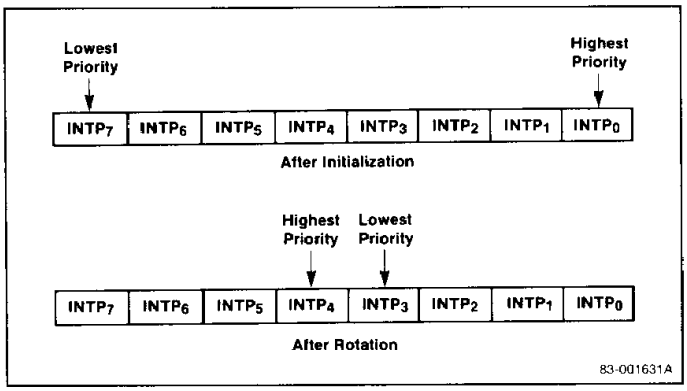


Figure 4. INTP Priority Order



CALL or Vector Modes

The μPD71059 passes interrupt routine address data to the CPU in two modes, depending on the CPU type. This mode is set by bit V/$\overline{C}$ in initialization word IW4. V/$\overline{C}$ is set to one to to select the vector mode for μPD70108/70116 CPUs, and reset to zero to select the CALL mode for μPD8085A CPUs.

CALL Mode [μPD8085A CPUs]

In this mode, when an interrupt is acknowledged by the CPU, the μPD71059 outputs three bytes of interrupt data to the data bus in its $\overline{INTAK}$ sequence. During the first $\overline{INTAK}$ pulse from the CPU, the μPD71059 outputs the CALL opcode 0CDH. During the next $\overline{INTAK}$ pulse, it outputs the lower byte of a two-byte interrupt routine address. During the third $\overline{INTAK}$ pulse, it outputs the upper byte of the address. The CPU interprets these three bytes as a CALL instruction and executes the CALL interrupt routine. See figure 5 and the $\overline{INTAK}$ sequence (CALL mode) μPD8085 diagram in the AC Timing Waveforms.

Interrupt routine addresses are set using words IW1 and IW2 during initialization. However, only the higher ten or eleven bits of the interrupt addresses are set, A_{15}-A_6 or A_{15}-A_5. The μPD71059 sets the remaining low bits (D_5-D_0 or D_4-D_0) to get the address of $INTP_n$'s interrupt routine. The addresses for $INTP_1$-$INTP_7$ are set in order of interrupt level. The space between interrupt addresses is determined by setting the AG4 bit (address gap 4 bytes) of IW1. When AG4 = 1, the interrupt routine starting addresses are 4 bytes apart. Therefore, the starting address for $INTP_n$ is the starting address for $INTP_0$ plus four times n. When AG4 = 0, starting addresses are eight bytes apart, so the starting address for $INTP_n$ is the starting address for $INTP_0$ plus eight times n. See figure 6.

Vector Mode [μPD70108/70116 CPUs]

In the vector mode, the μPD71059 outputs a one-byte interrupt vector number to the data bus in the $\overline{INTAK}$ sequence. The CPU uses that vector number to generate an interrupt routine address. See figure 7.

The higher five bits of the vector number, V_7-V_3, are set by IW2 during initialization. The μPD71059 sets the remaining three bits to the number of the interrupt input (0 for $INTP_0$, 1 for $INTP_1$, etc). See figure 8.

The CPU generates an interrupt vector by multiplying the vector number by four, and using the result as the address of a location in an interrupt vector table located at addresses 000H-3FFH. See figure 9.

System Scale Modes

The μPD71059 can operate in either single mode, with up to eight interrupt lines or extended mode, with more than one μPD71059 and more than eight interrupt lines. In extended mode a μPD71059 is in either master or slave mode.

Bit D_1, SNGL (single mode), of the first initialization word IW1 designates the scale of the interrupt system. SNGL = 1 designates that only one μPD71059 is being used (single mode system). SNGL = 0 designates an extended mode system with a master and slave μPD71059s. In the single mode (SNGL = 1), the SV input and IW4 buffer mode bits D_3 and D_2 do not indicate a master/slave relation for the μPD71059.

5f

Single Mode

This mode is the normal mode of μPD71059 operation. It has been described in the Interrupt Operation description. See figure 10 for a system example.

Extended Mode

In this mode, up to 64 interrupt requests can be processed using a master (μPD71059 in master mode) connected to a maximum of eight slaves (μPD71059s in slave mode). See figure 11 for a system example.

Figure 5. CALL Mode Interrupt Sequence

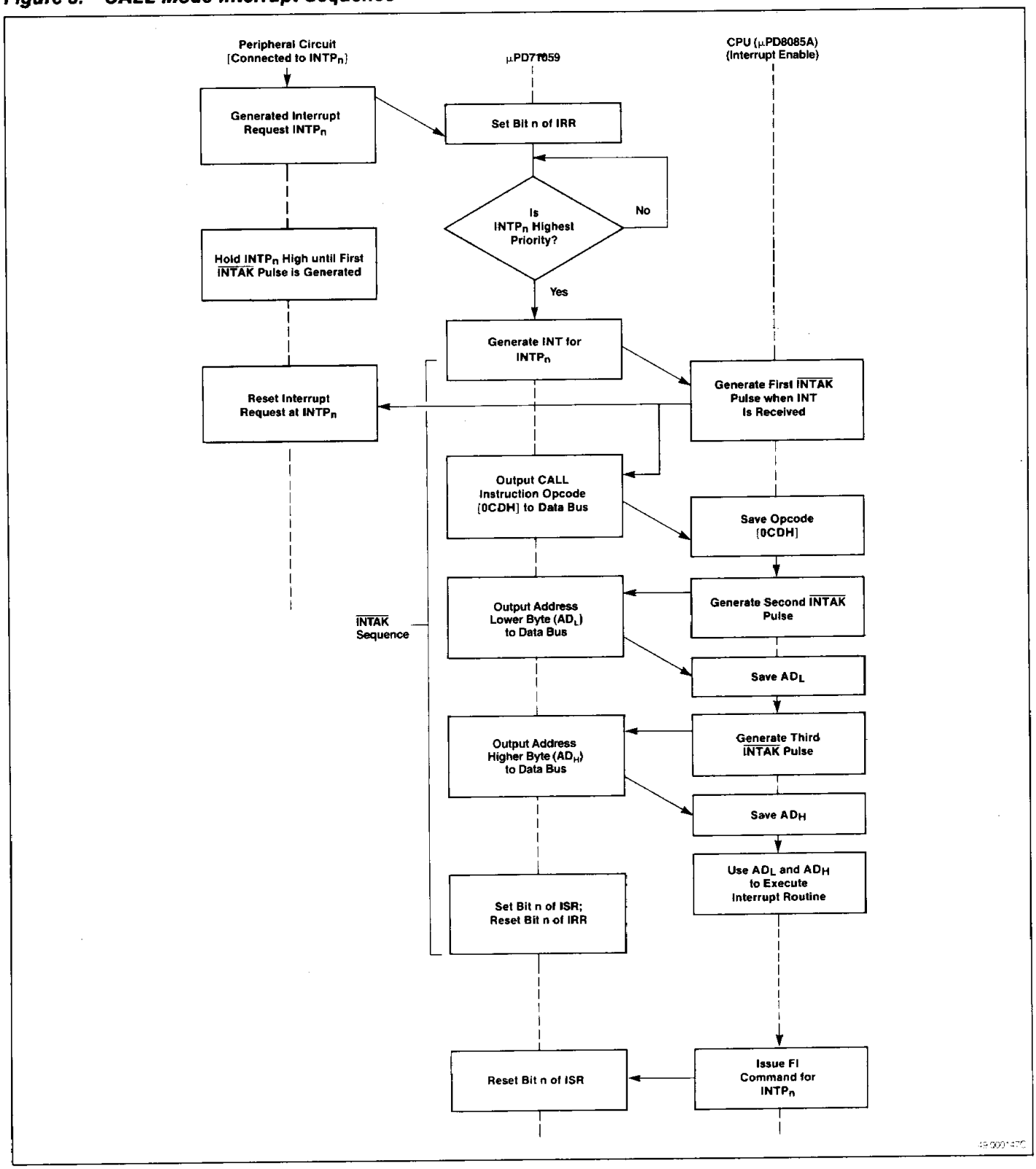


Figure 6. CALL Mode Interrupt Address Sequence

• Address Lower Byte [AD_L] During Second $\overline{INTAK}$

AG4 = 1 (4-Byte Spacing Address)

Interrupt Level	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
$INTP_0$	A_7	A_6	A_5	0	0	0	0	0
$INTP_1$	A_7	A_6	A_5	0	0	1	0	0
$INTP_2$	A_7	A_6	A_5	0	1	0	0	0
$INTP_3$	A_7	A_6	A_5	0	1	1	0	0
$INTP_4$	A_7	A_6	A_5	1	0	0	0	0
$INTP_5$	A_7	A_6	A_5	1	0	1	0	0
$INTP_6$	A_7	A_6	A_5	1	1	0	0	0
$INTP_7$	A_7	A_6	A_5	1	1	1	0	0

AG4 = 0 (8-Byte Spacing Address)

Interrupt Level	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
$INTP_0$	A_7	A_6	0	0	0	0	0	0
$INTP_1$	A_7	A_6	0	0	1	0	0	0
$INTP_2$	A_7	A_6	0	1	0	0	0	0
$INTP_3$	A_7	A_6	0	1	1	0	0	0
$INTP_4$	A_7	A_6	1	0	0	0	0	0
$INTP_5$	A_7	A_6	1	0	1	0	0	0
$INTP_6$	A_7	A_6	1	1	0	0	0	0
$INTP_7$	A_7	A_6	1	1	1	0	0	0

Note: When AG4 = 0, bit A_5 is ignored.

• Address Higher Byte [AD_H] During Third $\overline{INTAK}$

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8

83-001632A

5f

Figure 7. Vector Mode Interrupt Sequence

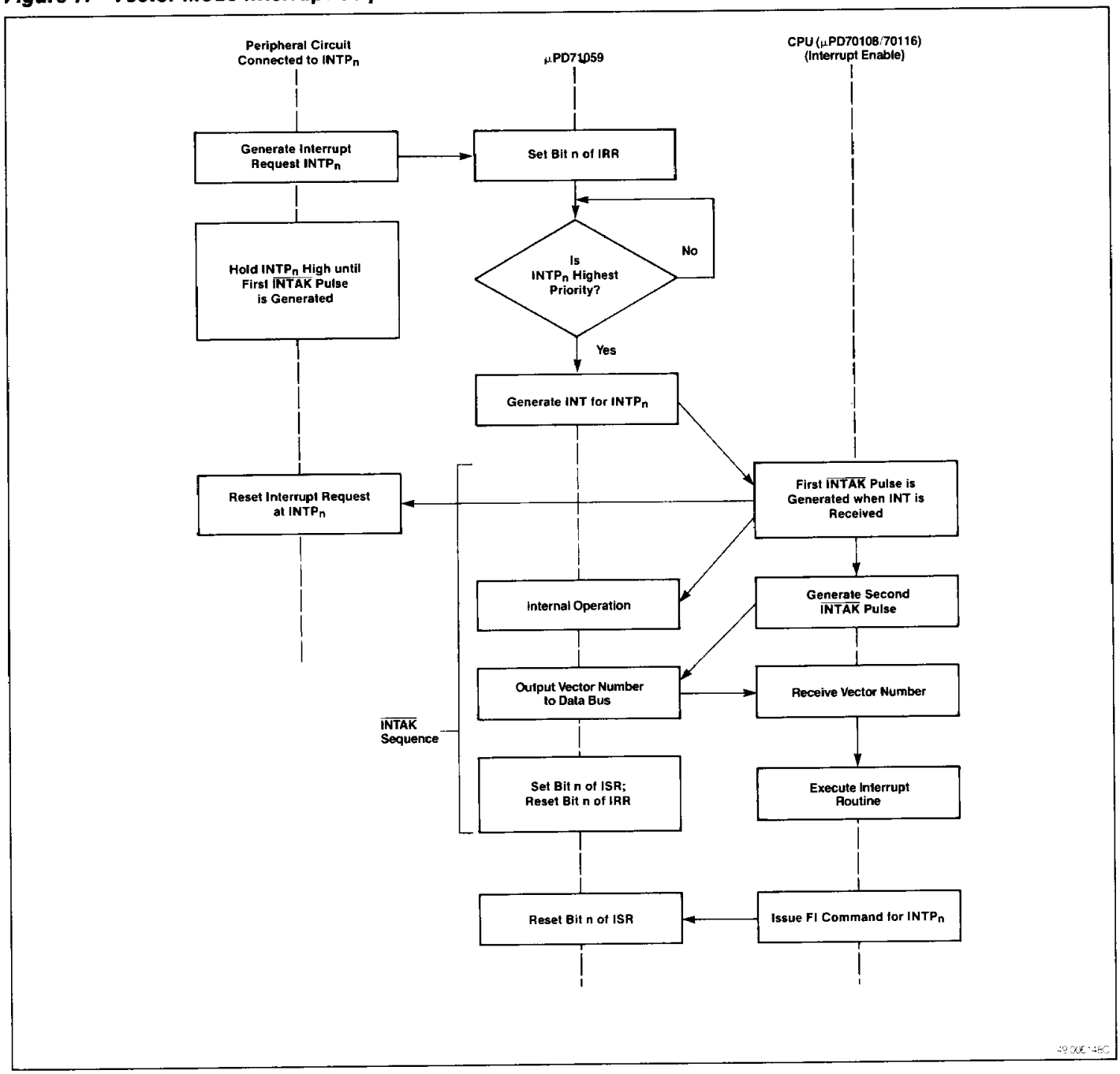


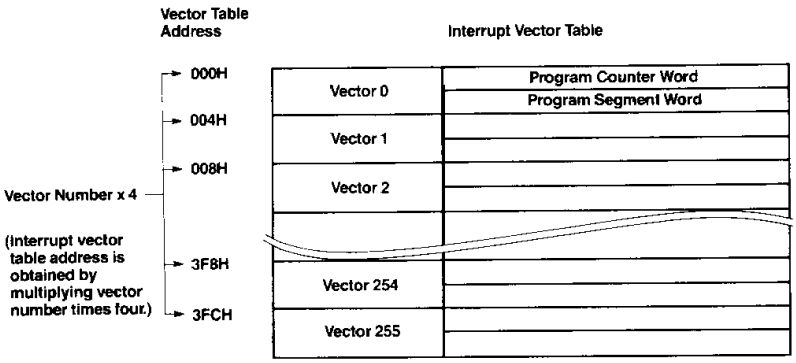
Figure 8. Vector Numbers Output in Vector Mode

Output During the Second $\overline{INTAK}$

Interrupt Levels	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
$INTP_0$	V_7	V_6	V_5	V_4	V_3	0	0	0
$INTP_1$	V_7	V_6	V_5	V_4	V_3	0	0	1
$INTP_2$	V_7	V_6	V_5	V_4	V_3	0	1	0
$INTP_3$	V_7	V_6	V_5	V_4	V_3	0	1	1
$INTP_4$	V_7	V_6	V_5	V_4	V_3	1	0	0
$INTP_5$	V_7	V_6	V_5	V_4	V_3	1	0	1
$INTP_6$	V_7	V_6	V_5	V_4	V_3	1	1	0
$INTP_7$	V_7	V_6	V_5	V_4	V_3	1	1	1

83-001633B

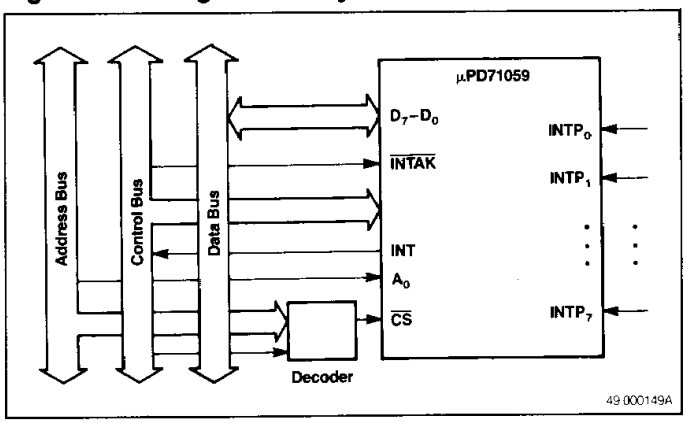
Figure 9. Interrupt Vectors for the μPD70108/70116



49-000152A

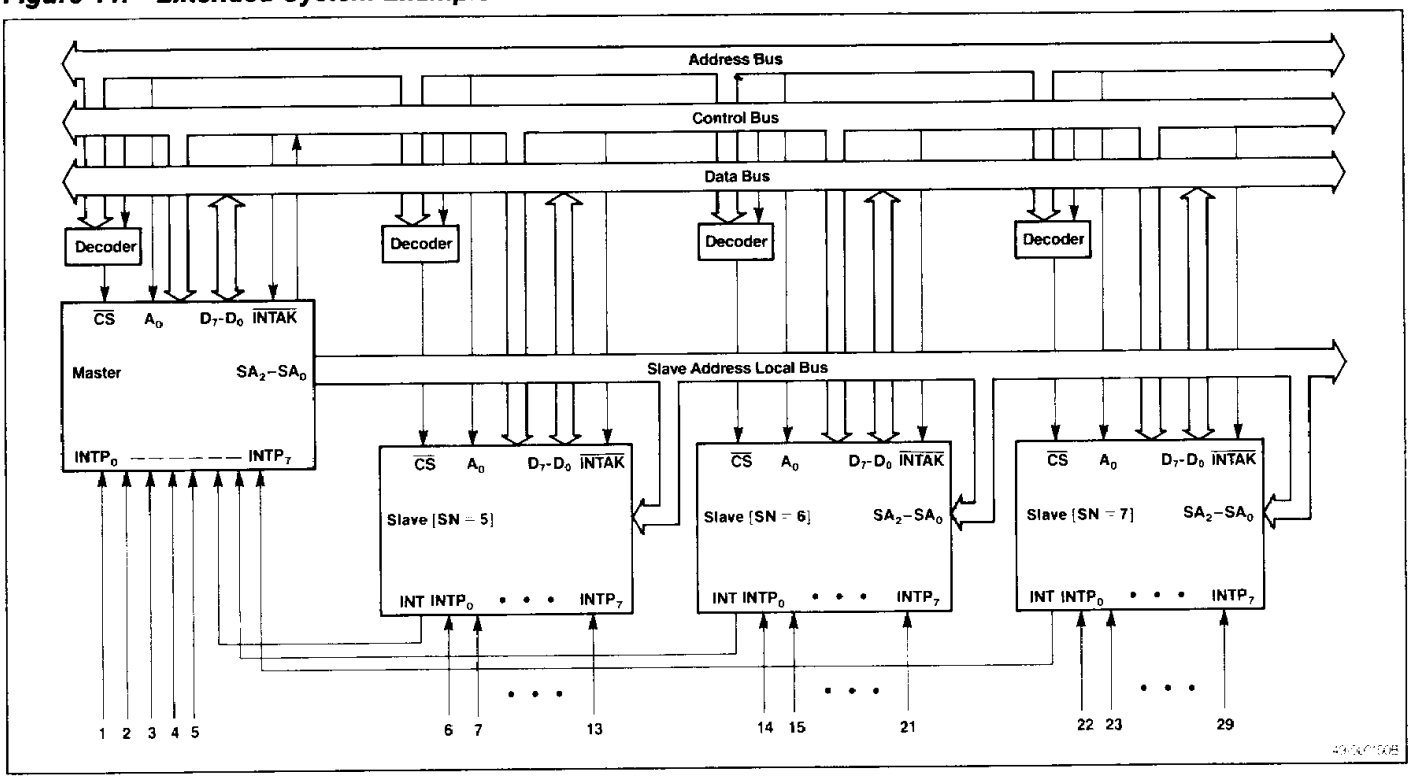
5f

Figure 10. Single Mode System



49-000149A

Figure 11. Extended System Example with Three Slaves



Master Mode

When a μPD71059 is a master in an extended mode system, S_7-S_0 of IW3 (master mode) define which of $INTP_7$-$INTP_0$ are inputs from slave μPD71059s or peripheral interrupts.

Consider an interrupt request from $INTP_n$. If $S_n = 0$, the interrupt is from a peripheral (for example, $INTP_0$ of the master μPD71059 in Figure 11), and the μPD71059 treats it the same way it would if it were in the single mode. SA_2-SA_0 outputs are low level and the master provides the interrupt address or vector number.

If $S_n = 1$, the interrupt is from a slave (for example, $INTP_7$ of the master). The master sends an interrupt to the CPU if the slave requesting the interrupt has priority. The master then outputs slave address n to pins SA_2-SA_0 on the first $\overline{INTAK}$ pulse by the CPU. It lets slave n perform the rest of the $\overline{INTAK}$ sequence.

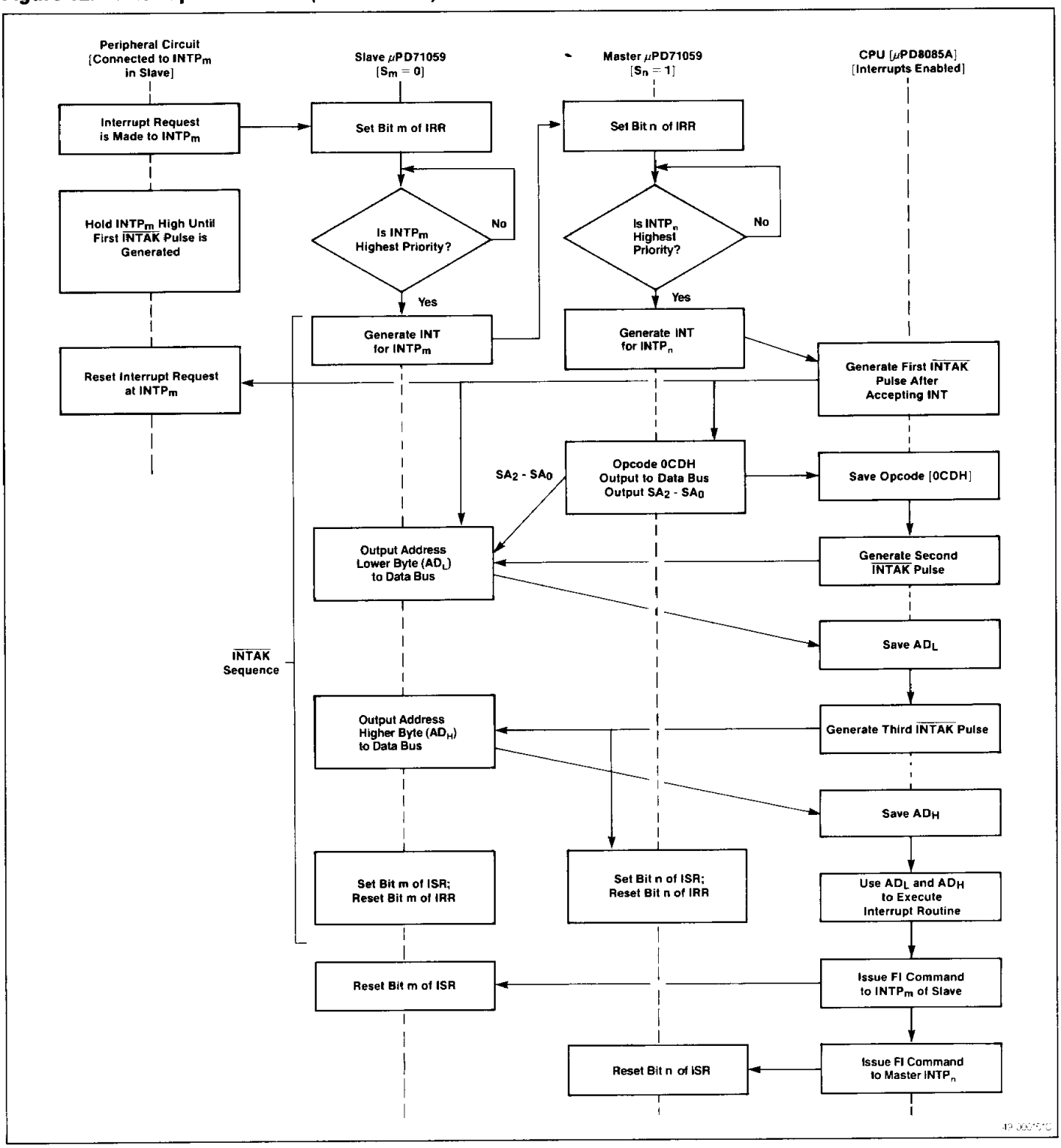
Slave Mode

When a slave receives an interrupt request from a peripheral, and the slave has no interrupts with higher priority in service, it sends an interrupt request to the master through its INT output. When the interrupt is accepted by the CPU through the master, the master outputs the slave's address on pins SA_2-SA_0. Each slave compares the address on SA_2-SA_0 to its own address. The slave that sent the interrupt will find a match. It completes the $\overline{INTAK}$ sequence the same way as a single μPD71059 would.

The master outputs slave address 0 when it is processing a non-slave interrupt. Therefore, do not use 0 as a slave address if there are less than eight slaves connected to the master.

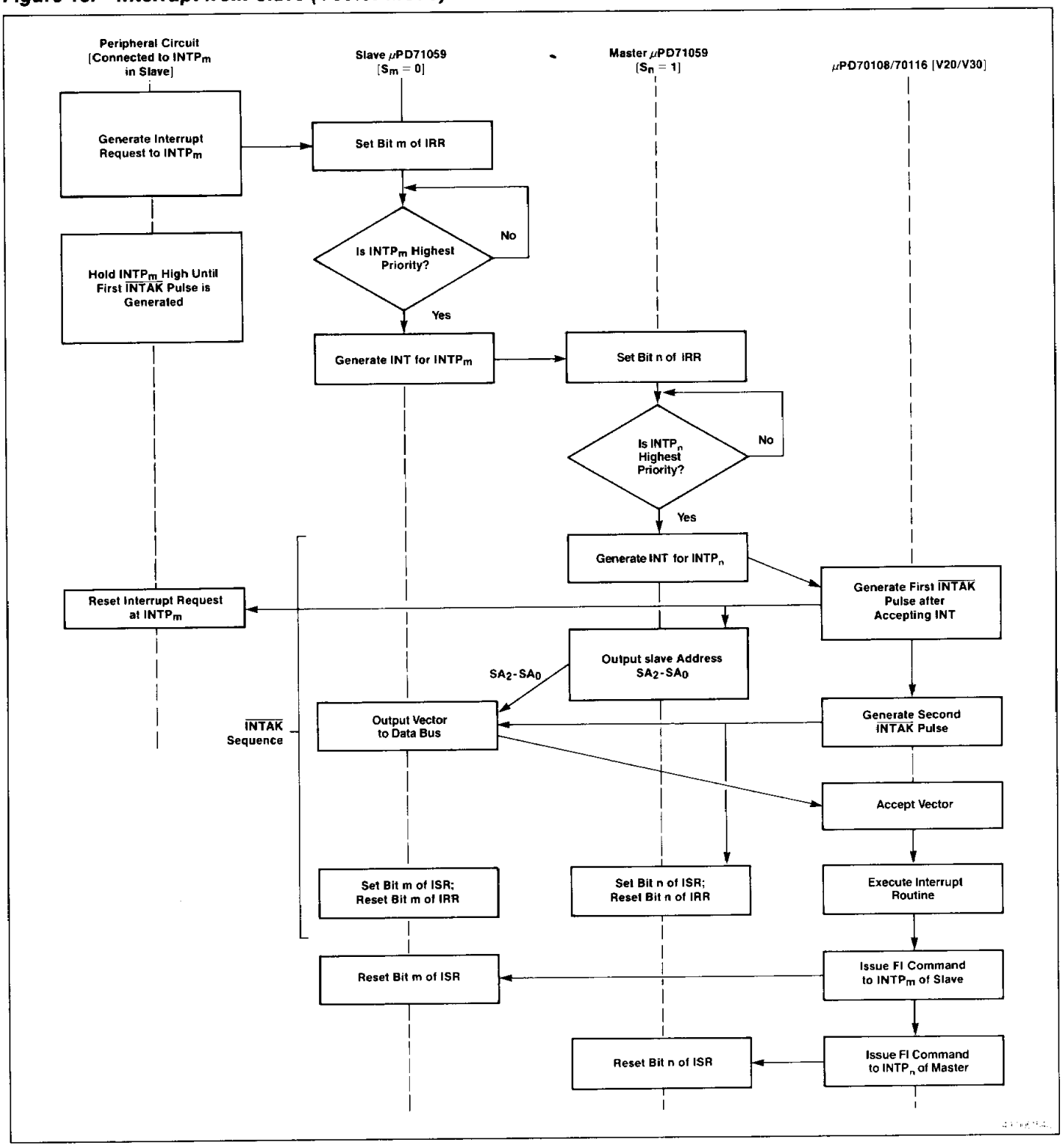
Figures 12 and 13 show the interrupt operating sequences for slaves in the extended mode.

Figure 12. Interrupt from Slave (CALL Mode)



5f

Figure 13. Interrupt from Slave (Vector Mode)



Buffer and Non-Buffer Modes

In a large system, a buffer may be needed by the μPD71059 to drive the data bus. A buffer mode is supplied, with a signal to specify the buffer direction. In the buffer mode, $\overline{SV}$ (BUF$\overline{R}$/W) is used to select the buffer direction and $\overline{SV}$ cannot be used to specify the master/slave mode. The master/slave selection must be set by IW4. IW4 bit D_3, BUF (buffer) and D_2, B$\overline{SV}$ (buffered slave) are used together to set the buffer mode and master/slave relation. When BUF = 0, the non-buffer mode is set and B$\overline{SV}$ has no meaning. When BUF = 1, the buffer mode is set. In buffer mode, the μPD71059 is a master when B$\overline{SV}$ = 1, a slave when BSV = 0. See figure 14.

Nesting Modes

The way a μPD71059 handles interrupts when there is already an interrupt in service depends on the nesting mode.

Normal Nesting Mode

This mode is set when IW4 is not written or when IW4 has EXTN = 0. It is the most common nesting mode. See figure 15.

When an interrupt is being executed in this mode (corresponding bit of ISR = 1), only interrupt requests with higher priority can be accepted.

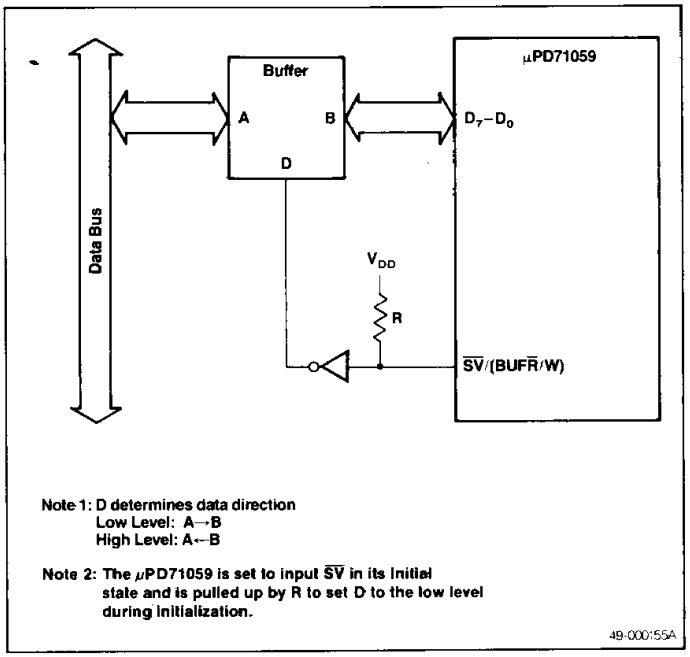
Extended Nesting Mode

This mode is only applicable to a master in the extended mode. A slave's eight interrupt priority levels become only one priority level when viewed by the master. Therefore, a request made by a slave with a higher priority than a previous request from the same slave will not be accepted. This cannot be called complete nesting since priority ranking within slaves loses its significance.

The extended nesting mode is set by setting bit D4 of IW4 in both the master and the slave. Interrupt requests of a higher level than the one currently being serviced can be accepted in the master from the same slave in the extended nesting mode.

Care should be exercised when issuing an FI (finish interrupt) command in the extended nesting mode. In an interrupt by a slave, the CPU first issues an FI command to the slave. Then, the CPU reads the slave's in-service register (ISR) to see if that slave still has interrupts in service. If there are no interrupts in service, (ISR = 00H) an FI command is issued to the master, as in the single mode when an interrupt is made by a peripheral.

Figure 14. Buffer Mode

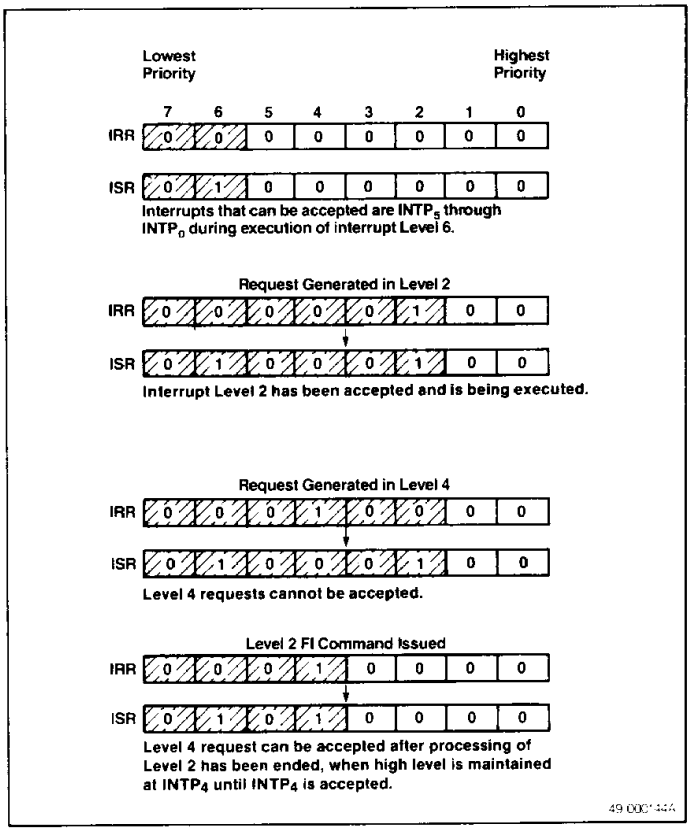


Note 1: D determines data direction
Low Level: A→B
High Level: A←B

Note 2: The μPD71059 is set to input $\overline{SV}$ in its initial state and is pulled up by R to set D to the low level during initialization.

49-000155A

Figure 15. Normal Nesting Mode



49-000134A

5f

Exceptional Nesting Mode

A μPD71059 in the normal or extended nesting mode cannot accept interrupts of a lower priority than the interrupts in service. Sometimes, however, it is desirable that requests with lower priority be accepted while higher-priority interrupts are being serviced. Setting the exceptional nesting mode allows this. After releasing the exceptional mode, the previous mode is resumed.

The exceptional nesting mode is controlled by the SNM (set nesting mode) and EXCN (exceptional nesting mode) bits (D_6 and D_5) of MCW. They set and release the exceptional nesting mode. The mode doesn't change when SNM = 0. Exceptional nesting is set if SNM and EXCN = 1 and released when SNM =1 and EXCN = 0.

Setting a bit in the IMW in the exceptional nesting mode, inhibits interrupts of that level and allows unmasked interrupts to all other levels, higher or lower priority.

The procedure for setting the exceptional nesting (EN) mode is as follows:

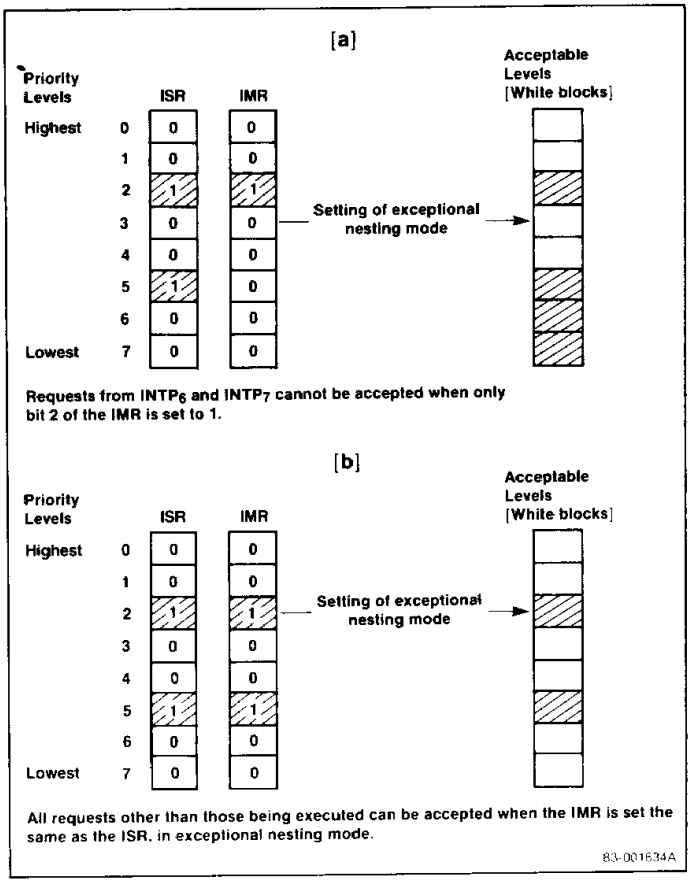
(1) Read the ISR.
(2) Write the ISR data to the IMR.
(3) Set the exceptional nesting mode.

In this way, all interrupt requests not currently in service will be enabled.

Figure 16 (a) shows what happens if IMR is not set to ISR. When the exceptional nesting is set, bit 2 of ISR will be ignored, and bit 5 will be serviced. Servicing bit 5 will mask the lower priority interrupts 6 and 7. When the ISR is set equal to the IMR as in (b), all interrupts except 2 and 5 can be serviced when the exceptional nesting mode is set.

Issuing an FI command to a level masked by the exceptional nesting mode requires caution. Since the ISR bit is masked, the normal FI command will not work. For this reason, a specific FI command specifying the ISR bit must be issued. After the exceptional mode is released, the normal FI command may be used.

Figure 16. Exceptional Nesting Mode



Finishing Interrupts (FI) and Changing the Priority Levels

The priority and finish control word (PFCW) issues FI commands and changes interrupt priorities.

Normal FI Command

	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
PFCW =	0	0	1	0	0	X	X	X

When a normal FI command is issued, the μPD71059 resets the ISR bit corresponding to the highest priority level selected from the interrupts in service. This operation assumes that the interrupt accepted last has ended.

When an interrupt routine changes the priority level or the exceptional nesting mode is set, this command will not operate correctly because the highest priority interrupt is not necessarily the last interrupt in service.

Specific FI Command

	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
PFCW =	0	1	1	0	0	IL_2	IL_1	IL_0

When the specific FI command is issued, the μPD71059 resets the ISR bit designated by bits IL_2-IL_0 of the PFCW. This command is used when the normal nesting mode isn't being used.

Self-FI Mode

When SFI of IW4 = 1, the μPD71059 is set to the self-FI mode. In this mode, the ISR bit corresponding to the interrupt is set and reset during the third $\overline{INTAK}$ pulse. Therefore, the CPU does not have to issue an FI command when the interrupt routine ends. In this mode, however, the ISR does not store the routine in service. Unless interrupts are disabled by the interrupt routine, newly generated interrupt requests are generated without priority limitation by the ISR. This can cause a stack overflow when frequent interrupt requests occur, or when the interrupt is level triggered.

Self-FI Rotation

Rotation of interrupt priorities can be added to the self-FI mode. In this case, the corresponding interrupt is set to the lowest priority level when a bit is reset in the ISR at the end of the $\overline{INTAK}$ sequence.

Self-FI Rotation Set:

	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
PFCW =	1	0	0	0	0	X	X	X

Self-FI Rotation Reset:

	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
PFCW =	0	0	0	0	0	X	X	X

Normal Rotation FI Command

	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
PFCW =	1	0	1	0	0	X	X	X

When the normal rotation FI command is issued, the μPD71059 resets the ISR bit corresponding to the highest priority level selected from the interrupts in service, then rotates the priority levels so that the interrupt just completed has the lowest priority.

Specific Rotation FI Command

	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
PFCW =	1	1	1	0	0	IL_2	IL_1	IL_0

When the specific rotation FI command is issued, the μPD71059 resets the ISR bit designated by bits IL_2-IL_0 of the PFCW and rotates the interrupt priorities so that the interrupt just reset becomes the lowest priority. This change in priority levels is different from the normal nesting mode, therefore, it is the user's responsibility to manage nesting.

Specific Rotation Command

	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
PFCW =	1	1	0	0	0	IL_2	IL_1	IL_0

When the specific rotation command is issued, the μPD71059 sets the interrupt priority specified by IL_2-IL_0 to the lowest priority. In this case also, the user must manage nesting.

5f

Triggering Mode

Bit D_3 of the first initialization word, IW1, is LEV (level-trigger mode bit). LEV sets the trigger mode of the INTP inputs. The level-trigger mode is set when LEV = 1. The rising-edge-triggered mode is set when LEV = 0.

Edge-Trigger Mode

In the edge-trigger mode, an interrupt is detected by the rising edge of the signal on an INTP input. Although an IRR bit goes high when INTP is high, the IRR bit is not latched until the CPU returns an $\overline{INTAK}$ pulse. Therefore, the INTP input should be maintained high until $\overline{INTAK}$ is received. This filters out noise spikes on the INT lines. To send the next interrupt request, temporarily lower the INTP input, then raise it.

Level-Trigger Mode

In the level-trigger mode, an IRR bit is set by the INTP input being at a high level. As in the edge-trigger mode, the INTP must be maintained high until the $\overline{INTAK}$ is received. Interrupts are requested as long as the INTP input remains high. Care should be taken so as not to cause a stack overflow in the CPU. See figure 17.

Note: The μPD71059 operates as if the $INTP_7$ interrupt had occurred if the $\overline{INTAK}$ pulse is sent to the μPD71059 by the CPU when the μPD71059 INT output level is low. Bit 7 of ISR is not set. Accordingly, if it is expected that this will occur, the $INTP_7$ interrupt should be reserved for servicing incomplete interrupts. The FI should not be issued for incomplete interrupts. See figure 18.

Figure 17. INTP Input

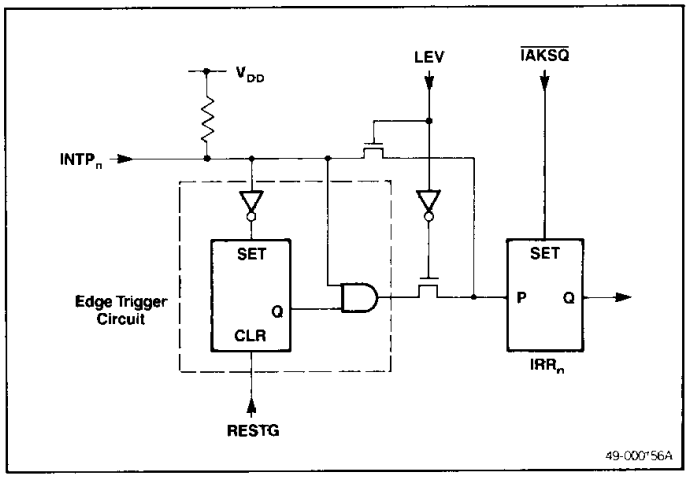
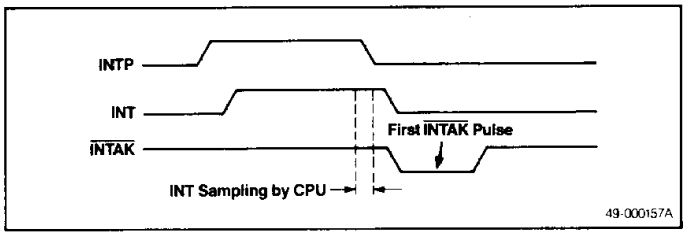


Figure 18. Incomplete Interrupt Request



Polling Operation

When polling, the CPU should disable its INT input. Next, it issues a polling command to the μPD71059 using MCW with POL = 1. This command sets the μPD71059 in polling mode until the CPU reads one of the μPD71059's registers.

When the CPU performs a read operation with $A_0 = 0$ in the polling mode, polling data as shown in figure 19 is read instead of ISR or IRR. The μPD71059 then ends the polling mode.

Figure 19. Polling Data

	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
MCW =	INT	0	0	0	0	PL_2	PL_1	PL_0

83-001635A

The INT bit has the same meaning as the INT pin. When it is set to 1, it means that the μPD71059 has accepted an INTP input.

The PL_2-PL_0 (permitted level) bits show which INTP input requested an interrupt when INT = 1.

If INT in the polling data is 1, the μPD71059 sets the ISR bit corresponding to the interrupt level shown by bits PL_2-PL_0 of the polling data and considers that interrupt as being executed. The CPU then processes the interrupt accordingly, based on the polling data read. An FI command should be issued when this processing ends.

Note: When a read is performed with A0 = 1 after the polling command is sent to the μPD71059, the IMR will be read instead of polling data. However, when the polling command is sent, the μPD71059 operates in the same manner when $A_0 = 0$ as it does when $A_0 = 1$. This means that although A_0 was set to 1, the μPD71059 will send the contents of the IMR, but it will also set an ISR bit just as it would if A0 had been set to zero. This may disturb the nesting. Therefore, performing a read operation with $A_0 = 1$ immediately after sending the polling command should be avoided.