NEC
NEC Electronics Inc.

µPD70327 (V25 Software Guard) 16-Bit Microcomputer: Software-Secure, Single-Chip, CMOS

Description

The µPD70327 (V25 Software Guard) is a high-performance, 16-bit, single-chip microcomputer with an 8-bit external data bus. The µPD70327 is fully software compatible with the µPD70108/116 (V20®/30®) as well as the µPD70320/330 (V25™/35™).

The µPD70327 allows external executable code to be encrypted by a user-defined translation table. The µPD70327 will automatically decode the encrypted op-codes internally before the instructions are moved into the instruction execution register. As a result, the µPD70327 offers identical performance to the standard V25 even during security mode operation. The security feature may be selected by hardware and/or software, and may be switched from one state to the other under software control.

The µPD70327 has the same complement of internal peripherals as the standard V25 and maintains compatibility with existing drivers. Other than the additional mode select pin, the µPD70327 also maintains pin compatibility with other members of the standard V25 family.

Note: The electrical specifications of the V25 Software Guard and the standard V25 are the same. The instruction sets are also the same except for (BRKS and BRKN added to control the Security and Normal operational modes. For electrical specifications and standard instructions, refer to the µPD70320/322 (V25) Data Sheet.

Features

- □ Security and normal operational modes
- □ System clock speeds to 8 MHz (16-MHz crystal)
- □ 16-bit CPU and internal data paths
- □ Functional compatibility with V25
- □ Software upward compatible with µPD8086
- □ New and enhanced V-Series instructions
- □ 6-byte prefetch queue
- □ Two-channel on-chip DMA controller
- □ Minimum instruction cycle: 250 ns at 8 MHz
- □ Internal 256-byte RAM memory
- □ 1-megabyte memory address space; 64K-byte I/O space
- □ Eight internal RAM-mapped register banks
- □ Four multifunction I/O ports
  - — 8-bit analog comparator port
  - — 20 bidirectional port lines
  - — Four input-only port lines
- □ Two independent full-duplex serial channels
- □ Priority interrupt controller
  - — Standard vectored service
  - — Register bank switching
  - — Macroservice
- □ Pseudo SRAM and DRAM refresh controller
- □ Two 16-bit timers
- □ On-chip time base counter
- □ Programmable wait state generator
- □ Two standby modes: STOP and HALT

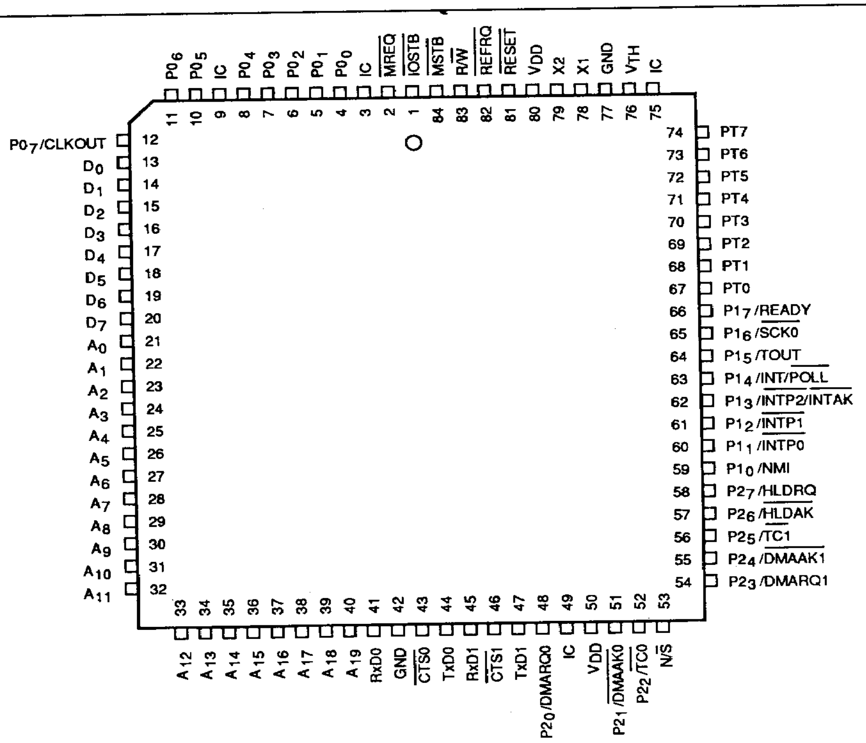
Ordering Information

Part Number	Clock (MHz)	Package
µPD70327L-8-xxx	8	84-pin PLCC
GJ-8-xxx	8	94-pin plastic QFP

V20 and V30 are registered trademarks of NEC Corporation.
V25 and V35 are trademarks of NEC Corporation.

Pin Configurations

84-Pin PLCC



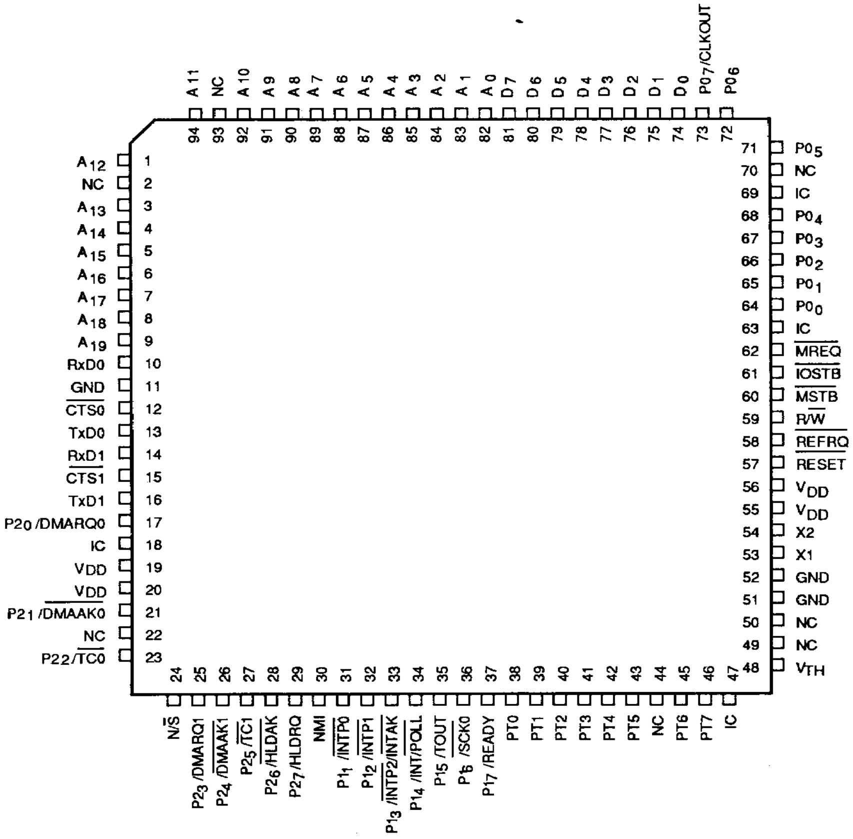
Notes:

(1) Pin functions are Identical to μPD70320.

(2) All IC pins should be tied together and pulled up to V_{DD} with a 10- to 20-kΩ resistor.

83SL-7039B

94-Pin Plastic QFP



Notes:

(1) Pin functions are identical to μPD70320.

(2) All IC pins should be tied together and pulled up to V_{DD} with a 10- to 20-kΩ resistor.

83SL-7040B

Pin Identification

Symbol	Function
A_0-A_{19}	Address bus output
$\overline{\text{CTS0}}$	Clear to send channel 0 input (Async mode); Receive clock input/output (I/O interface mode)
$\overline{\text{CTS1}}$	Clear to send channel 1 input
D_0-D_7	Bidirectional data bus
$\overline{\text{IOSTB}}$	I/O read or write strobe output
$\overline{\text{MREQ}}$	Memory request output
$\overline{\text{MSTB}}$	Memory read/write strobe output
N/$\overline{\text{S}}$	Normal mode/security mode select input
$P0_0$-$P0_6$	I/O port 0
$P0_7$/ CLKOUT	I/O port 0; System clock output
$P1_0$/NMI	Port 1 input line; Nonmaskable interrupt input
$P1_1$-$P1_2$/$\overline{\text{INTP0}}$-$\overline{\text{INTP1}}$	Port 1 input lines; External interrupt input lines
$P1_3$/$\overline{\text{INTP2}}$/$\overline{\text{INTAK}}$	Port 1 input line; External interrupt input line; Interrupt acknowledge output
$P1_4$/INT/$\overline{\text{POLL}}$	I/O port 1; Interrupt request input; Poll input
$P1_5$/TOUT	I/O port 1; Timer out
$P1_6$/$\overline{\text{SCK0}}$	I/O port 1; Serial clock output
$P1_7$/READY	I/O port 1; Ready input
$P2_0$/DMARQ0	I/O port 2; DMA request 0 input
$P2_1$/$\overline{\text{DMAAK0}}$	I/O port 2; DMA acknowledge 0 output
$P2_2$/$\overline{\text{TC0}}$	I/O port 2; DMA terminal count 0 output
$P2_3$/DMARQ1	I/O port 2; DMA request 1 input
$P2_4$/$\overline{\text{DMAAK1}}$	I/O port 2; DMA acknowledge 1 output
$P2_5$/$\overline{\text{TC1}}$	I/O port 2; DMA terminal count 1 output
$P2_6$/$\overline{\text{HLDAK}}$	I/O port 2; Hold acknowledge output
$P2_7$/HLDRQ	I/O port 2; Hold request input
PT0-PT7	Comparator port input lines
$\overline{\text{REFRQ}}$	DRAM refresh pulse output
$\overline{\text{RESET}}$	Reset input
RxD0	Serial receive data channel 0 input
RxD1	Serial receive data channel 1 input
R/$\overline{\text{W}}$	Read cycle/write cycle ID output
TxD0	Serial transmit data channel 0 output
TxD1	Serial transmit data channel 1 output
X1, X2	Crystal connection terminals
V_{DD}	+5-volt power supply; connect both pins
V_{TH}	Threshold voltage input
GND	Ground reference; connect both pins
IC	Internal connection

PIN FUNCTIONS

A_0-A_{19} (Address Bus)

A_0-A_{19} is the 20-bit address bus used to access all external devices.

CLKOUT (System Clock)

This is the internal system clock. It can be used to synchronize external devices to the CPU.

$\overline{\text{CTS}}$n, RxDn, TxDn, $\overline{\text{SCK0}}$ (Clear to Send, Receive Data, Transmit Data, Serial Clock Out)

The two serial ports (channels 0 and 1) use these lines for transmitting and receiving data, handshaking, and serial clock output.

D_0-D_7 (Data Bus)

D_0-D_7 is the 8-bit external data bus.

DMARQn, $\overline{\text{DMAAK}}$n, TCn (DMA Request, DMA Acknowledge, Terminal Count)

These are the control signals to and from the on-chip DMA controller.

$\overline{\text{HLDAK}}$ (Hold Acknowledge)

The $\overline{\text{HLDAK}}$ output (active low) informs external devices that the CPU has released the system bus.

HLDRQ (Hold Request)

The HLDRQ input (active high) is used by external devices to request the CPU to release the system bus to an external bus master. The following lines go into a high-impedance status with internal 4.7-kΩ pullup resistors: A_0-A_{19}, D_0-D_7, $\overline{\text{MREQ}}$, R/$\overline{\text{W}}$, $\overline{\text{MSTB}}$, $\overline{\text{REFRQ}}$, and $\overline{\text{IOSTB}}$.

INT (Interrupt Request)

INT is a maskable, active-high, vectored interrupt request . After assertion, external hardware must provide the interrupt vector number.

The INT pin allows direct connection of slave μPD71059 interrupt controllers.

$\overline{\text{INTAK}}$ (Interrupt Acknowledge)

After INT is asserted, the CPU will respond with $\overline{\text{INTAK}}$ (active low) to inform external devices that the interrupt request has been granted.

INTP0-INTP2 (External Interrupt)

INTP0-INTP2 allow external devices to generate interrupts. Each can be programmed to be rising or falling edge triggered.

$\overline{\text{IOSTB}}$ (I/O Strobe)

$\overline{\text{IOSTB}}$ is asserted during read and write operations to external I/O.

$\overline{\text{MREQ}}$ (Memory Request)

$\overline{\text{MREQ}}$ (active low) informs external memory that the current bus cycle is a memory access bus cycle.

$\overline{\text{MSTB}}$ (Memory Strobe)

$\overline{\text{MSTB}}$ (active low) is asserted during read and write operations to external memory.

NMI (Nonmaskable Interrupt)

NMI cannot be masked through software and is typically used for emergency processing. Upon execution, the interrupt starting address is obtained from interrupt vector number 2. NMI can release the standby modes and can be programmed to be either rising or falling edge triggered.

N/$\overline{\text{S}}$ (N Mode/S Mode)

Normal or security mode is selected by a fixed high level (N) or low level (S) at this pin. This pin is sampled at system reset.

$P0_0$-$P0_7$ (Port 0)

$P0_0$-$P0_7$ are the lines of port 0, an 8-bit bidirectional parallel I/O port, specifiable by bit.

$P1_0$-$P1_7$ (Port 1)

The status of $P1_0$-$P1_3$ can be read but these lines are always control functions. $P1_4$-$P1_7$ are the remaining lines of parallel port 1; each line is individually programmable as either an input, an output, or a control function.

$P2_0$-$P2_7$ (Port 2)

$P2_0$-$P2_7$ are the lines of port 2, an 8-bit bidirectional parallel I/O port specifiable by bit. The lines can also be used as control signals for the on-chip DMA controller.

$\overline{\text{POLL}}$ (Poll)

Upon execution of the POLL instruction, the CPU checks the status of this pin and, if low, program execution continues. If high, the CPU checks the level of the line every five clock cycles until it is low. $\overline{\text{POLL}}$ can be used to synchronize program execution to external conditions.

PT0-PT7 (Comparator Port)

PT0-PT7 are inputs to the analog comparator port.

READY (Ready)

After READY is de-asserted low, the CPU synchronizes and inserts at least two wait states into a read or write cycle to memory or I/O. This allows the processor to accommodate devices whose access times are longer than nominal μPD70325 bus cycles.

$\overline{\text{REFRQ}}$ (Refresh)

This active-low output pulse can refresh nonstatic RAM. It can be programmed to meet system specifications and is internally synchronized so that refresh cycles do not interfere with normal CPU operation.

$\overline{\text{RESET}}$ (Reset)

A low on $\overline{\text{RESET}}$ resets the CPU and all on-chip peripherals. $\overline{\text{RESET}}$ can also release the standby modes. After $\overline{\text{RESET}}$ returns high, program execution begins from address FFFF0H.

R/$\overline{\text{W}}$ (Read/Write)

R/$\overline{\text{W}}$ output allows external hardware to determine if the current operation is a read or a write cycle. It can also control the direction of bidirectional buffers.

TOUT (Timer Out)

TOUT is the square-wave output signal from the internal timer.

X1, X2 (Crystal Connections)

The internal clock generator requires an external crystal across these terminals. By programming the PRC register, the system clock frequency can be selected as the oscillator frequency (f_{OSC}) divided by 2, 4, or 8.

V_{DD} (Power Supply)

Two positive power supply pins (V_{DD}) reduce internal noise.

V_{TH} (Threshold Voltage)

The comparator port uses this pin to determine the analog reference point. The actual threshold to each comparator line is programmable to V_{TH} x n/16 where n =1 to 16.

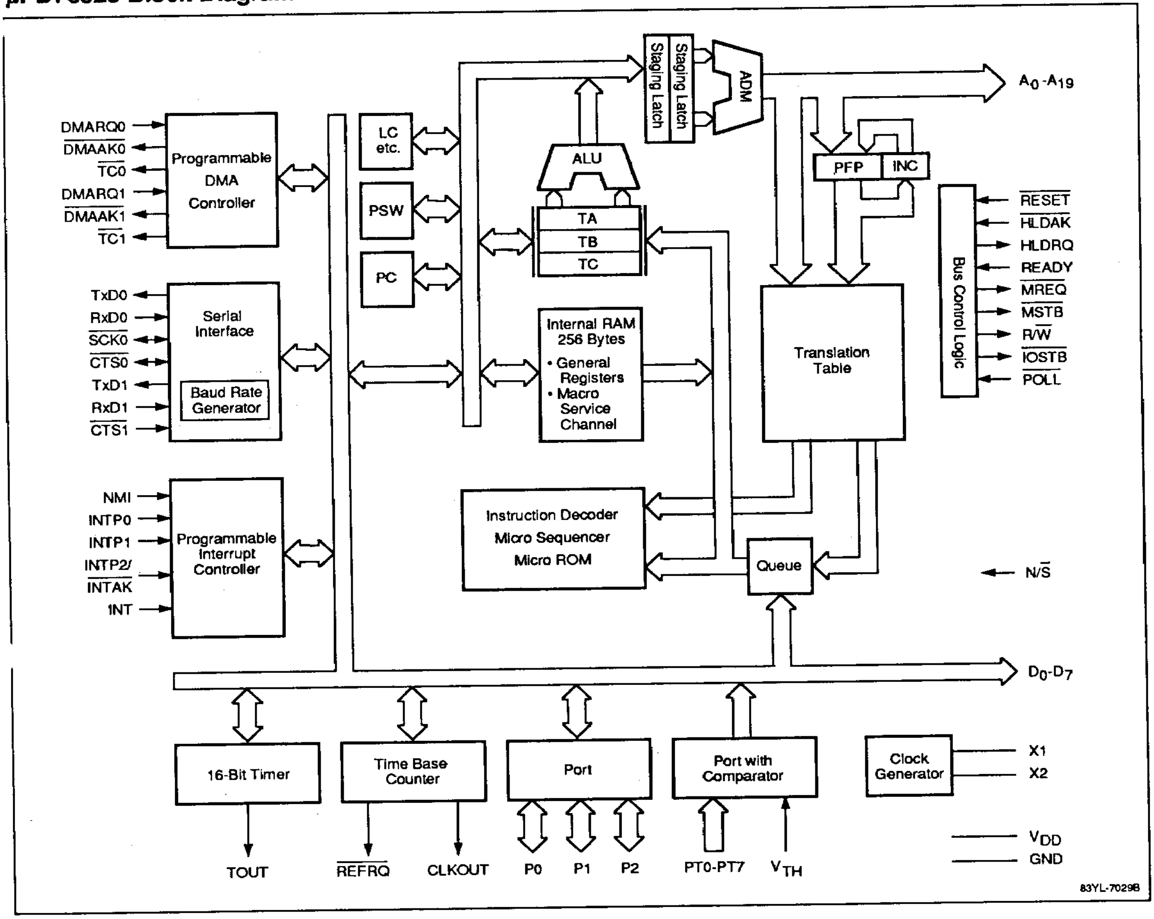
GND (Ground)

Two ground connections reduce internal noise.

IC (Internal Connection)

All IC pins should be tied together and pulled up to V_{DD} with a 10- to 20-kΩ resistor.

µPD70325 Block Diagram



FUNCTIONAL DESCRIPTION

The following architectural enhancements enable the μPD70327 to perform high-speed execution of instructions.

- Dual internal data bus
- 16- and 32-bit temporary registers/shifters
- 16-bit loop counter
- Program counter and prefetch pointer

Dual Data Bus

The μPD70327 has two internal 16-bit data buses, main and secondary. This reduces the processing time required for addition/subtraction and logical comparison instructions by one third over single-bus systems. The dual data bus method allows two operands to be fetched simultaneously from the general-purpose registers and transferred to the ALU.

16- and 32-Bit Temporary Registers/Shifters

The 16-bit temporary registers/shifters (TA and TB) allow high-speed execution of multiplication/division and shift/rotate instructions. Using the temporary registers, the μPD70327 can execute multiplication/division instructions about four times faster than with the microprogrammed method.

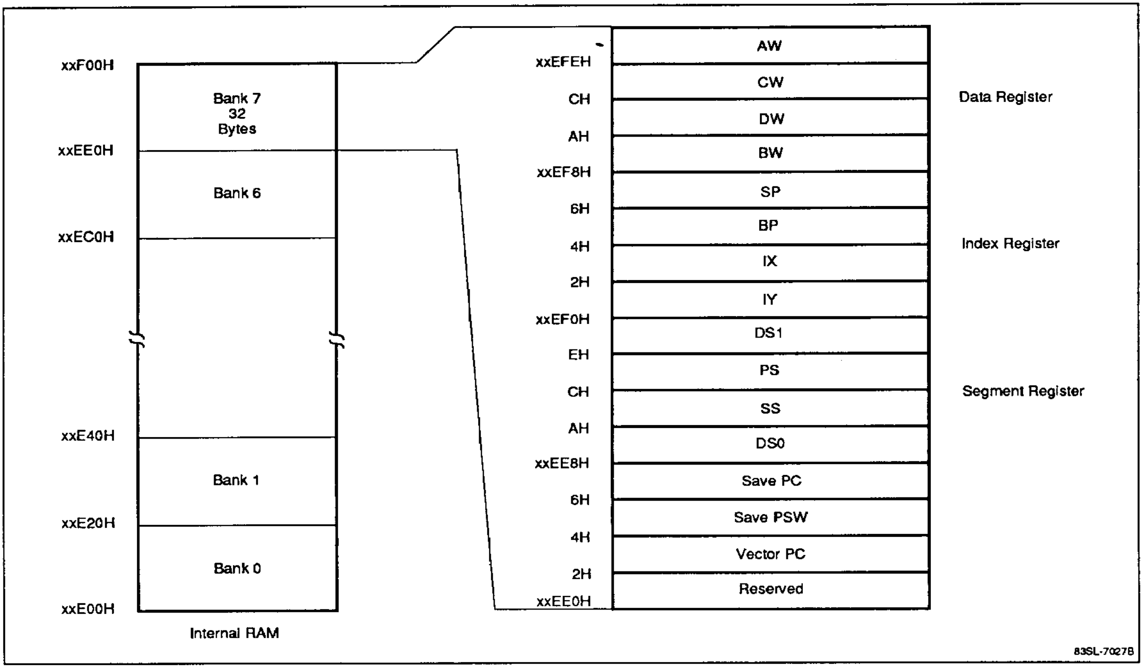
Loop Counter (LC)

The dedicated hardware loop counter (LC) counts the number of iterations for string operations and the number of shifts performed for multiple-bit shift/rotate instructions. The loop counter works with internal dedicated shifters to speed the processing of multiplication/division instructions.

Program Counter and Prefetch Pointer (PC and PFP)

The hardware PC addresses the memory location of the instruction to be executed next. The hardware PFP addresses the program memory location to be accessed by the instruction queued next. Several clock cycles are saved for branch, call, return, and break instructions.

Figure 1. µPD70327 Internal RAM Register Mapping



83SL-7027B

Register Set

Figure 1 shows the eight banks of internal registers, which the µPD70327 maps into internal RAM. Each bank contains general-purpose registers, pointer and index registers, segment registers, and save areas for context switching.

Although these memory locations may be accessed as normal RAM with the full set of memory addressing modes provided by the V25 family, the capability of context switching provides superior speed in register access. When used in the internal memory disabled state, many instructions execute considerably faster.

The eight macroservice channel control blocks are mapped into register banks 0 and 1. The µPD70327 also maps the DMA control registers over macroservice channels 0 and 1 within register bank 0.

As a result of this mapping, interrupt priorities, register banks, and macroservice channels should be allocated from the lowest priority to the highest (that is, starting from bank/priority 7 and proceeding to bank priority 0). Be careful not to map user RAM locations and/or DMA or macroservice control registers into the same internal RAM locations.

General-Purpose Registers. Four 16-bit general-purpose registers (AW, BW, CW, and DW) can serve as 16-bit registers or as four sets of dual 8-bit registers (AH, AL, BH, BL, CH, CL, DH, and DL). The instruction classes default to the following general-purpose registers.

AW Word multiplication/division, word I/O, data conversion.

AL Byte multiplication/division, byte I/O, BCD rotation, data conversion, translation.

AH Byte multiplication/division.

BW Translation

CW Loop control, branch, and repeat prefixes.

CL Shift instructions, rotate instructions, BCD operations.

DW Word multiplication/division, indirect I/O addressing.

Pointers and Index Registers. These registers are 16-bit base pointers (SP, BP) or index registers (IX, IY) in based addressing, indexed addressing, and based indexed addressing. They are used as default registers under the following conditions.

SP	Stack operations
IX	Block transfer (source), BCD string operations
IY	Block transfer (destination), BCD string operations

Segment Registers. The segment registers divide the 1M-byte address space into 64K-byte blocks. Each segment register functions as a base address to a block; the effective address is an offset from that base. Physical addresses are generated by shifting the associated segment register left by four binary digits and then adding the offset address. The segment registers and default offsets are listed below.

Segment Register	Default Offset
PS (Program Segment)	PC (Program Counter)
SS (Stack Segment)	SP and Effective Address
DS0 (Data Segment 0)	IX and Effective Address
DS1 (Data Segment 1)	IY and Effective Address

Save Registers. Save PC and save PSW are used as the storage areas during register bank context-switching operations. The Vector PC save location contains the effective address of the interrupt service routine when register bank switching is used to service interrupts.

Program Counter. The PC is a 16-bit binary counter that contains the offset address from the program segment of the next instruction to be executed. It is incremented every time an instruction is received from the queue. It is loaded with a new location whenever the branch, call, return, break, or interrupt is executed.

Program Status Word. The PSW contains status and control flags used by the CPU and two general-purpose user flags. The configuration of this 16-bit register is shown below.

15							8
MD	RB2	RB1	RB0	V	DIR	IE	BRK

7							0
S	Z	F1	AC	F0	P	BRKI	CY

Status Flags	
V	Overflow bit
S	Sign
Z	Zero
AC	Auxiliary carry
P	Parity
CY	Carry

Control Flags	
DIR	Direction of string processing
IE	Interrupt enable
BRK	Break (after every instruction)
RBn	Current register bank flags
BRKI	I/O trap enable
F0, F1	General-purpose user flags (accessed by flag SFR)
MD	Normal/Security mode select

The eight low-order bits of the PSW can be stored in register AH and restored using a MOV instruction. The only way to alter the RBn bits with software is to execute one of the special bank switch instructions.

Memory Map

The μPD70327 has a 20-bit address bus that can directly access 1M-byte memory. Unlike the standard V25, the V25 Software Guard does not support internal ROM.

The internal data area (IDA) is a 256-byte internal RAM area followed consecutively by a 256-byte special function register (SFR) area.

All the data and control registers for on-chip peripherals and I/O are mapped into the SFR area and accessed as RAM.

The IDA is dynamically relocatable in 4K-byte increments by changing the value in the internal data base (IDB) register. The value in this register is assigned as the uppermost eight bits of the IDA address. The IDB register is accessed from two memory locations, FFFFFH and XXFFFH, where XX is the value in the IDB register.

On reset, the internal data base register is set to FFH, which maps the IDA into the uppermost area of memory.

In large-scale systems where internal RAM is not required for data memory, internal RAM can be removed completely from the address space and dedicated entirely to registers and control functions such as macroservice and DMA channels. Clearing the RAMEN bit in the processor control register does this. When the RAMEN bit is cleared, internal RAM can only be accessed by register addressing or internal control processes. Many instructions execute faster when internal RAM is disabled.

INSTRUCTIONS

The V25 family instruction set is fully compatible with the V20 native mode. The V20 instruction set is a superset of the μPD8086/8088 instruction set with different execution times and mnemonics.

The μPD70327 does not support the V20 8080 emulation mode. All instructions pertaining to this mode have been deleted from the V25 family instruction set.

Enhanced Instructions

In addition to the basic V20 instruction set, the μPD70327 supports numerous enhanced instructions, which together form the standard V-Series instruction set. These enhanced instructions are described in full in the V25/V35 User's Manual and are listed below.

PUSH imm
PUSH R
POP R
MUL imm
SHL imm8
ROL imm8
ROLC imm8
CHKIND
INM
OUTM
PREPARE
DISPOSE
ROR imm8
RORC imm8

Unique Instructions

The μPD70327 supports a set of unique instructions, which are also discussed in the V25/V35 User's Manual. These instructions belong to one of the following groups: Variable Length Bit Field Operations, Packed BCD Instructions, Bit Manipulation Instructions, Repeat Prefixes, and Special V25 Family Register Bank Switching Instructions.

INS
ADD4S
ROL4
TEST1
CLR1
BTCLR
REPNC
TSKSW reg16
MOVSPB
EXT
CMP4S
SET1
ROR4
NOT1
REPC
BRKCS reg16
MOVSPA
RETRBI

INTERRUPT STRUCTURE

The μPD70327 can service interrupts generated by both hardware and software. Software interrupts are serviced through vectored interrupt processing. The following interrupts are provided.

Divide Error
Overflow
BRK imm8
Escape Trap
Single Step
BRK3
Array Bounds
I/O Trap

When executing software written for another system, it is preferable to implement I/O using the on-chip peripherals. However, since the μPD70327 peripherals are memory mapped, some software conversion may be required. This mapping strategy allows the internal peripherals to be accessed using all standard addressing modes, including the advanced bit and bit-field manipulation instructions of the μPD70327. Also, the I/O trap feature of the μPD70327 allows an easy hardware solution to remapping external devices to internal μPD70327 peripherals.

Interrupt Vectors

Table 1 lists the interrupt vectors beginning at physical address 00H. External memory is required to service these routines. By servicing interrupts via the macroservice function or context switching, this requirement can be eliminated.

Each interrupt vector is 4 bytes wide. To service a vectored interrupt, the lower addressed word is transferred to the PC and the upper word to the PS. The pseudocode for the vectored interrupt service overhead is shown below.

```
(SP-1, SP-2) ← PSW
(SP-3, SP-4) ← PS
(SP-5, SP-6) ← PC
SP ← SP-6
IE ← 0, BRK ← 0
PS ← vector high bytes
PC ← vector low bytes
```

Table 1. Interrupt Vectors

Address	Vector	Assigned Use
00	0	Divide error
04	1	Break flag
08	2	NMI
0C	3	BRK3 instruction
10	4	BRKV instruction
14	5	CHKIND instruction
18	6	General purpose
1C	7	FPO instructions
20-2C	8-11	General purpose
30	12	INTSER0 (Interrupt serial error, channel 0)
34	13	INTSR0 (Interrupt serial receive, channel 0)
38	14	INTST0 (Interrupt serial transmit, channel 0)

Table 1. Interrupt Vectors (cont)

Address	Vector	Assigned Use
3C	15	General purpose
40	16	INTSER1 (Interrupt serial error, channel 1)
44	17	INTSR1 (Interrupt serial receive, channel 1)
48	18	INTST1 (Interrupt serial transmit, channel 1)
4C	19	I/O trap
50	20	INTD0 (Interrupt from DMA, channel 0)
54	21	INTD1 (Interrupt from DMA, channel 1)
58	22	General purpose
5C	23	General purpose
60	24	INTP0 (Interrupt from peripheral 0)
64	25	INTP1 (Interrupt from peripheral 1)
68	26	INTP2 (Interrupt from peripheral 2)
6C	27	General purpose
70	28	INTTU0 (Interrupt from timer unit 0)
74	29	INTTU1 (Interrupt from timer unit 1)
78	30	INTTU2 (Interrupt from timer unit 2)
7C	31	INTTB (Interrupt from time base counter)
080-3FF	32- 255	General purpose

Hardware Interrupt Configuration

The μPD70327 features a high-performance on-chip controller capable of controlling multiple processing for interrupts from up to 17 different sources (5 external, 12 internal). The interrupt configuration includes system interrupts that are functionally compatible with those of the V20/V30. In addition, two unique high-speed microcontroller interrupt servicing methods are available.

Interrupt Sources

The 17 interrupt sources are divided into groups for management by the interrupt controller. Using software, each of the groups can be assigned a priority from 0 (highest) to 7 (lowest). The priority of individual interrupts within a group is fixed in hardware.If interrupts from different groups occur simultaneously and the groups have the same priority level, the priority followed will be as shown in the Default Priority column of table 2.

Table 2. Interrupt Sources

	Interrupt Source (Priority Within Group)			
Group	**1**	**2**	**3**	**Default Priority**
Nonmaskable interrupt	NMI	–	–	0
Timer unit	INTTU0	INTTU1	INTTU2	1
DMA controller	INTD0	INTD1	–	2
External peripheral interrupt	INTP0	INTP1	INTP2	3
Serial channel 0	INTSER0	INTSR0	INTST0	4
Serial channel 1	INTSER1	INTSR1	INTST1	5
Time base counter	INTTB	–	–	6
Interrupt request	INT	–	–	7

The priority of the currently active interrupt is stored in the ISPR special function register. Bits PR_7-PR_0 correspond to the eight possible interrupt request priority groups. The ISPR keeps track of the priority of active interrupts by setting the appropriate bit of this register. The address of this 8-bit register is xxFFCH, and the format is shown below. Again it is recommended that priorities be assigned starting with the lowest (programmed to 6 or 7) and proceed up in priority to minimize the risk of overlapping macroservice and register bank switch interrupt service control blocks.

PR_7	PR_6	PR_5	PR_4	PR_3	PR_2	PR_1	PR_0

NMI and INT are system-type external vectored interrupts. NMI is not maskable by software. INT is maskable by the IE bit in the PSW and requires that an external device provide the interrupt vector number. It is designed to allow the interrupt controller to be expanded by the addition of an external interrupt controller such as the μPD71059.

NMI, INTP0, and INTP1 are edge-sensitive inputs. By selecting the appropriate bits in the interrupt mode register, these inputs can be programmed to be either rising- or falling-edge triggered. Bits ES0-ES2 correspond to INTP0-INTP2, respectively, as shown in figure 2.

Figure 2. External Interrupt Mode Register (INTM)

0	ES2	0	ES1	0	ES0	0	ESNMI
7			Address xxF40H				0

ES2	INTP2 Input Effective Edge
0	Falling edge
1	Rising edge
ES1	**INTP1 Input Effective Edge**
0	Falling edge
1	Rising edge
ES0	**INTP0 Input Effective Edge**
0	Falling edge
1	Rising edge
ESNMI	**NMI Input Effective Edge**
0	Falling edge
1	Rising edge

Interrupt Processing Modes

Interrupts, with the exception of NMI, INT, and INTTB have high-performance capability and can be processed in any of three modes: standard vector interrupt, register bank context switching and macroservice. The processing mode for a given interrupt can be chosen by enabling the appropriate bits in the corresponding interrupt request control register. Each interrupt, except INT and NMI, has its own associated IRC register. The general format for each of these registers is shown in figure 3.

All interrupt processing routines other than those for NMI and INT must end with the execution of the FINT instruction. Otherwise, subsequently, only interrupts of higher priority will be accepted.

In the vectored service mode, the CPU traps to a vector location.

Figure 3. Interrupt Request Control Registers (IRC)

IF	IMK	MS/INT	ENCS	0	PR_2	PR_1	PR_0
7							0

IF	Interrupt Flag
0	No interrupt request generated
1	Interrupt request generated
IMK	**Interrupt Mask**
0	Open (interrupts enabled)
1	Closed (interrupts disabled)
MS/INT	**Interrupt Response Method**
0	Vector interrupt or register bank switching
1	Macroservice function
ENCS	**Register Bank Switching Function**
0	Not used
1	Used
PR_2-PR_0	**Interrupt Group Priority (0-7)**
0 0 0	Highest (0) ↓
1 1 1	Lowest (7)

Register Bank Switching.

Register bank context switching allows interrupts to be processed rapidly by switching register banks. After an interrupt, the new register bank selected has the same bank number (0-7) as the priority programmed in the associated IRC register. The PC and PSW are automatically saved in the save areas of the new register bank, and the address of the interrupt routine is loaded from the vector PC storage register in the new register bank.

As in the vectored mode, the IE and BRK bits in the PSW are cleared to zero. After interrupt processing, execution of the RETRBI instruction returns control to the original register bank and restores the former PC and PSW. Figures 4 and 5 show register bank context switching and register bank return.

This method of interrupt service offers a dramatic performance advantage over normal vectored service because there is no need to store and retrieve data/ registers on the stack. This also allows hardware-based real-time task switching in high-speed environments.

In addition to context switching, the μPD70327 has a task switch opcode (TSKSW) that allows multiple independent processes to be internally resident. Figure 6 shows the task switching function.

Figure 4. Register Bank Context Switching

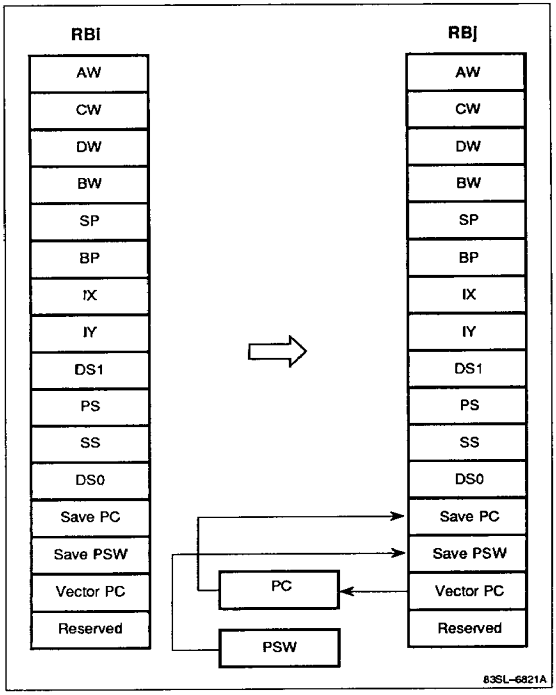
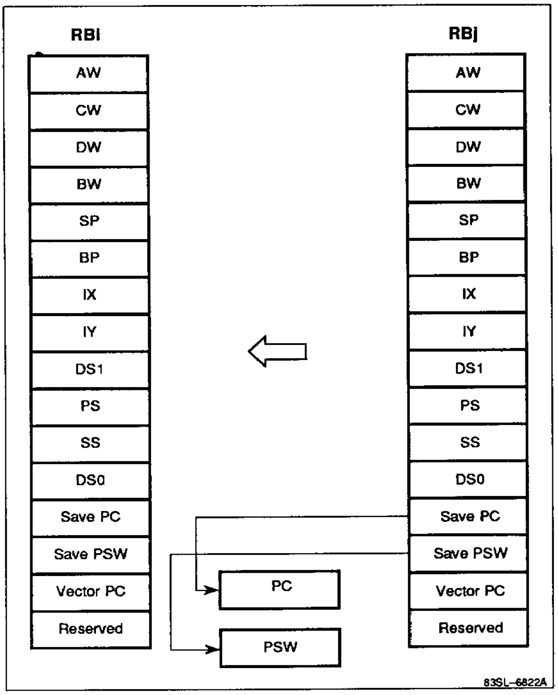


Figure 5. Register Bank Return



Execution of a vectored interrupt occurs as follows:

(SP-1, SP-2) ← PSW
(SP-3, SP-4) ← PS
(SP-5, SP-6) ← PC
SP ← SP-6
IE ← 0, BRK ← 0
PS ← vector high bytes
PC ← vector low bytes

Hardware Interrupt Configuration

The V35 Plus features a high-performance on-chip controller capable of controlling multiple processing for interrupts from up to 17 different sources (5 external, 12 internal). The interrupt configuration includes system interrupts that are functionally compatible with those of the V20/V30 and unique high-performance microcontroller interrupts.

Interrupt Sources

The interrupt sources on the V35 Plus are similar to those on the V35. The 17 interrupt sources (table 3) are divided into groups for management by the interrupt controller. Using software, each of the groups can be assigned a priority from 0 (highest) to 7 (lowest). The priority of individual interrupts within a group is fixed in hardware.

The ISPR is an 8-bit SFR; bits PR_0-PR_7 correspond to the eight possible interrupt request priorities. The ISPR keeps track of the priority of the interrupt currently being serviced by setting the appropriate bit. The address of the ISPR is XXFFCH. The ISPR format is shown below.

PR_7	PR_2	PR_5	PR_4	PR_3	PR_2	PR_1	PR_0

NMI and INT are system-type external vectored interrupts. NMI is not maskable via software. INTR is maskable (IE bit in PSW) and requires that an external device provide the interrupt vector number. It allows expansion by the addition of an external interrupt controller (µPD71059).

NMI, INTP0, and INTP1 are edge-sensitive maskable interrupt inputs. By selecting the appropriate bits in the interrupt mode register, these inputs can be programmed to be either rising or falling edge triggered. ES0-ES2 correspond to INTP0-INTP2, respectively. See figure 5.

Figure 5. External Interrupt Mode Register (INTM)

0	ES2	0	ES1	0	ES0	0	ESNMI
7							0

ES2	INTP2 Input Effective Edge
0	Falling edge
1	Rising edge
ES1	**INTP1 Input Effective Edge**
0	Falling edge
1	Rising edge
ES0	**INTP0 Input Effective Edge**
0	Falling edge
1	Rising edge
ESNMI	**NMI Input Effective Edge**
0	Falling edge
1	Rising edge

Interrupt Factor Register

The primary enhancement of the V35 Plus interrupt control unit is the addition of a special function register that stores the vector type that last caused an interrupt. This IRQS register (figure 5A) stores the vector until the next interrupt request is accepted, but is not changed by response to NMI, INT, or macroservice interrupts.

The main purpose of the IRQS register is to allow several interrupts within a given priority level to be serviced with context switching. Once the interrupt service routine is executing, the cause of the interrupt can be determined only by reading this register, rather than by long and time-consuming software determination. It is recommended that the contents of the IRQS register be read before interrupts are re-enabled to avoid confusion within multiprocessing environments.

Setting the macroservice mode requires programming the corresponding macroservice control register. Each individual interrupt, excluding INT, NMI, serial error, DMA, and TBC, has its own associated MSC register. Figure 8 shows the generic format for all MSC registers.

Figure 8. Macroservice Control Registers (MSC)

MSM_2	MSM_1	MSM_0	DIR	0	CH_2	CH_1	CH_0
7							0

MSM_2-MSM_0	Macroservice Mode
0 0 0	Normal (8-bit transfer)
0 0 1	Normal (16-bit transfer)
1 0 0	Character search (8-bit transfer)
	Other combinations are not allowed.
DIR	**Data Transfer Direction**
0	Memory to SFR
1	SFR to memory
CH_2-CH_0	**Macroservice Channel**
0 0 0	Channel 0
	↓
1 1 1	Channel 7

TIMER UNIT

The μPD70327 (figure 9) has two programmable 16-bit interval timers (TM0 and TM1) on chip, each with variable input clock frequencies. Each of the two 16-bit timer registers has an associated 16-bit modulus register (MD0 and MD1). Timer 0 operates in the interval timer mode or one-shot mode; timer 1 has only the interval timer mode.

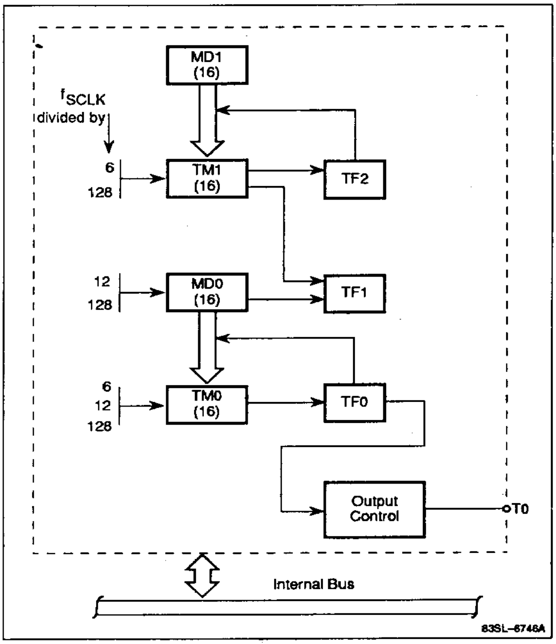
Interval Timer Mode

In this mode, TM0/TM1 are decremented by the selected input clock, and, after counting out, the registers are automatically reloaded from the modulus registers and counting continues. Each time TM1 counts out, interrupts are generated through TF1 and TF2 (timer flags 1 and 2). When TM0 counts out, an interrupt is generated through TF0. The timer-out signal can be used as a square-wave output whose half-cycle is equal to the count time.

Two input clocks derived from the system clock are SCLK/6 and SCLK/128. Typical timer values shown below are based on f_{OSC} = 10 MHz and f_{SCLK} = $f_{OSC}/2$.

Clock	Timer Resolution	Full Count
SCLK/6	1.2 μs	78.643 μs
SCLK/128	25.6 μs	1.678 s

Figure 9. Timer Unit Block Diagram



One-Shot Mode

In the one-shot mode, TM0 and MD0 operate as independent one-shot timers. Starting with a preset value, each is decremented to zero. At zero, counting ceases and an interrupt is generated by TF0 (from TM0) or TF1 (from MD0).

When TM0 is programmed to one-shot mode, TM1 may still operate in interval mode.

Two input clocks derived from the system clock are SCLK/12 and SCLK/128. Typical timer values shown below are based on f_{OSC} = 10 MHz and f_{SCLK} = $f_{OSC}/2$.

Clock	Timer Resolution	Full Count
SCLK/12	2.4 μs	157.283 ms
SCLK/128	25.6 μs	1.678 s

Timer Control Registers

Setting the desired timer mode requires programming the timer control register. See figures 10 and 11 for the TMC register format.

Figure 10. Timer Control Register 0 (TMC0)

TS0	TCLK0	MS0	MCLK0	ENT0	ALV	MOD$_1$	MOD$_0$

7 Address xxF90H 0

TS0	TM0 in Either Mode
0	Stop countdown
1	Start countdown

MOD$_1$	MOD$_0$	TCLK0	TM0 Register Clock Frequency
0	0	0	$f_{SCLK}/6$ (Interval)
0	0	1	$f_{SCLK}/128$ (Interval)
0	1	0	$f_{SCLK}/12$ (One-shot)
0	1	1	$f_{SCLK}/128$ (One-shot)

MS0	MD0 Register Countdown (One-Shot Mode)
0	Stop
1	Start

MCLK0	MD0 Register Clock Frequency
0	$f_{SCLK}/12$
1	$f_{SCLK}/128$

ENT0	TOUT Square-Wave Output
0	Disable
1	Enable

ALV	TOUT Initial Level When TOUT Disabled by ENT0 = 0
0	Low
1	High

MOD$_1$	MOD$_0$	Timer Unit Mode
0	0	Interval timer
0	1	One-shot
1	X	Reserved

Figure 11. Timer Control Register 1 (TMC1)

TS1	TCLK1	0	0	0	0	0	0

7 Address xxF91H 0

TS1	Timer 1 Countdown
0	Stop
1	Start

TCLK1	Timer 1 Clock Frequency
0	$f_{SCLK}/6$
1	$f_{SCLK}/128$

TIME BASE COUNTER

The 20-bit free-running time base counter (TBC) controls internal timing sequences and is available to the user as the source of periodic interrupts at lengthy intervals. One of four interrupt periods can be selected by programming the TB0 and TB1 bits in the processor control register (PRC). The TBC interrupt is unlike the others because it is fixed as a level 7 vectored interrupt. Macroservice and register bank switching cannot be used to service this interrupt. See figures 12 and 13.

Figure 12. Time Base Interrupt Request Control Register (TBIC)

TBF	TBMK	0	0	0	1	1	1

7 Address xxFECH 0

TBF	Time Base Interrupt Flag
0	No interrupt generated
1	Interrupt generated

TBMK	Time Base Interrupt Mask
0	Unmasked
1	Masked

Figure 13. Processor Control Register (PRC)

0	RAMEN	0	0	TB$_1$	TB$_0$	PCK$_1$	PCK$_0$

7 Address xxFEBH 0

RAMEN	Built-In RAM
0	Disable
1	Enable

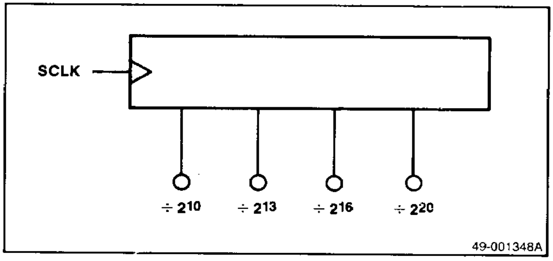
TB$_1$	TB$_0$	Time Base Interrupt Period
0	0	$2^{10}/f_{SCLK}$
0	1	$2^{13}/f_{SCLK}$
1	0	$2^{16}/f_{SCLK}$
1	1	$2^{20}/f_{SCLK}$

PCK$_1$	PCK$_0$	System Clock Frequency (f_{CLK})
0	0	$f_{OSC}/2$
0	1	$f_{OSC}/4$
1	0	$f_{OSC}/8$
1	1	Reserved

The RAMEN bit in the PRC register allows the internal RAM to be removed from the memory address space to implement faster instruction execution.

The TBC (figure 14) uses the system clock as the input frequency. The system clock can be changed by programming the PCK0 and PCK1 bits in the processor control register (PRC). Reset initializes the system clock to $f_{OSC}/8$ (f_{OSC} = external oscillator frequency).

Figure 14. Time Base Counter (TBC) Block Diagram



REFRESH CONTROLLER

The μPD70327 has an on-chip refresh controller for dynamic and pseudostatic RAM memory. The refresh controller generates refresh cycles between the normal CPU bus cycles according to the refresh specifications programmed.

The refresh controller outputs a 9-bit refresh address on address bits A_8-A_0 during the refresh bus cycle. Address bits A_{19}-A_9 are fixed to 0s during this cycle. The 9-bit refresh address is automatically incremented at every refresh cycle for 512 row addresses. The 8-bit refresh mode (RFM) register (figure 15) specifies the refresh operation and allows refresh during both CPU HALT and $\overline{\text{HOLD}}$ modes. Refresh cycles are automatically timed to $\overline{\text{REFRQ}}$ following read/write cycles to minimize the effect on system throughput.

The following shows the $\overline{\text{REFRQ}}$ pin level in relation to bits 4 (RFEN) and 7 (RFLV) of the refresh mode register.

RFEN	RFLV	$\overline{\text{REFRQ}}$ Level
0	0	0
0	1	1
1	0	0
1	1	Refresh pulse output

Figure 15. Refresh Mode Register (RFM)

RFLV	HLDRF	HLTRF	RFEN	RFW_1	RFW_0	RFT_1	RFT_0
7			Address xxFE1H				0

RFLV	RFEN	$\overline{\text{REFRQ}}$ Output Signal Level
0	0	0
1	0	1
0	1	0
1	1	Refresh pulse

HLDRF	Automatic Refresh Cycle in HOLD Mode
0	Disabled
1	Enabled

HLTRF	Automatic Refresh Cycle in HALT Mode
0	Disabled
1	Enabled

RFEN	Automatic Refresh Cycle
0	Refresh pin = RFLV
1	Refresh enabled

RFW_1	RFW_0	No. of Wait States Inserted in Refresh Cycle
0	0	0
0	1	1
1	0	2
1	1	2

RFT_1	RFT_0	Refresh Period
0	0	16/SCLK
0	1	32/SCLK
1	0	64/SCLK
1	1	128/SCLK

SERIAL INTERFACE

The μPD70327 has two full-duplex UARTs, channels 0 and 1. Each channel (figure 16) has a transmit line (TxDn), a receive line (RxDn), and a clear-to-send (CTSn) handshaking line. Communication is synchronized by a start bit, and either even, odd, or no parity may be programmed. Character length may be programmed to either 7 or 8 bits, and either 1 or 2 stop bits.

Each serial channel of the μPD70327 has a dedicated baud rate generator, so there is no need to obligate any of the on-chip timers to handle this function. The baud rate generators allow individual transfer rates for each channel and support rates up to 1.25 Mb/s. All standard baud rates are available and are not restricted by the value of the particular external crystal.

Each baud rate generator has an 8-bit data register (BRGn) that functions as a prescaler to a programmable input clock selected by the serial communication control register (SCCn). Together these must be set to generate a frequency equivalent to the desired baud rate.

The baud rate generator can be programmed to obtain the desired transmission rate according to the following formula:

$$B \times G = \frac{SCLK \times 10^6}{2^{n+1}}$$

where:

B = baud rate
G = baud rate generator register (BRGn) value
n = input clock specification; the value loaded into the SCCn register (n = 0 to 8)
SCLK = system clock frequency (MHz)

Based on the above formula, the following table shows the baud rate generator values used to obtain standard transmission rates when SCLK = 5 MHz.

Baud Rate	n	BRGn	Error (%)
110	7	178	0.25
150	7	130	0.16
300	6	130	0.16
600	5	130	0.16
1200	4	130	0.16
2400	3	130	0.16
4800	2	130	0.16
9600	1	130	0.16
19.2k	0	130	0.16
38.4k	0	65	0.16
1.25M	0	2	0.00

In addition to the asynchronous mode, channel 0 has a synchronous I/O interface mode. In this mode, each bit of data transferred is synchronized to a serial clock (SCLK0). This mode is compatible with the μCOM75 and μCOM87 series, and allows direct interfacing to these devices.

Figures 17, 18, and 19 show the three serial communication registers: SCM, SCC, and SCE.

Figure 16. Serial Interface Block Diagram

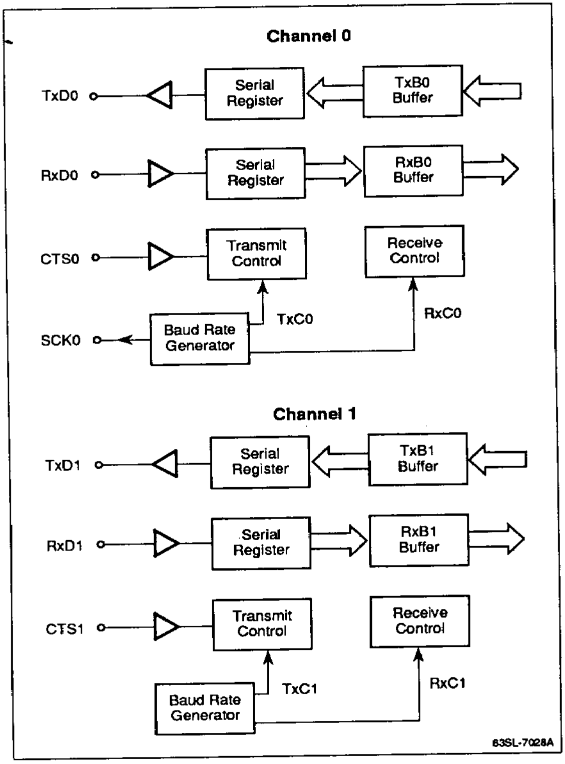


Figure 17. Serial Communication Mode Registers (SCM)

TxRDY	RxB	PRTY1	PRTY0	CLTSK	SLRSCK	MD1	MD0
7							0

TxRDY	Transmitter Control
0	Disabled
1	Enabled
RxB	**Receiver Control**
0	Disabled
1	Enabled
PRTY1-PRTY0	**Parity Control**
0 0	No parity
0 1	0 parity (Note 1)
1 0	Odd parity
1 1	Even parity
CLTSK	**Character Length (Async Mode) Tx Shift Clock (I/O Interface Mode)**
0	7 bits (Async) No effect (I/O intfc)
1	8 bits (Async) Trigger transmit (I/O intfc)
SLRSCK	**Stop Bits (Async Mode) Receiver Clock (I/O Interface Mode)**
0	1 stop bit (Async) Ext clock input on CTS0 (I/O intfc)
1	2 stop bits (Async) Int clock output on CTS1 (I/O intfc)
MD1-MD0	**Mode**
0 0	I/O interface (Note 2)
0 1	Asynchronous
1 x	Reserved

Notes:

(1) Parity is 0 during transmit and ignored during receive.

(2) I/O interface mode only.

(3) Channel 0 only.

Figure 18. Serial Communication Control Register (SCC)

0	0	0	0	PRS_3	PRS_2	PRS_1	PRS_0
7							0

PRS_3-PRS_0	Baud Rate Generator Input Clock Frequency
0000	$f_{SCLK}/2$ (n = 0)
0001	$f_{SCLK}/4$
0010	$f_{SCLK}/8$
0011	$f_{SCLK}/16$
0100	$f_{SCLK}/32$
0101	$f_{SCLK}/64$
0110	$f_{SCLK}/128$
0111	$f_{SCLK}/256$
1000	$f_{SCLK}/512$ (n = 8)

Figure 19. Serial Communications Error Register (SCE)

RxDn	0	0	0	0	ERP	ERF	ERO
7							0

RxDn	Receive Terminal State
0, 1	Status of RxD pin
ERP	**Parity Error Flag**
0	No error
1	Transmit and receive parity are different
ERF	**Framing Error Flag**
0	No error
1	Stop bit not detected
ERO	**Overrun Error Flag**
0	No error
1	Data is received before receive buffer outputs previous data

DMA CONTROLLER

The µPD70327 has a two-channel, on-chip Direct Memory Access (DMA) controller. This allows rapid data transfer between memory and auxiliary devices. The DMA controller supports four modes of operation, two for memory-to-memory transfers and two for I/O to memory; in all cases, transfer direction is programmable.

Memory-to-Memory Transfers

In the single-step mode, a single DMA request will commence the alternation of one DMA cycle with one CPU cycle until the prescribed number of transfers (terminal count) is reached. Interrupts are accepted while in this mode.

Alternatively, in the burst mode, one DMA request causes DMA transfer cycles to continue consecutively until the DMA terminal count decrements to zero. Software can initiate memory-to-memory transfers.

I/O-to-Memory Transfers

The single transfer mode will yield exactly one DMA transfer per DMA request. After the transfer, the bus is returned to the CPU. Alternatively, in demand release mode, the rising edge of DMARQ enable DMA cycles which continue while the DMA request remains active.

DMA Registers

Figures 20 and 21 show the DMA mode registers (DMAM) and the DMA address control registers (DMAC).

In all modes, the TC (Terminal Count) output pin will pulse low and a DMA end-of-service interrupt request will be internally generated after the programmed number of transfers have been completed. The bottom of internal RAM contains all the necessary address information for the designated DMA channels. The DMA channel mnemonics are as follows.

TC	Terminal count
SAR	Source address register
SARH	Source address register high
DAR	Destination address register
DARH	Destination address register high

Figure 20. DMA Mode Registers (DMAM)

MD_2	MD_1	MD_0	W	EDMA	TDMA	0	0
7							0

MD_2-MD_0	Transfer Mode
000	Single-step (memory to memory)
001	Demand release (I/O to memory)
010	Demand release (memory to I/O)
011	Reserved
100	Burst (memory to memory)
101	Single-transfer (I/O to memory)
110	Single-transfer (memory to I/O)
111	Reserved

W	Transfer Method
0	Byte transfer
1	Word transfer

EDMA	TDMA	Transfer Condition
0	0	Disabled
1	0	Maintain condition
1	1	Start DMA transfer

Figure 21. DMA Address Control Registers (DMAC)

0	0	PD_1	PD_0	0	0	PS_1	PS_0
7							0

PD_1-PD_0	Destination Address Offset
00	No modification
01	Increment
10	Decrement
11	No modification

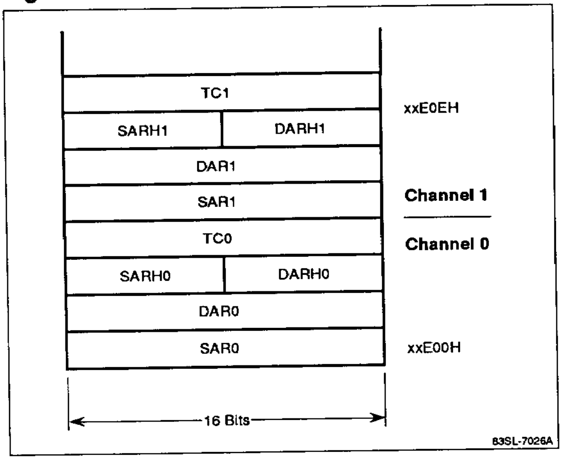
PS_1-PS_0	Source Address Offset
00	No modification
01	Increment
10	Decrement
11	No modification

These control registers (figure 22) are mapped into the same area of register bank 0 as the macroservice control block registers. These macroservice channels should not be used when the DMA controller is active.

The DMA controller generates the physical source and destination addresses by offsetting Address High register 12 bits to the left and then adding the Address register. The source and destination address registers can be programmed to increment or decrement independently for DMA operation.

When the EDMA bit is set, the internal DMARQ flag is cleared. Therefore, subsequent requests are recognized only after the EDMA bit has been set.

Figure 22. DMA Channels



PARALLEL I/O PORTS

Ports P0, P1, P2.

The μPD70327 has three 8-bit parallel I/O ports: P0, P1, and P2. Associated registers are shown in figures 23, 24, and 25. Special-function register (SFR) locations can access these ports as memory. The port lines are individually programmable as inputs or outputs. Many of the port lines have dual functions as port or control lines.

Use the associated port mode control (PMC) and port mode (PM) registers to select the function for a given I/O line.

Figure 23. Port 0 Registers (PMC0, PM0)

$PMC0_7$	0	0	0	0	0	0	0
7			PMC0 Register				0

$PM0_7$	$PM0_6$	$PM0_5$	$PM0_4$	$PM0_3$	$PM0_2$	$PM0_1$	$PM0_0$
7			PM0 Register				0

		$PMC0_7 = 0$	
Port Pin	**$PMC0_7 = 1$**	**$PM0_n = 1$**	**$PM0_n = 0$**
$P0_7$	CLKOUT	Input port	Output port
$P0_6$	—	Input port	Output port
$P0_5$	—	Input port	Output port
$P0_4$	—	Input port	Output port
$P0_3$	—	Input port	Output port
$P0_2$	—	Input port	Output port
$P0_1$	—	Input port	Output port
$P0_0$	—	Input port	Output port

Figure 24. Port 1 Registers (PMC1, PM1)

$PMC1_7$	$PMC1_6$	$PMC1_5$	$PMC1_4$	$PMC1_3$	0	0	0
7			PMC1 Register				0

$PM1_7$	$PM1_6$	$PM1_5$	$PM1_4$	1	1	1	1
7			PM1 Register				0

		$PMC1_n = 0$	
Port Pin	**$PMC1_7 = 1$**	**$PM1_n = 1$**	**$PM1_n = 0$**
$P1_7$	READY input	Input port	Output port
$P1_6$	$\overline{SCKO}$ output	Input port	Output port
$P1_5$	TOUT output	Input port	Output port
$P1_4$	INT input	$\overline{POLL}$ input	Output port
$P1_3$	$\overline{INTAK}$ output	INTP2 input	—
$P1_2$	—	INTP1 input	—
$P1_1$	—	INTP0 input	—
$P1_0$	—	NMI input	—

Figure 25. Port 2 Registers (PMC2, PM2)

$PMC2_7$	$PMC2_6$	$PMC2_5$	$PMC2_4$	$PMC2_3$	$PMC2_2$	$PMC2_1$	$PMC2_0$
7			PMC2 Register				0

$PM2_7$	$PM2_6$	$PM2_5$	$PM2_4$	$PM2_3$	$PM2_3$	$PM2_2$	$PM2_1$	$PM2_0$
7				PM2 Register				0

		$PMC2_n = 0$	
Port Pin	**$PMC2_n = 1$**	**$PM2_n = 1$**	**$PM2_n = 0$**
$P2_7$	HLDRQ input	Input port	Output port
$P2_6$	$\overline{HLDAK}$ output	Input port	Output port
$P2_5$	$\overline{TC1}$ output	Input port	Output port
$P2_4$	$\overline{DMAAK1}$ output	Input port	Output port
$P2_3$	DMARQ1 input	Input port	Output port
$P2_2$	$\overline{TC0}$ output	Input port	Output port
$P2_1$	$\overline{DMAAK0}$ output	Input port	Output port
$P2_0$	DMARQ0 input	Input port	Output port

Port PT

The analog comparator port (PT) compares each input line to a reference voltage. The reference voltage can be programmed to the V_{TH} input x n/16, where n = 1 to 16. See figure 26.

Figure 26. Port T Mode Register (PMT)

0	0	0	0	PMT_3	PMT_2	PMT_1	PMT_0
7							0

Comparator Reference Voltage (V_{REF})	PMT_3	PMT_2	PMT_1	PMT_0
V_{TH} x 16/16	0	0	0	0
1/16	0	0	0	1
2/16	0	0	1	0
3/16	0	0	1	1
4/16	0	1	0	0
5/16	0	1	0	1
6/16	0	1	1	0
7/16	0	1	1	1
8/16	1	0	0	0
9/16	1	0	0	1
10/16	1	0	1	0
11/16	1	0	1	1
12/16	1	1	0	0
13/16	1	1	0	1
14/16	1	1	1	0
15/16	1	1	1	1

PROGRAMMABLE WAIT STATES

You can generate wait states internally to further reduce the necessity for external hardware. Insertion of these wait states allows direct interface to devices whose access times cannot meet the CPU read/write timing requirements.

When using this function, the entire 1 megabyte of memory address space is divided into 128K-blocks. Each block can be programmed for zero, one, or two wait states, or two plus those added by the external READY signal. The top two blocks are programmed together as one unit.

The appropriate bits in the wait control word (WTC) control wait-state generation. Programming the upper two bits in the wait control word sets the wait-state conditions for the entire I/O address space. Figure 27 shows the memory map for programmable wait-state generation.

Figure 28 diagrams the wait control word. Note that READY pin control is enabled only when two internally generated wait states are selected by the "11" option.

Figure 27. Programmable Wait State Generation

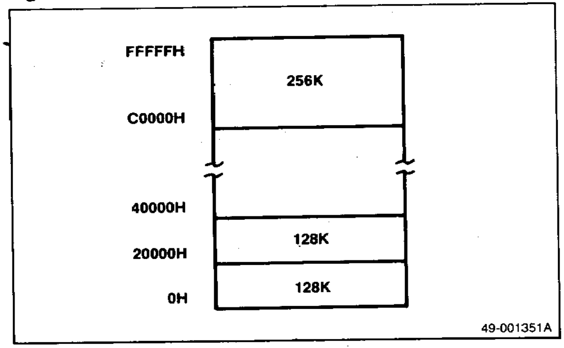


Figure 28. Wait Control Word (WTC)

IO1	IO0	Block 61	Block 60	Block 51	Block 50	Block 41	Block 40
7			Wait Control, High				0

Block 31	Block 30	Block 21	Block 20	Block 11	Block 10	Block 01	Block 00
7			Wait Control, Low				0

Wait States	Block n1	Block n0
0	0	0
1	0	1
2	1	0
2 or more (external control via READYpin)	1	1

n = 0 thru 6

STANDBY MODES

The two low-power standby modes are HALT and STOP. Both modes are entered under software control.

HALT Mode

In HALT mode, the CPU is inactive and thus the chip consumes much less power than when fully operational. The external oscillator remains functional and all internal peripherals are active. Internal status and output port line conditions are maintained. Any unmasked interrupt can release this mode. In the EI state, interrupts are processed subsequently in vector mode. In the DI state, program execution is restarted with the instruction following the HALT instruction.

STOP Mode

The STOP mode allows the largest power reduction while maintaining internal RAM. The oscillator is stopped, halting the CPU as well as all internal peripherals. Internal status is maintained. Only a reset or NMI can release this mode.

A standby flag in the SFR area is reset by rises in the supply voltage. Its status is maintained during normal operation and standby. The STBC register (figure 29) is not initialized by $\overline{\text{RESET}}$. Use the standby flag to determine whether program execution is returning from standby or from a cold start by setting this flag before entering STOP mode.

Figure 29. Standby Register (STBC)

0	0	0	0	0	0	0	SBF
7			Address xxFE0H				0

SBF	Standby Flag
0	No changes in V_{DD} (standby)
1	Rising edge on V_{DD} (cold start)

SPECIAL-FUNCTION REGISTERS

Table 3 shows the special function register mnemonic, type, address, reset value, and function. The 8 high-order bits of each address (xx) are specified by the IDB register.

SFR area addresses not listed in table 3 are reserved. If read, the contents of these addresses are undefined, and any write operation will be meaningless.

Table 3. Special Function Registers

Name	Byte/Word	Address	Reset Value (Note 2)	R/W (Note 1)	Function
P0	B	xxF00H		R/W	Port 0
PM0	B	xxF01H	FFH	W	Port mode control 0
P1	B	xxF09H	00H		R/W Port 1
PM1	B	xxF09H	FFH	W	Port mode 1
PMC1	B	xxF0AH	00H	W	Port mode control 1
P2	B	xxF10H		R/W	Port 2
PM2	B	xxF11H	FFH	W	Port mode 2
PMC2	B	xxF12H	00H	W	Port mode control 2
PT	B	xxF38H		R	Port T
PMT	B	xxF3BH	00H	R/W	Port mode T
INTM	B	xxF40H	00H	R/W	Interrupt mode
EMS0	B	xxF44H		R/W	External interrupt macro service 0
EMS1	B	xxF45H		R/W	External interrupt macro service 1
EMS2	B	xxF46H		R/W	External interrupt macro service 2
EXIC0	B	xxF4CH	47H	R/W	External interrupt control 0
EXIC1	B	xxF4DH	47H	R/W	External interrupt contol 1
EXIC2	B	xxF4EH	47H	R/W	External interrupt control 2
RXB0	B	xxF60H		R	Receive buffer 0
TXB0	B	xxF62H		W	Transfer buffer 0
SRMS0	B	xxF65H		R/W	Serial receive macro service 0
STMS1	B	xxF66H		R/W	Serial transmit macro service 1
SCM0	B	xxF68H	00H	R/W	Serial communication mode 0
SCC0	B	xxF69H	00H	R/W	Serial communication control 0
BRG0	B	xxF6AH	00H	R/W	Baud rate generator 0
SCE0	B	xxF6BH	00H	R	Serial communication error 0
SEIC0	B	xxF6CH	47H	R/W	Serial error interrupt control 0

Table 3. Special Function Registers (cont)

Name	Byte/Word	Address	Reset Value (Note 2)	R/W (Note 1)	Function
SRIC0	B	xxF6DH	47H	R/W	Serial receive interrupt control 0
STIC0	B	xxF6EH	47H	R/W	Serial transmit interrupt control 0
RXB1	B	xxF70H		R	Receive buffer 1
TXB1	B	xxF72H		W	Transmit buffer 1
SRMS1	B	xxF75H		R/W	Serial receive macro service 1
STMS1	B	xxF76H		R/W	Serial transmit macro service 1
SCM1	B	xxF78H	00H	R/W	Serial communication mode 1
SCC1	B	xxF79H	00H	R/W	Serial communication control 1
BRG1	B	xxF7AH	00H	R/W	Baud rate generator register 1
SCE1	B	xxF7BH	00H	R	Serial communication error 1
SELIC1	B	xxF7CH	47H	R/W	Serial error interrupt control 1
SRIC1	B	xxF7DH	47H	R/W	Serial receive interrupt control 1
STIC1	B	xxF7EH	47H	R/W	Serial transmit interrupt control 1
TM0	W	xxF80H		R/W	Timer register 0
TM0L	B	XXF80H		R/W	Timer register 0 low
TM0H	B	xxF81H		R/W	Timer register 0 high
MD0	W	xxF82H		R/W	Modulo register 0
MD0L	B	xxF82H		R/W	Modulo register 0 low
MD0H	B	xxF83H		R/W	Modulo register 0 high
TM1	W	xxF88H		R/W	Timer register 1
TM1L	B	xxF88H		R/W	Timer register 1 low
TM1H	B	xxF89H		R/W	Timer register 1 high
MD1	W	xxF8AH		R/W	Modulo register 1
MD1L	B	xxF8AH		R/W	Modulo register 1 low
MD1H	B	xxF98BH		R/W	Modulo register 1 high
TMC0	B	xF90H	00H	R/W	Timer control 0
TMC1	B	xxF91H	00H	R/W	Timer control 1
TMMS0	B	xxF94H		R/W	Timer macro service 0
TMMS1	B	xxF95H		R/W	Timer macro service 1
TMMS2	B	xxF96H		R/W	Timer macro service 2
TMIC0	B	xxF9CH	47H	R/W	Timer interrupt control 0
TMIC1	B	xxF9DH	47H	R/W	Timer interrupt control 1
TMIC2	B	xxF9EH	47H	R/W	Timer interrupt control 2
DMAC0	B	xxFA0H		R/W	DMA control 0
DMAM0	B	xxFA1H		R/W	DMA mode 0
DMAC1	B	xxFA2H		R/W	DMA control 1
DMAM1	B	xxFA3H	00H	R/W	DMA mode 1
DIC0	B	xxFACH	47H	R/W	DMA interrupt control 0
DIC1	B	xxFADH	47H	R/W	DMA interrupt control 1
STBC	B	xxFE0H		R/W	Standby control
RFM	B	xxFE1H	FCH	R/W	Refresh mode

Table 3. Special Function Registers (cont)

Name	Byte/Word	Address	Reset Value (Note 2)	R/W (Note 1)	Function
WTC	W	xxFE8H	FFH	R/W	Wait control
WTCL	B	xxFE8H	FFH	R/W	Wait control low
WTCH	B	xxFE9H	FFH	R/W	Wait control high
FLAG	B	xxFEACH	00H	R/W	Flag register
PRC	B	xxFEBH	4EH	R/W	Processor control
TBIC	B	xxFECH	47H	R/W	Time base IRC register
ISPR	B	xxFFCH		R	In service priority register
IDB	B	xxFFFH, FFFFFH		R/W	Internal data area base

Notes:

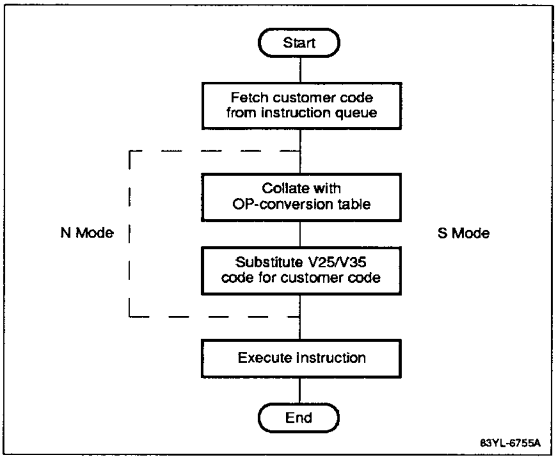
(1) R/W indicates whether register is available for read/write operations.

(2) Reset values not specified are undefined.

SECURITY MODE OPERATION

The security mode of the μPD70327 is designed to protect proprietary user software algorithms by encoding the user's programs resident in external EPROM or ROM memory. The process encodes only the first byte of each opcode via a linear translation table. The decoding process is performed in real time within the μPD70327 and thus does not impact system performance. The flow chart of the conversion process is shown in figure 30.

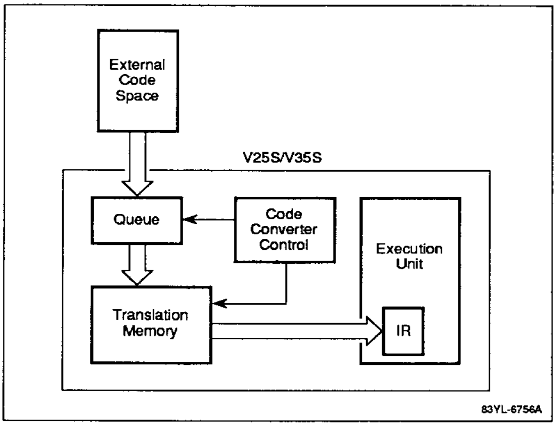
Figure 30. Opcode Translation Flowchart


Start
Fetch customer code from instruction queue
Collate with OP-conversion table
Substitute V25/V35 code for customer code
N Mode
S Mode
Execute instruction
End
83YL-6755A

The translation table is user-defined and is inserted into each μPD70327 mask at the factory. The μPD70327 can be dynamically switched from secure mode to normal mode, thus providing an additional measure of security as well as compatibility with existing ROM versions of V25 software. Note, however, that the V25 Software Guard does not support internal ROM.

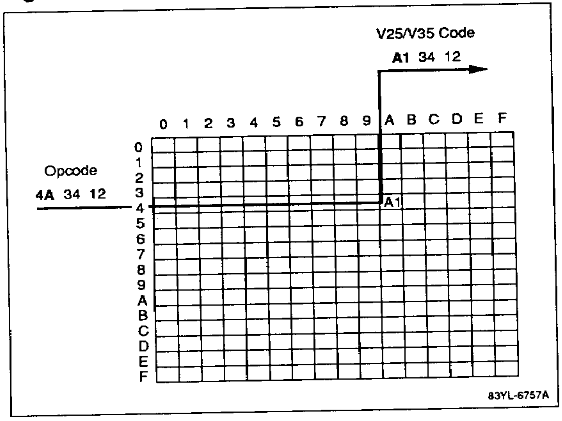
The opcode translator is effectively a look-up table that is inserted between the instruction prefetch queue and the instruction register of the μPD70327. A conceptual diagram of this is in figure 31.

Figure 31. Code Converter Functional Diagram



The code converter uses the encrypted opcode from the prefetch queue as an address, and provides the correct V25 opcode as data to the instruction register. An example of this is shown in figure 32. Again, only the first byte of each opcode is decoded, and subsequent bytes are passed directly from the prefetch unit to the execution unit.

Figure 32. Opcode Converter Translation Table

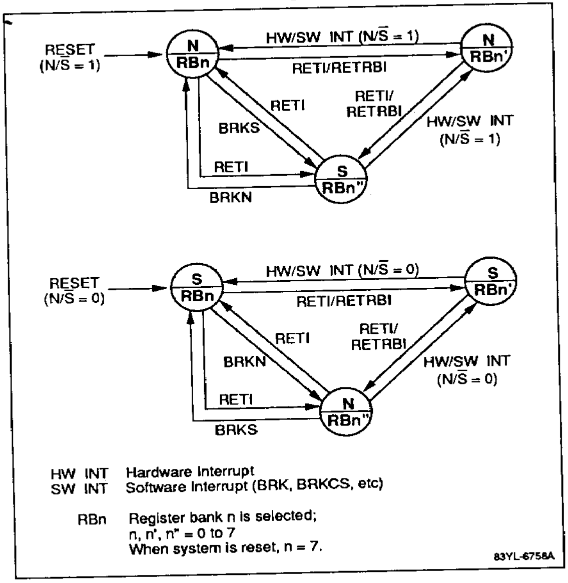

V25/V35 Code
A1 34 12
0 1 2 3 4 5 6 7 8 9 A B C D E F
Opcode
4A 34 12
0 1 2 3 4 5 6 7 8 9 A B C D E F
A1
83YL-6757A

Mode Switching

The transition from normal V25 instruction execution to secure instruction decoding and execution can be performed in either hardware or software. The hardware trigger source is provided by the N/$\overline{S}$ pin of the μPD70327. This pin is listed as an internal connection pin on standard V25 systems, and as such, should be pulled up to V_{DD} through a resistor. Thus a μPD70327 used in a standard V25 design will execute in normal mode identically to the standard V25.

The state of the N/$\overline{S}$ pin is read by the processor at system reset and determines the operational mode of the device at that point. Regardless of the state of this pin, the μPD70327 will begin program execution using register bank 7 as the default register set. (See figure 33.) If the processor samples the N/$\overline{S}$ pin in the low state, the first opcode fetched from the reset address will be decoded using the on-chip translation table. The N/$\overline{S}$ pin has an internal pull-up resistor that will set the device to normal mode operation with no external connections. The N/$\overline{S}$ pin should be set in hardware to a fixed logic state.

Figure 33. Operational Mode State Transition Diagram



HW INT Hardware Interrupt
SW INT Software Interrupt (BRK, BRKCS, etc)

RBn Register bank n is selected;
n, n', n" = 0 to 7
When system is reset, n = 7.

Software control of the operating mode is performed by the BRKS (Break for Secure Operation) and the BRKN (Break for Normal Operation) instructions. These opcodes are undefined codes on the standard V25, and should not be ported to standard V25 processor environments. These instructions are detailed in the instruction set section.

The operational state of the μPD70327 is specified by bit 15 (MD) of the Program Status Word (PSW). The remainder of the PSW is identical to that of the standard V25. Since portability of V25 and V25 Software Guard systems is sometimes desired, bit 15 of the PSW should always be written as a logical 1 in standard V25 systems. As with the V25/V35, the upper 4 bits of the PSW cannot be updated by POP; the upper 8 bits of the PSW cannot be updated by MOV. Refer to the PSW diagram shown previously in the "Register Set" section of this data sheet.

As can be seen in figure 33, the N/$\overline{S}$ pin is also sampled upon receipt of an interrupt (either software or hardware). The state of the N/$\overline{S}$ pin will determine the execution mode of the interrupt service routine. The mode of the interrupted routine will be restored by the RETI or RETRBI that terminates the interrupt handler. Software mode changes (via the BRKS and BRKN instructions described later) will always change the state of the MD bit of the PSW. The MD bit for these software interrupts is restored by the RETI instruction at the end of the mode switch software interrupt.

Operation Timing

Operational execution of the standard V25 and that of the V25 Software Guard are identical regardless of the operational mode selected for the V25 Software Guard. However, since the μPD70327 is a ROMless device, all memory cycles are nominally two system clock periods long. (This is in contrast to the one clock cycle ROM code fetch of the μPD70322.) Due to its ROMless nature, the μPD70327 does not support the $\overline{\text{EA}}$ pin of the standard V25, and this pin (labeled IC) should be fixed to a logical high level in the hardware.

ELECTRICAL SPECIFICATIONS

The electrical specifications of the V25 Software Guard and the standard V25 are the same. Refer to the μPD70320/322 (V25) Data Sheet.

INSTRUCTION SET

The instruction set of the V25 Software Guard and the standard V25 are the same except for the addition of two mode change instructions for the V25 Software Guard (BRKS and BRKN) described below.

BRKS Instruction

The BRKS instruction switches operation to security (S) mode and generates a vectored interrupt. In S mode, the fetched operation code is executed after conversion in accordance with the built-in translation table.

The RETI instruction is used to return to the operating mode prior to execution of the BRKS instruction.

BRKN Instruction

The BRKN instruction switches operation to normal (N) mode and generates a vectored interrupt. In N mode, the fetched instruction is executed as a μPD70320/70322 (V25) operation code.

The RETI instruction is used to return to the operating mode prior to execution of the BRKN instruction.

Opcodes

Clock counts and opcodes applicable to the added mode change instructions are in tables 4 and 5.

Table 4. Instruction Clock Counts

Mnemonic	Operand	*Clocks
BRKS	imm8 (≠3)	56 + 10T [44 + 10T]
BRKN	imm8 (≠3)	56 + 10T [44 + 10T]

* Clock counts are specified for RAM enabled and [RAM disabled].

Table 5. Mode Change Instructions

			Operation Code									
Mnemonic	Operand	Operation	7	6	5	4	3	2	1	0	No. of Bytes	Flags
BRKS	imm8 (≠3)	(SP – 1, SP – 2) ← PSW, (SP – 3, SP – 4) ← PS, (SP – 5, SP – 6) ← PC, SP ← SP – 6 IE ← 0, BRK ← 0, MD ← 0 PC ← (n x 4 + 1, n x 4) PS ← (n x 4 + 3, n x 4 + 2) n = imm8	1	1	1	1	0	0	0	1	2	Not applicable
BRKN	imm8 (≠3)	(SP – 1, SP – 2) ← PSW, (SP – 3, SP – 4) ← PS, (SP – 5, SP – 6) ← PC, SP ← SP – 6 IE ← 0, BRK ← 0, MD ← 1 PC ← (n x 4 + 1, n x 4) PS ← (n x 4 + 3, n x 4 + 2) n = imm8	0	1	1	0	0	0	1	1	2	Not applicable