μPD77C20A, 7720A, 77P20
Digital Signal Processors

Description

The μPD77C20A, μPD7720A, and μPD77P20—three signal processing interface (SPI) chips that are functionally the same—are advanced architecture microcomputers optimized for signal processing algorithms. Their speed and flexibility allow these SPIs to efficiently implement signal processing functions in a wide range of environments and applications.

The 7720A SPI, a revision of the 7720, the original mask ROM chip, uses a third less power than the 7720.

The 77C20A is a CMOS pin-for-pin compatible version of the NMOS version, 7720A. This advanced architecture CMOS microcomputer has power requirements 80 percent less than the 7720A, making the 77C20A appropriate for portable applications and other designs requiring low power and low heat dissipation.

Minor differences between 7720A and 77C20A are described in the Instruction Timing section.

The 77P20 is an ultraviolet erasable and electrically programmable (EPROM) version of the 7720A. Program and data ROM, masked for the 7720A, are implemented in EPROM for the 77P20. The 77P20 is useful in prototype applications or in systems where product quantities are insufficient for masked ROM development.

Since the inception of 7720 and its companion EPROM version, 77P20, there have been several mask revisions to improve manufacturability and function. A 77P20 must always be used to verify the functions of a user's system before ROM code for 77C20A or 7720A is submitted, but certain early versions of 77P20 must not be used for final verification. Refer to the section on μPD77P20 for details.

Features

- Low-power CMOS: 24 mA typical current use (77C20A)
- Fast instruction execution: 240 ns with 8.333-MHz clock
- 16-bit data word
- Multioperation instructions for fast program execution: multiply, accumulate, move data, adjust memory pointers—all in one instruction cycle
- Modified Harvard architecture with three separate memory areas
  - Program ROM (512 x 23 bits)
  - Data ROM (510 x 13 bits)
  - Data RAM (128 x 16 bits)
- 16 x 16-bit multiplier; 31-bit product with every instruction
- Dual 16-bit accumulators
- External maskable interrupt
- Four-level stack for subroutines and/or interrupt
- Multiple I/O capabilities
  - Serial: 8- or 16-bit (480 ns/bit)
  - Parallel: 8- or 16-bit
  - DMA
- Compatible with most μPs, including:
  - μPD8080
  - μPD8085
  - μPD8086/88
  - μPD780 (Z80®)
- Single +5-volt power supply
- NMOS technology (7720A, 77P20)
- Extended temperature range

Applications

- Portable telecommunications equipment
- Digital filtering
- High-speed data modems
- Fast Fourier transforms (FFT)
- Speech synthesis and analysis
- Dual-tone multifrequency (DTMF) transmitters/receivers
- Equalizers
- Adaptive control
- Numerical processing

Performance Benchmarks

- Second-order digital filter (biquad): 2.21 μs
- Sin/cos of angles: 5.16 μs
- μ/A law to linear conversion: 0.49 μs
- FFT
  - 32-point complex: 0 7 ms
  - 64-point complex: 1.6 ms

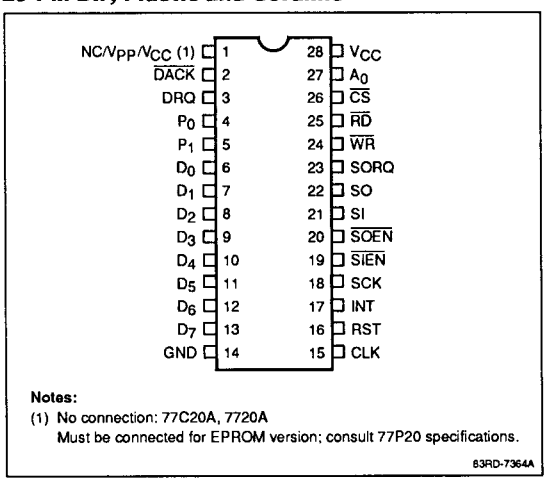
Z80 is a registered trademark of Zilog Corporation.

Ordering Information

Part Number	Package	Max Frequency of Operation	Normal Temperature Range
μPD77C20AC	28-pin plastic DIP	8.33 MHz	−40 to +85°C
ALK	28-pin PLCC		
AL	44-pin PLCC		
AGW	32-pin SOP		
μPD7720AC	28-pin plastic DIP	8.33 MHz	−10 to +70°C
μPD77P20D	28-pin cerdip	8.196 MHz	−10 to +70°C

Pin Configurations

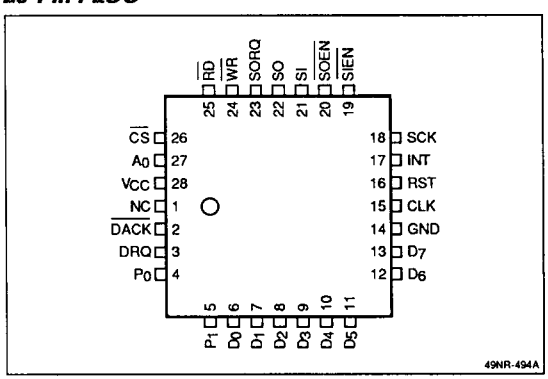
28-Pin DIP, Plastic and Ceramic



Notes:
(1) No connection: 77C20A, 7720A
Must be connected for EPROM version; consult 77P20 specifications.

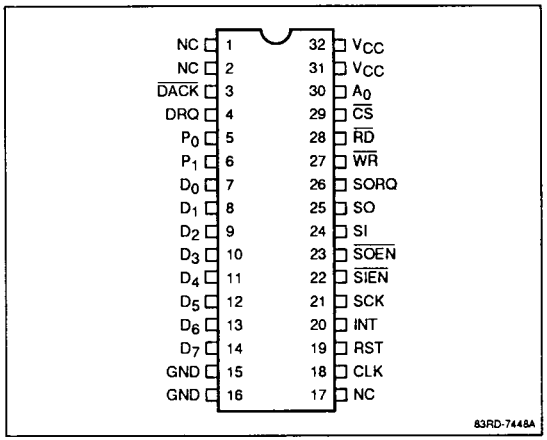
83RD-7364A

28-Pin PLCC



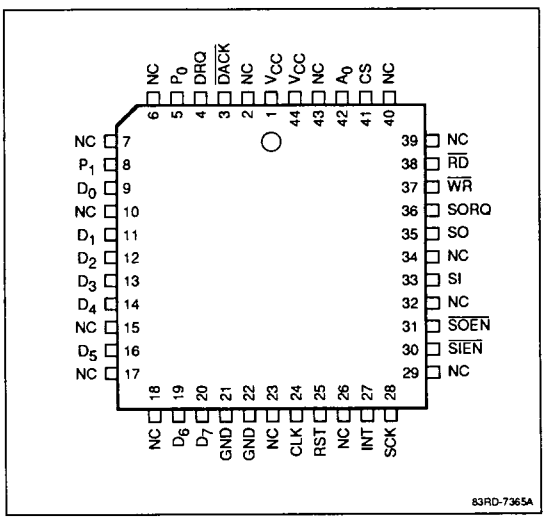
49NR-494A

32-Pin SOP



83RD-7448A

44-Pin PLCC



83RD-7365A

Pin Identification

Symbol	Function
A_0	Status/data register select input
CLK	Single-phase master clock input
$\overline{CS}$	Chip select input
D_0-D_7	Three-state I/O data bus
$\overline{DACK}$	DMA request acknowledge input
DRQ	DMA request output
INT	Interrupt input
P_0, P_1	General-purpose output control lines
$\overline{RD}$	Read control signal input
RST	Reset input
SCK	Serial data I/O clock input
SI	Serial data input
$\overline{SIEN}$	Serial input enable input
SO	Three-state serial data output
$\overline{SOEN}$	Serial output enable input
SORQ	Serial data output request
$\overline{WR}$	Write control signal input
GND	Ground
V_{CC}	+5 V power supply
NC/V_{PP}/V_{CC}	No connection (77C20A, 7720A)/ programming voltage (77P20)

PIN FUNCTIONS

A_0 (Status/Data Register Select)

This input selects data register for read/write (low) or status register for read (high).

CLK

This is the single-phase master clock input.

$\overline{CS}$ (Chip Select)

This input enables data transfer through the data port with $\overline{RD}$ or $\overline{WR}$.

D_0-D_7 (Data Bus)

This three-state I/O data bus transfers data between the data register or status register and the external data bus.

$\overline{DACK}$ (DMA Request Acknowledge)

This input indicates to the SPI that the data bus is ready for a DMA transfer (DACK = CS and A_0 = 0).

DRQ (DMA Request)

This output signals that the SPI is requesting a data transfer on the data bus.

INT (Interrupt)

A low-to-high transition on this pin executes a call instruction to location 100H if interrupts were previously enabled.

P_0, P_1

These pins are general-purpose output control lines.

$\overline{RD}$ (Read Control Signal)

This input latches data from the data or status register to the data port where it is read by an external device.

RST (Reset)

This input initializes the SPI internal logic and sets the PC to 0.

SCK (Serial Data I/O Clock)

When this input is high, a serial data bit is transferred.

SI (Serial Data Input)

This pin inputs 8- or 16-bit serial data words from an external device such as an A/D converter.

$\overline{SIEN}$ (Serial Input Enable)

This input enables the shift clock to the serial input register.

SO (Serial Data Output)

This three-state port outputs 8- or 16-bit data words to an external device such as a D/A converter.

$\overline{SOEN}$ (Serial Output Enable)

This input enables the shift clock to the serial output register.

SORQ (Serial Data Output Request)

This output specifies to an external device that the serial data register has been loaded and is ready for output. SORQ is reset when the entire 8- or 16-bit word has been transferred.

$\overline{WR}$ (Write Control Signal)

This input writes data from the data port into the data register.

GND

This is the connection to ground.

V_{CC} (Power Supply)

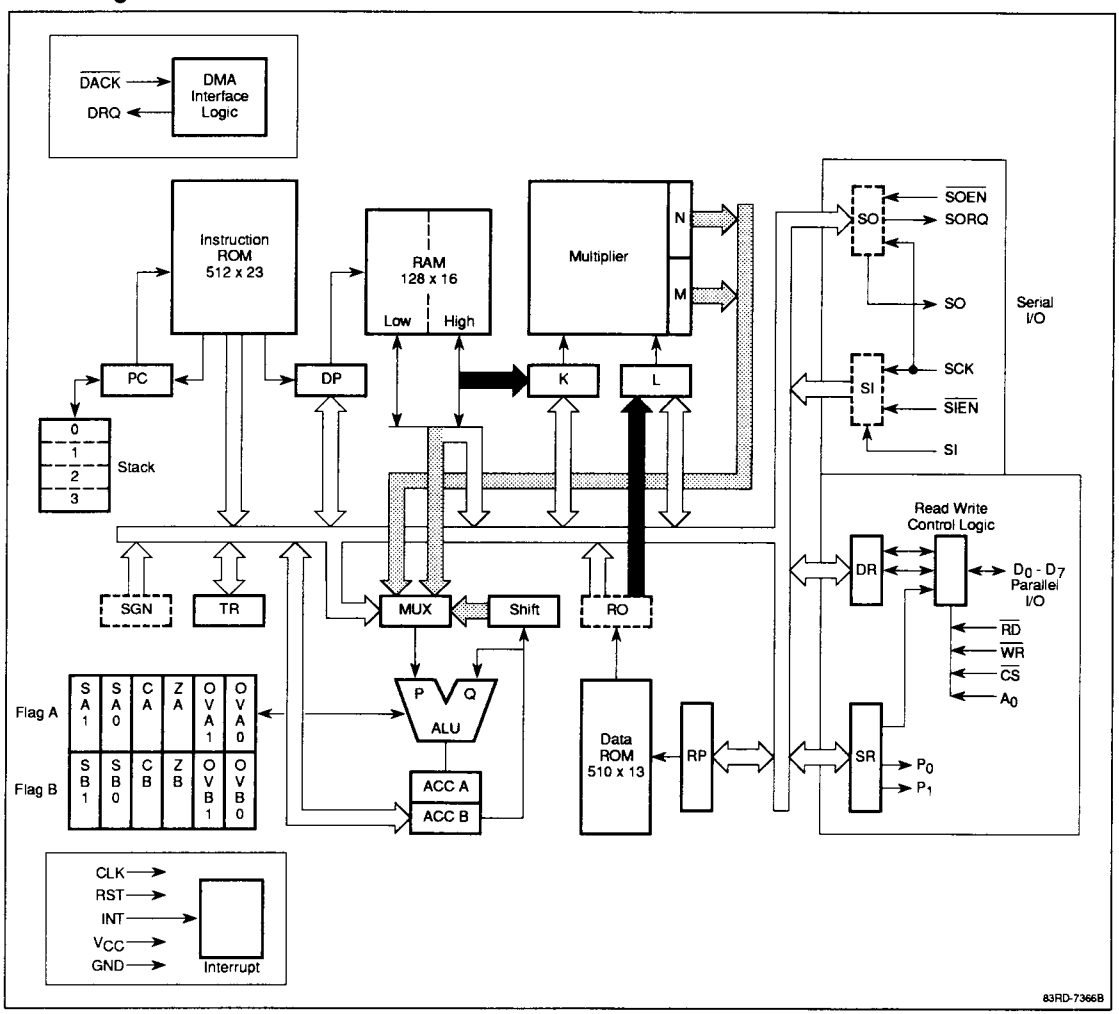
This pin is the + 5-volt power supply.

NC/V_{PP}/V_{CC}

This pin is not internally connected in the 77C20A and 7720A. In the 77P20, this pin inputs the programming voltage (V_{PP}) when the part is being programmed.

This pin must be connected to V_{CC} for proper 77P20 operation. Consult the section on the μPD77P20 for details.

Block Diagram



FUNCTIONAL DESCRIPTION

The primary bus (unshaded in the block diagram) makes a data path between all of the registers (including I/O), memory, and the processing sections. This bus is referred to as the IDB (internal data bus). The multiplier input registers K and L can be loaded not only from the IDB but alternatively via buses (darkened in the block diagram) directly from RAM to the K register and directly from data ROM to the L register. Output from the multiplier in the M and N registers is typically added via buses (shaded in the block diagram) to either accumulator A or B as part of a multioperation instruction.

The SPI is a complete 16-bit microcomputer on a single chip. ROM space provides program and coefficient storage; the on-chip RAM may be used for temporary data, coefficients, and results. A 16-bit arithmetic/logic unit (ALU) and a separate 16 x 16-bit, fully-parallel multiplier provide computational power. This combination allows the implementation of a "sum of products" operation in a single 240-ns instruction cycle. In addition, each arithmetic instruction allows a number of data movement operations to further increase throughput.

Two serial I/O ports interface to codecs and other serial-oriented devices; a parallel port provides both data and status information to conventional microprocessors. Handshaking signals, including DMA controls, allow the SPI to act as a sophisticated programmable peripheral as well as a standalone microcomputer.

MEMORY

Memory is divided into three types: instruction ROM, data ROM, and data RAM. The 512 x 23-bit words of instruction ROM are addressed by a 9-bit program counter that can be modified by an external reset, interrupt, call, jump, or return instruction.

The data ROM is organized in 510 x 13-bit words that are addressed through a 9-bit ROM pointer (RP register). The RP may be modified simultaneously with arithmetic instructions so that the next value is available for the next instruction. The data ROM is ideal for storing the necessary coefficients, conversion tables, and other constants for your processing needs.

Do not use data ROM locations 0 and 1 in the 77C20A or 7720A. These locations are reserved for storage of test pattern data. (When submitting code, set these locations to 0). Note that 77P20 allows use of these locations, but using them is not advised.

The data RAM is 128 x 16-bit words and is addressed through a 7-bit data pointer (DP register). The DP has extensive addressing features that operate simultaneously with arithmetic instructions, eliminating additional time for addressing or address modification.

ARITHMETIC CAPABILITIES

One of the unique features of the SPI's architecture is its arithmetic facilities. With a separate multiplier, ALU, and multiple internal data paths, the SPI is capable of carrying out a multiply, an add, or other arithmetic operation, and a data move between internal registers in a single instruction cycle.

3a

ALU

The ALU is a 16-bit two's complement unit capable of executing 16 distinct operations on virtually any of the SPI's internal registers, thus giving the SPI both speed and versatility for efficient data management.

Accumulators (ACCA/ACCB)

Associated with the ALU are two 16-bit accumulators, each with its own set of flags, which are updated at the end of each arithmetic instruction (except NOP). Table 1 shows the ACC A/B flag registers. In addition to zero result, sign, carry, and overflow flags, the SPI incorporates auxiliary overflow and sign flags (SA1, SB1, OVA1, OVB1). These flags enable the detection of an overflow condition and maintain the correct sign after as many as three successive additions or subtractions.

Table 1. ACC A/B Flag Registers

Flag A	SA1	SA0	CA	ZA	OVA1	OVA0
Flag B	SB1	SB0	CB	ZB	OVB1	OVB0

Sign Register (SGN)

When OVA1 is set, the SA1 bit will hold the corrected sign of the overflow. The SGN register will use SA1 to automatically generate saturation constants 7FFFH(+) or 8000H(–) to permit efficient limiting of a calculated value. The SGN register is not affected by arithmetic operations on accumulator B, but flags SB1, SB0, CB, ZB, OVB1, and OVB0 are affected.

Multiplier

Thirty-one bit results are developed by a 16 x 16-bit two's complement multiplier in 240 ns. The result is automatically latched to two 16-bit registers, M and N, at the end of each instruction cycle. The sign bit and 15 higher bits are in M and the 15 lower bits are in N; the

LSB in N is zero. A new product is available for use after every instruction cycle, providing significant advantages in maximizing processing speed for real-time signal processing.

Stack

The SPI contains a four-level program stack for efficient program usage and interrupt handling.

Interrupt

The SPI supports a single-level interrupt. Upon sensing a high level on the INT pin, a subroutine call to location 100H is executed. The EI bit of the status register automatically resets to 0, disabling the interrupt facility until it is reenabled under program control.

INPUT/OUTPUT

General

The SPI has three communication ports as shown in figure 1: two serial and one 8-bit parallel, each with its own control lines for interface handshaking. Parallel port operation is software-configurable to be in either polled mode or DMA mode. A general-purpose, two-line output port rounds out a full complement of interface capability.

Serial I/O

The two shift registers (SI, SO) are software-configurable to single- or double-byte transfers. The shift registers are externally clocked (SCK) to provide a simple interface between the SPI and serial peripherals such as A/D and D/A converters, codecs, or other SPIs. Figure 2 shows serial I/O timing

Figure 1. SPI Communication Ports

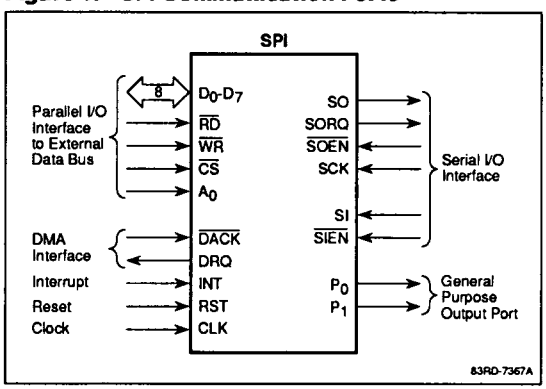
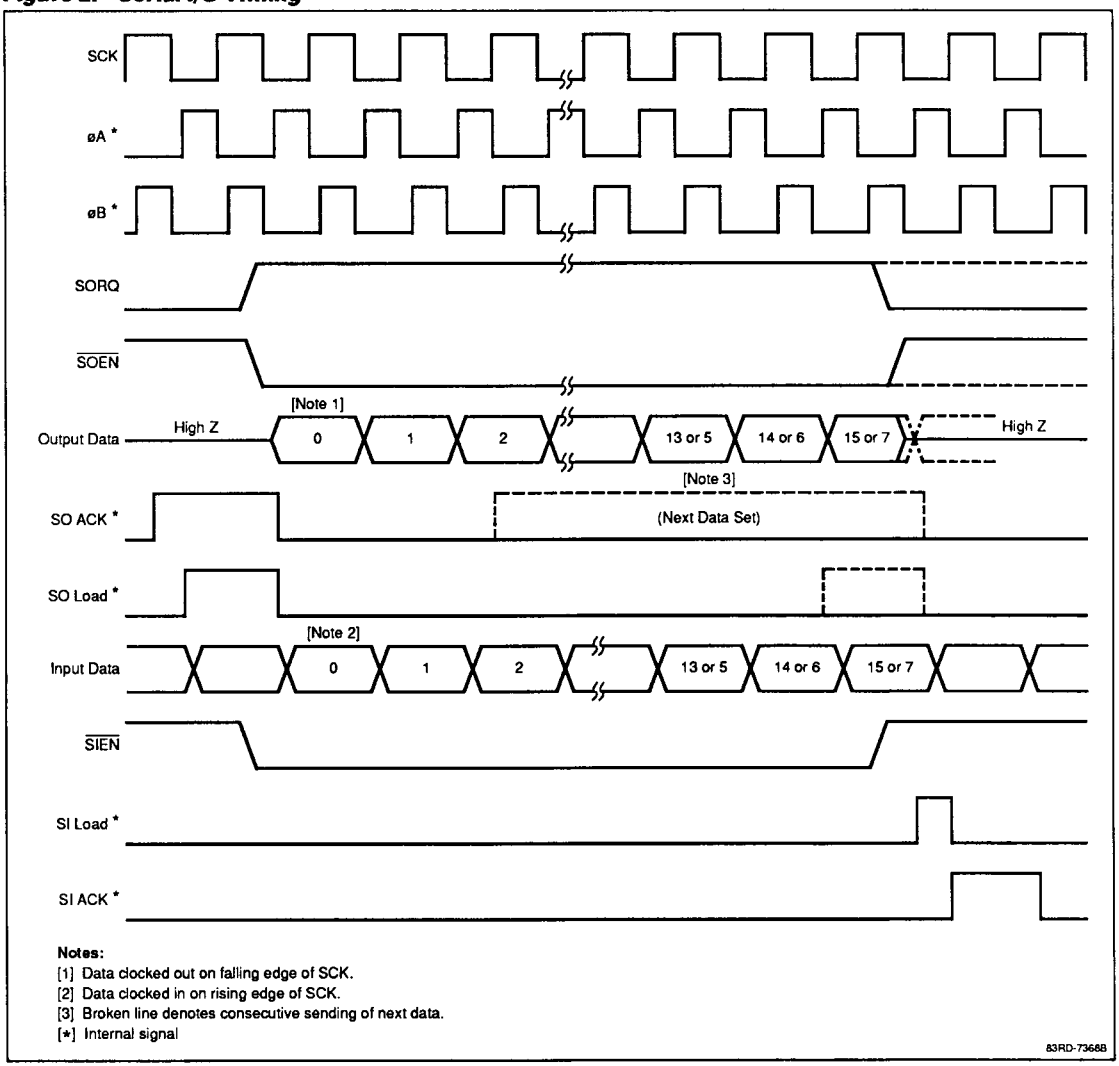


Figure 2. Serial I/O Timing



Notes:
[1] Data clocked out on falling edge of SCK.
[2] Data clocked in on rising edge of SCK.
[3] Broken line denotes consecutive sending of next data.
[*] Internal signal

83RD-73688

Parallel I/O

The 8-bit parallel I/O port may be used for transferring data or reading the SPI's status as shown in table 2. Data transfer is handled through a 16-bit data register (DR) that is software-configurable for double- or single-byte data transfers. The port is ideally suited for operating with 8080, 8085, and 8086 processor buses and may be used with other processors and computer systems.

DMA Mode Option

Parallel data transfers may be controlled (optionally) via DMA control lines DRQ and $\overline{DACK}$. DMA mode allows high-speed transfers and reduced processor overhead. When in DMA mode, $\overline{DACK}$ input resets DRQ output when data transfer is completed. $\overline{DACK}$ does not affect any status register bit or flag bit.

Table 2. Parallel R/W Operation

$\overline{CS}$	A_0	$\overline{WR}$	$\overline{RD}$	Operation
1	X	X	X	No effect on internal operation; D_0-D_7 are at high impedance levels.
X	X	1	1	
0	0	0	1	Data from D_0-D_7 is latched to DR (Note 1)
0	0	1	0	Contents of DR are output to D_0-D_7 (Note 1)
0	1	0	1	Illegal (SR is read only)
0	1	1	0	Eight MSBs of SR are output to D_0-D_7
0	X	0	0	Illegal (may not read and write simultaneously)

Notes:

(1) Eight MSBs or 8 LSBs of data register (DR) are used, depending on DR status bit (DRS). The condition of DACK = 0 is equivalent to A_0 = CS = 0.

Status Register

The status register (figure 3) is a 16-bit register in which the eight most significant bits may be read by the system's microprocessor for the latest parallel data I/O status. The RQM and DRS bits can only be affected by parallel data moves. The other bits can be written to (or read) by the SPI's load immediate (LDI) or move (MOV) instructions. The EI bit is automatically reset when an interrupt is serviced.

Figure 3. Status Register (SR)

15	14	13	12	11	10	9	8
RQM	USF1	USF0	DRS	DMA	DRC	SOC	SIC

MSB

7	6	5	4	3	2	1	0
EI	0	0	0	0	0	P1	P0

LSB

Table 3. Status Register Flags

Flag	Description
RQM (Request for Master)	A read or write from DR to IDB sets RQM = 1. An external read (write) resets RQM = 0.
USF1 and USF0 (User Flags 1 and 0)	General-purpose flags which may be read by an external processor for user-defined signaling
DRS (DR Status)	For 16-bit DR transfers (DRC = 0). DRS = 1 after first 8 bits have been transferred. DRS = 0 after all 16 bits have been transferred.
DMA (DMA Enable)	DMA = 0 (Non-DMA transfer mode) DMA = 1 (DMA transfer mode)
DRC (DR control)	DRC = 0 (16-bit mode) DRC = 1 (8-bit mode)
SOC (SO Control)	SOC = 0 (16-bit mode) SOC = 1 (8-bit mode)
SIC (SI Control)	SIC = 0 (16-bit mode) SIC = 1 (8-bit mode)
EI (Enable Interrupt)	EI = 0 (interrupts disabled) EI = 1 (interrupts enabled)
P0, P1 (Ports 0 and 1)	P0 and P1 directly control the state of output pins P_0 and P_1

INSTRUCTIONS

The SPI has three types of instructions: Load Immediate, Branch, and the multifunction OP instruction. Each type takes the form of a 23-bit word and executes in 240 ns.

Instruction Timing

To control the execution of instructions, the external 8-MHz clock is divided into four phases for internal execution. The various elements of the 23-bit instruction word are executed in a set order. Multiplication automatically begins first. Also, data moves from source to destination before other elements of the instruction. Data being moved on the internal data bus (IDB) is available for use in ALU operations (if P-select field of the instruction specifies IDB). However, if the accumulator specified in the ASL field is also specified as the destination of the data move, the ALU operation becomes an NOP as the data move supersedes the ALU operation.

Pointer modifications occur at the end of the instruction cycle after their values have been used for data moves. The result of multiplication is available at the end of the instruction cycle for possible use in the next instruction. If a return is specified as part of an OP instruction, it is executed last.

An assembly language OP instruction may consist of what looks like one to six lines of assembly code, but all of these lines are assembled together into one 23-bit instruction word. Therefore, the order of the six lines makes no difference in the order of execution described above. However, for understanding the SPI's operation and to eliminate confusion, write assembly code in the order described; that is, data move, ALU operations, data pointer modifications, and then return.

Minor differences exist between 7720A and 77C20A in internal instruction execution timing. Using normal programming instruction statements, the differences will not appear. However, an instruction such as the following will yield a difference between NMOS and CMOS operation.

OP MOV @MEM,B XOR ACCB, RAM

The instruction, which is acceptable using the NEC assembler (AS77201), has an inherent conflict in that data is simultaneously being moved into memory and fetched in one instruction. ALU instructions involving either ACCA or ACCB should not be used. In summary, observe the following rules.

(1) DST should not be @MEM when PSEL is RAM.

(2) When SRC is NON, DST must be @NON.

(3) A should not be used as both DST and ASL.

(4) B should not be used as both DST AND ASL.

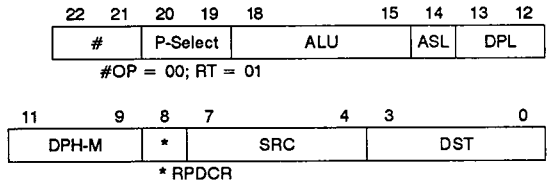
OP/RT Instruction Field Specification

Figure 4 illustrates the OP/RT instruction field specification. There are two instructions of this type, both of which are capable of executing all ALU functions listed in table 4. The ALU functions operate on the value specified by the P-select field (see table 5).

Besides the arithmetic functions, these instructions can also (1) modify the RAM data pointer DP, (2) modify the data ROM pointer RP, and (3) move data along the on-chip data bus from a source register to a destination register. The possible source and destination registers are listed in tables 6 and 7, respectively.

The difference in the two instructions of this type is that RT executes a subroutine or interrupt return at the end of the instruction cycle, but the OP does not. Tables 8, 9, 10, and 11 show the ASL, DPL, DPH, and RPDCR fields, respectively.

Figure 4. OP/RT Instruction Field



3a

Table 4. ALU Field

Mnemonic	D_{18}	D_{17}	D_{16}	D_{15}	ALU Function	SA1, SB1	SA0, SB0	CA, CB	ZA, ZB	OVA1, OVB1	OVA0, OVB0
NOP	0	0	0	0	No operation	—	—	—	—	—	—
OR	0	0	0	1	OR	x	Δ	0	Δ	0	0
AND	0	0	1	0	AND	x	Δ	0	Δ	0	0
XOR	0	0	1	1	Exclusive OR	x	Δ	0	Δ	0	0
SUB	0	1	0	0	Subtract	Δ	Δ	Δ	Δ	Δ	Δ
ADD	0	1	0	1	ADD	Δ	Δ	Δ	Δ	Δ	Δ
SBB	0	1	1	0	Subtract with borrow	Δ	Δ	Δ	Δ	Δ	Δ
ADC	0	1	1	1	Add with carry	Δ	Δ	Δ	Δ	Δ	Δ
DEC	1	0	0	0	Decrement ACC	Δ	Δ	Δ	Δ	Δ	Δ
INC	1	0	0	1	Increment ACC	Δ	Δ	Δ	Δ	Δ	Δ
CMP	1	0	1	0	Complement ACC (one's complement)	x	Δ	0	Δ	0	0
SHR1	1	0	1	1	1-Bit right shift	x	Δ	Δ	Δ	0	0
SHL1	1	1	0	0	1-Bit left shift	x	Δ	Δ	Δ	0	0
SHL2	1	1	0	1	2-Bit left shift	x	Δ	0	Δ	0	0
SHL4	1	1	1	0	4-Bit left shift	x	Δ	0	Δ	0	0
XCHG	1	1	1	1	8-Bit exchange	x	Δ	0	Δ	0	0

Δ May be affected, depending on the results. 0 Reset
— Previous status can be held. x Indefinite

Table 5. P-Select Field

Mnemonic	D_{20}	D_{19}	ALU Input
RAM	0	0	RAM
IDB	0	1	Internal Data Bus (Note 1)
M	1	0	M Register
N	1	1	N Register

Notes:

(1) Any value on the on-chip data bus. Value may be selected from any of the registers listed in table 6 source register selections.

Table 6. SRC Field

Mnemonic	D_7	D_6	D_5	D_4	Source Register
NON	0	0	0	0	No register
A	0	0	0	1	ACCA (Accumulator A)
B	0	0	1	0	ACCB (Accumulator B)
TR	0	0	1	1	TR temporary register
DP	0	1	0	0	DP data pointer
RP	0	1	0	1	RP ROM pointer
RO	0	1	1	0	RO ROM output data
SGN	0	1	1	1	SGN sign register
DR	1	0	0	0	DR data register
DRNF	1	0	0	1	DR no flag (Note 1)
SR	1	0	1	0	SR status register
SIM	1	0	1	1	SI serial in MSB (Note 2)
SIL	1	1	0	0	SI serial in LSB (Note 3)
K	1	1	0	1	K register
L	1	1	1	0	L register
MEM	1	1	1	1	RAM

Notes:

(1) DR to IDB, RQM not set. In DMA, DRQ not set.

(2) First bit in goes to MSB, last bit to LSB.

(3) First bit goes to LSB, last bit to MSB (bit reversed).

Table 7. DST Field

Mnemonic	D_3	D_2	D_1	D_0	Destination Register
@NON	0	0	0	0	No register
@A	0	0	0	1	ACCA (Accumulator A)
@B	0	0	1	0	ACCB (Accumulator B)
@TR	0	0	1	1	TR temporary register
@DP	0	1	0	0	DP data pointer
@RP	0	1	0	1	RP ROM pointer
@DR	0	1	1	0	DR data register
@SR	0	1	1	1	SR status register

Table 7. DST Field (cont)

Mnemonic	D_3	D_2	D_1	D_0	Destination Register
@SOL	1	0	0	0	SO serial out LSB (Note 1)
@SOM	1	0	0	1	SO serial out MSB (Note 2)
@K	1	0	1	0	K (Mult)
@KLR	1	0	1	1	IDB → K, ROM → L (Note 3)
@KLM	1	1	0	0	Hi RAM → K, IDB → L (Note 4)
@L	1	1	0	1	L (Mult)
@NON	1	1	1	0	No register
@MEM	1	1	1	1	RAM

Notes:

(1) LSB is first bit out.

(2) MSB is first bit out.

(3) Internal data bus to K, and ROM to L register.

(4) Contents of RAM address specified by $DP_6 = 1$, is placed in K register, IDB is placed in L (that is, 1, DP_5, DP_4 DP_3-DP_0).

Table 8. ASL Field

Mnemonic	D_{14}	ACC Selection
ACCA	0	ACCA
ACCB	1	ACCB

Table 9. DPL Field

Mnemonic	D_{13}	D_{12}	Low DP Modify (DP_3-DP_0)
DPNOP	0	0	No operation
DPINC	0	1	Increment DPL
DPDEC	1	0	Decrement DPL
DPCLR	1	1	Clear DPL

Table 10. DPH Field

Mnemonic	D_{11}	D_{10}	D_9	High DP Modify
M0	0	0	0	Exclusive OR of DPH (DP_6-DP_4) with the mask defined by the three bits (D_{11}-D_9) of the DPH field
M1	0	0	1	
M2	0	1	0	
M3	0	1	1	
M4	1	0	0	
M5	1	0	1	
M6	1	1	0	
M7	1	1	1	

Table 11. RPDCR Field

Mnemonic	D_8	RP Operation
RPNOP	0	No operation
RPDEC	1	Decrement RP

Jump/Call/Branch

Figure 5 shows the JP instruction field specification. Three types of program counter modifications accommodated by the processor are listed in table 12. All the instructions, if unconditional or if the specified condition is true, take their next program execution address from the next address field (NA); otherwise PC = PC + 1.

For the conditional jump instruction, the condition field specifies the jump condition. Table 13 lists all the instruction mnemonics of the jump/call/branch codes. BRCH or CND values not in table 13 are prohibited.

Load Data (LDI)

Figure 6 shows the LD instruction field specification. The load data instruction will take the 16-bit value contained in the immediate data field (ID) and place it in the location specified by the destination field (DST) See table 7.

Figure 5. JP Instruction Field

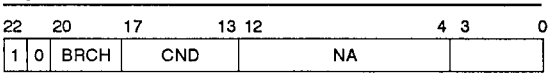


Figure 6. LD Instruction Field

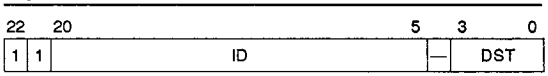


Table 12. BRCH Field

D_{20}	D_{19}	D_{18}	Branch Instruction
1	0	0	Unconditional jump
1	0	1	Subroutine call
0	1	0	Conditional jump

Table 13. BRCH/CND Fields

Mnemonic	D_{20}	D_{19}	D_{18}	D_{17}	D_{16}	D_{15}	D_{14}	D_{13}	Conditions
JMP	1	0	0	0	0	0	0	0	No condition
CALL	1	0	1	0	0	0	0	0	No condition
JNCA	0	1	0	0	0	0	0	0	CA = 0
JCA	0	1	0	0	0	0	0	1	CA = 1
JNCB	0	1	0	0	0	0	1	0	CB = 0
JCB	0	1	0	0	0	0	1	1	CB = 1
JNZA	0	1	0	0	0	1	0	0	ZA = 0
JZA	0	1	0	0	0	1	0	1	ZA = 1
JNZB	0	1	0	0	0	1	1	0	ZB = 0
JZB	0	1	0	0	0	1	1	1	ZB = 1
JNOVA0	0	1	0	0	1	0	0	0	OVA0 = 0
JOVA0	0	1	0	0	1	0	0	1	OVA0 = 1
JNOVB0	0	1	0	0	1	0	1	0	OVB0 = 1
JOVB0	0	1	0	0	1	0	1	1	OVB0 = 1
JNOVA1	0	1	0	0	1	1	0	0	OVA1 = 0
JOVA1	0	1	0	0	1	1	0	1	OVA1 = 1
JNOVB1	0	1	0	0	1	1	1	0	OVB1 = 0
JOVB1	0	1	0	0	1	1	1	1	OVB1 = 1
JNSA0	0	1	0	1	0	0	0	0	SA0 = 0
JSA0	0	1	0	1	0	0	0	1	SA0 = 1
JNSB0	0	1	0	1	0	0	1	0	SB0 = 0
JSB0	0	1	0	1	0	0	1	1	SB0 = 1
JNSA1	0	1	0	1	0	1	0	0	SA1 = 0
JSA1	0	1	0	1	0	1	0	1	SA1 = 1
JNSB1	0	1	0	1	0	1	1	0	SB1 = 0
JSB1	0	1	0	1	0	1	1	1	SB1 = 1
JDPL0	0	1	0	1	1	0	0	0	DPL = 0
JDPLF	0	1	0	1	1	0	0	1	DPL = FH
JNSIAK	0	1	0	1	1	0	1	0	SI ACK = 0
JSIAK	0	1	0	1	1	0	1	1	SI ACK = 1
JNSOAK	0	1	0	1	1	1	0	0	SO ACK = 0
JSOAK	0	1	0	1	1	1	0	1	SO ACK = 1
JNRQM	0	1	0	1	1	1	1	0	RQM = 0
JRQM	0	1	0	1	1	1	1	1	RQM = 1

3a

ELECTRICAL SPECIFICATIONS

Absolute Maximum Ratings

Supply voltage, V_{CC}	
77C20A	−0.5 to +7.0 V
7720A	−0.5 to +7.0 V
77P20	−0.3 to +7.0 V
Programming voltage, V_{PP} (77P20)	−0.3 to +22 V
Input voltage, V_I	
77C20A	−0.5 to V_{CC} + 0.5 V
7720A	−0.5 to +7.0 V
77P20	−0.3 to +7.0 V
Output voltage, V_O	
77C20A	−0.5 to V_{CC} + 0.5 V
7720A	−0.5 to +7.0 V
77P20	−0.3 to +7.0 V
Operating temperature, T_{OPT}	
77C20A	−40 to +85°C
7720A, 77P20	−10 to +70°C
Storage temperature, T_{STG}	−65 to +150°C

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

Capacitance

Parameter	Symbol	Min	Max	Unit	Conditions
CLK, SCK capacitance	$C\phi$		20	pF	f_c = 1 MHz
Input pin capacitance	C_{IN}		10	pF	
Output pin capacitance	C_{OUT}		20	pF	

DC Characteristics

T_A = −10 to +70°C; V_{CC} = +5 V ±5%

Parameter	Symbol	Min	Typ	Max	Unit	Conditions
Input low voltage	V_{IL}					
77C20A		−0.3		0.8	V	
7720A, 77P20		−0.5		0.8	V	
Input high voltage	V_{IH}					
77C20A		2.2		V_{CC} + 0.3	V	
7720A, 77P20		2.0		V_{CC} + 0.5	V	
CLK low voltage	$V_{\phi L}$					
77C20A		−0.3		0.45	V	
7720A, 77P20		−0.5		0.45	V	
CLK high voltage	$V_{\phi H}$					
77C20A		3.5		V_{CC} + 0.3	V	
7720A, 77P20		3.5		V_{CC} + 0.5	V	
Output low voltage	V_{OL}			0.45	V	I_{OL} = 2.0 mA
Output high voltage	V_{OH}	2.4			V	I_{OH} = −400 μA
Input load current	I_{LIL}			−10	μA	V_{IN} = 0 V
Input load current	I_{LIH}			10	μA	V_{IN} = V_{CC}
Output float leakage	I_{LOL}			−10	μA	V_{OUT} = 0.47 V
Output float leakage	I_{LOH}			10	μA	V_{OUT} = V_{CC}

DC Characteristics (cont)

Parameter	Symbol	Min	Typ	Max	Unit	Conditions
Power supply current	I_{CC}					
77C20A			24	40	mA	f_{CLK} = 8.192 MHz
7720A			120	170	mA	
77P20			270	350	mA	
V_{PP} current (77P20 only)	I_{PP}	—		70	mA	Program mode max pulse current (Note 1)
		0.5		3.0	mA	Program verify, inhibit (Note 2)

Notes:

(1) V_{PP} = 21 ± 0.5 V

(2) For K-level parts: V_{PP} max = (V_{CC} − 0.6 V) + 0.25 V
V_{PP} min = (V_{CC} − 0.6 V) − 0.25 V

For all other step levels: V_{PP} max = V_{CC} + 0.25 V
V_{PP} min = V_{CC} − 0.85 V

3a

AC Characteristics

T_A = −10 to +70°C; V_{CC} = +5 V ±5%

Parameter	Symbol	Min	Typ	Max	Unit	Conditions
CLK cycle time	ϕCY					
77C20A, 7720A		120		2000	ns	
77P20		122		2000	ns	
CLK pulse width	ϕD	60			ns	Note 4
CLK rise time	ϕR			10	ns	Note 1
CLK fall time	ϕF			10	ns	Note 1
Address setup time for $\overline{RD}$	t_{AR}	0			ns	
Address hold time for $\overline{RD}$	t_{RA}	0			ns	
$\overline{RD}$ pulse width	t_{RR}	250			ns	
Data delay from $\overline{RD}$	t_{RD}			150	ns	C_L = 100 pF
Read to data floating	t_{DF}	10		100	ns	C_L = 100 pF
Address setup time for $\overline{WR}$	t_{AW}	0			ns	
Address hold time for $\overline{WR}$	t_{WA}	0			ns	
$\overline{WR}$ pulse width	t_{WW}	250			ns	
Data setup time for $\overline{WR}$	t_{DW}	150			ns	
Data hold time for $\overline{WR}$	t_{WD}	0			ns	
$\overline{RD}$, $\overline{WR}$, recovery time	t_{RV}	250			ns	Note 2
DRQ delay	t_{AM}			150	ns	C_L = 100 pF
$\overline{DACK}$ delay time	t_{DACK}	1			ϕD	Note 2
$\overline{DACK}$ pulse width	t_{DD}					
77C20A		250				
7720A		250		2000	ns	
77P20		250		50,000	ns	
SCK cycle time	t_{SCY}	480		DC	ns	
SCK pulse width	t_{SCK}	230			ns	
SCK rise/fall time	t_{RSC}/t_{FSC}		20		ns	

AC Characteristics (cont)

Parameter	Symbol	Min	Typ	Max	Unit	Conditions
SORQ delay	t_{DRQ}	30		150	ns	C_L = 100 pF
$\overline{SOEN}$ hold time	t_{CSO}	30			ns	
$\overline{SOEN}$ setup time	t_{SOC}	50			ns	
SO delay from SCK = low	t_{DCK}			150	ns	
SO delay from SCK before 1st bit (Note 3)	t_{DZRQ}	20		300	ns	Note 2
SO delay from SCK	t_{DZSC}	20		300	ns	Note 2
SO delay for $\overline{SOEN}$	t_{DZE}	20		180	ns	Note 2
$\overline{SOEN}$ to SO floating	t_{HZE}	20		200	ns	Note 2
SCK to SO floating with SORQ high	t_{HZSC}	20		300	ns	Note 2
SO delay from SCK for last bit	t_{HZRQ}	70		300	ns	Note 2
$\overline{SIEN}$, SI setup time	t_{DC}	55			ns	Note 2
$\overline{SIEN}$, SI hold time	t_{CD}	30			ns	
P_0, P_1 delay	t_{DP}			ϕCY + 150	ns	
RST pulse width	t_{RST}	4			ϕCY	
INT pulse width	t_{INT}	8			ϕCY	

Notes:

(1) Voltage at timing measuring point: 1.0 V and 3.0 V.

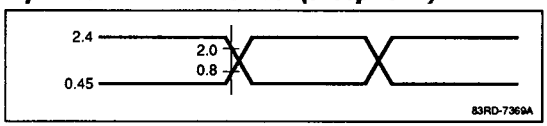
(2) Voltage at ac timing measuring point:
$V_{IL} = V_{OL} = 0.8$ V
$V_{IH} = V_{OH} = 2.0$ V

(3) SO goes out of tristate, but data is not valid yet.

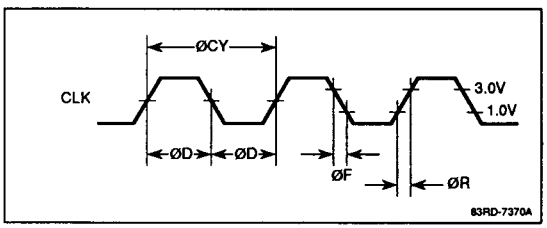
(4) Pulse width includes CLK rise and fall times. Refer to Clock Timing Waveform.

Timing Waveforms

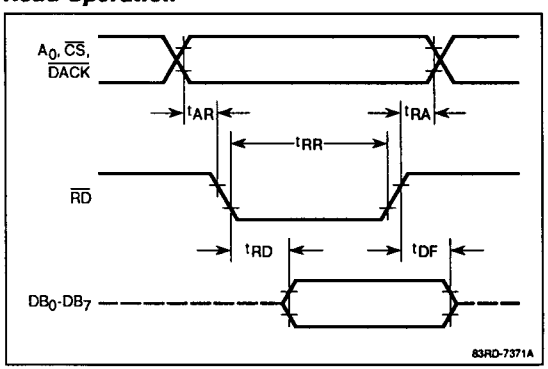
Input Waveform of AC Test (except CLK)



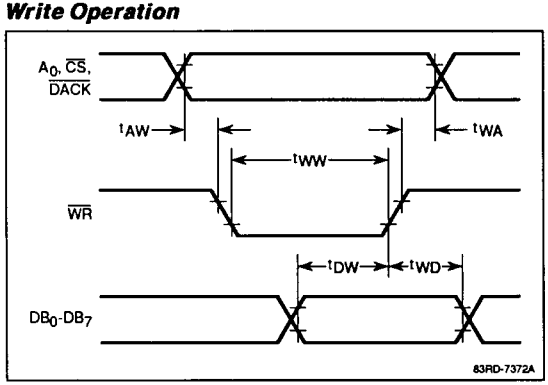
Clock



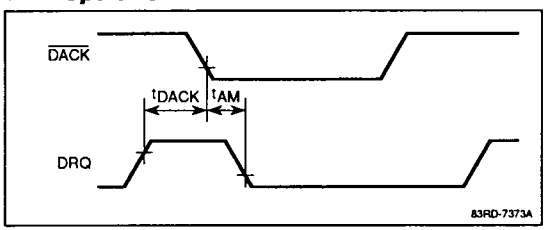
Read Operation



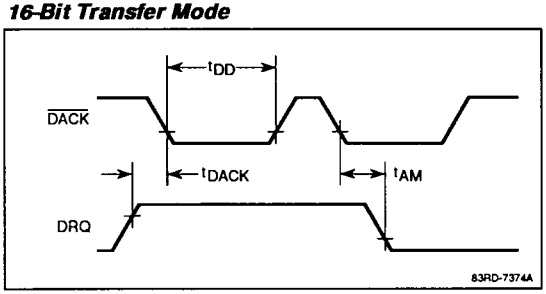
Write Operation



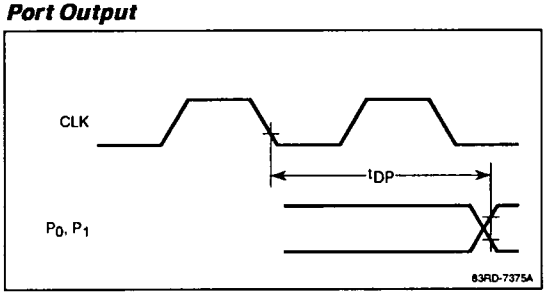
DMA Operation



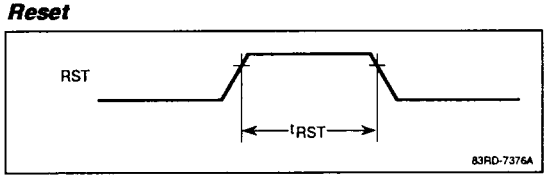
16-Bit Transfer Mode



Port Output



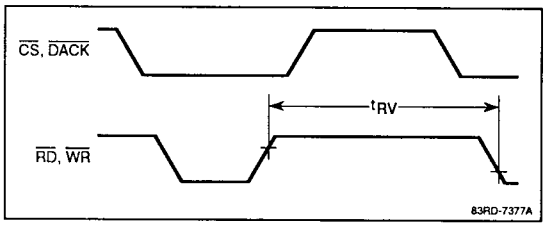
Reset



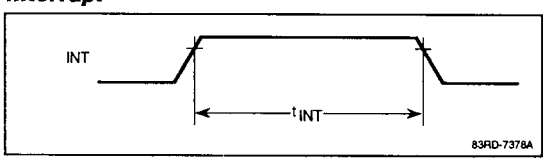
3a

Timing Waveforms (cont)

Read/Write Cycle



Interrupt



SERIAL TIMING

Serial Output, Case 1

Figure 7 shows serial output timing when $\overline{SOEN}$ is asserted in response to SORQ when SCK is low. If $\overline{SOEN}$ is held inactive until after SORQ is asserted, and then $\overline{SOEN}$ is asserted while SCK is low ($\overline{SOEN}$ should be held inactive until the period of t_{CSO} after the falling edge of SCK), SO will become active but not valid t_{DZSC} after the next rising edge of SCK. SO will become valid with the first bit t_{DCK} after the next falling edge of SCK for use by an external device at the subsequent rising edge of SCK.

Subsequent bits will be shifted out t_{DCK} after subsequent falling edges of SCK for use at subsequent rising edges of SCK. The last bit to be shifted out will also follow this pattern and will be held valid t_{HZRQ} after the corresponding rising edge of SCK at which it is to be used. SORQ will be held t_{DRQ} after this same rising edge of SCK and then removed. $\overline{SOEN}$ should be released at least t_{SOC} before the next falling edge of SCK.

Serial Output, Case 2

Figure 8 shows timing for serial output when $\overline{SOEN}$ is asserted in response to SORQ when SCK is high. If $\overline{SOEN}$ is held inactive until after SORQ is asserted, and then $\overline{SOEN}$ is asserted while SCK is high (at least t_{SOC} before the falling edge of SCK), SO will become active but not valid t_{DZE} after the falling edge of $\overline{SOEN}$. SO will become valid t_{DCK} after the falling edge of SCK for use by an external device at the subsequent rising edge of SCK.

Note that although figure 8 shows $\overline{SOEN}$ being asserted during a different SCK pulse than the one in which SORQ is asserted, it is permissible for these to occur during the same pulse of SCK as long as $\overline{SOEN}$ is still asserted t_{SOC} before the falling edge of SCK. The timing for the second through the last bits is identical to the timing shown in figure 7.

Serial Output, Case 3

Figure 9 shows output timing when $\overline{SOEN}$ is active before SORQ is high. If $\overline{SOEN}$ is held active before SORQ is high, data will be shifted out whenever it becomes available in the serial output register (assuming previous data is already shifted out). In this case, SORQ will rise t_{DRQ} after a rising edge of SCK. SO will become active (but not valid yet) t_{DZRQ} after the same rising edge of SCK. The first valid SO bit occurs t_{DCK} after the next falling edge of SCK for use by an external device at the subsequent rising edge of SCK.

Subsequent bits will be shifted out t_{DCK} after subsequent falling edges of SCK for use at subsequent rising edges of SCK. The last bit to be shifted out will also follow this pattern and will be held valid t_{HZRQ} after the corresponding rising edge of SCK at which it is to be used. SORQ will be held t_{DRQ} after this same rising edge of SCK and then removed.

Figure 7. Serial Output Case 1: $\overline{SOEN}$ Asserted in Response to SORQ When SCK Is Low

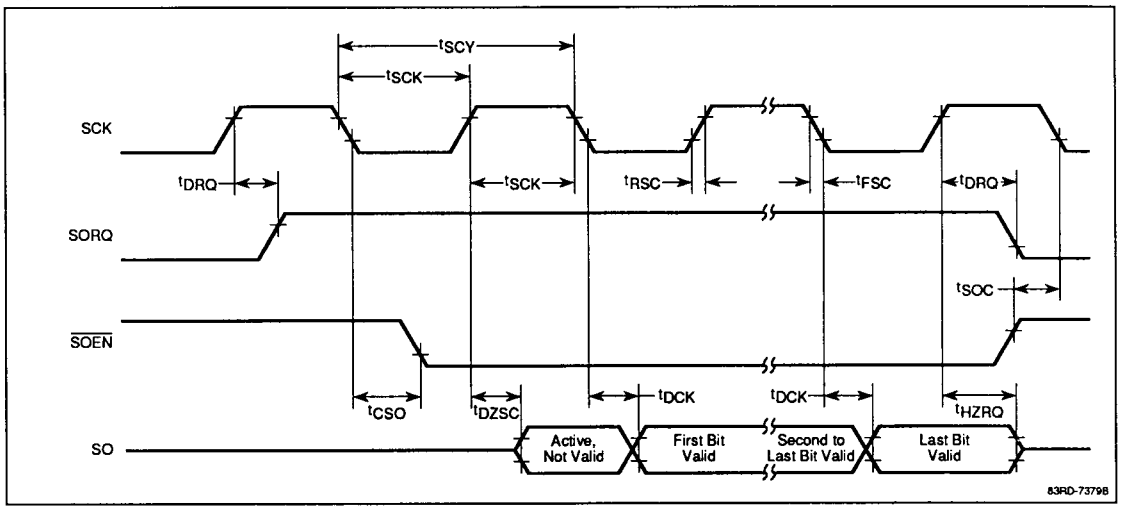
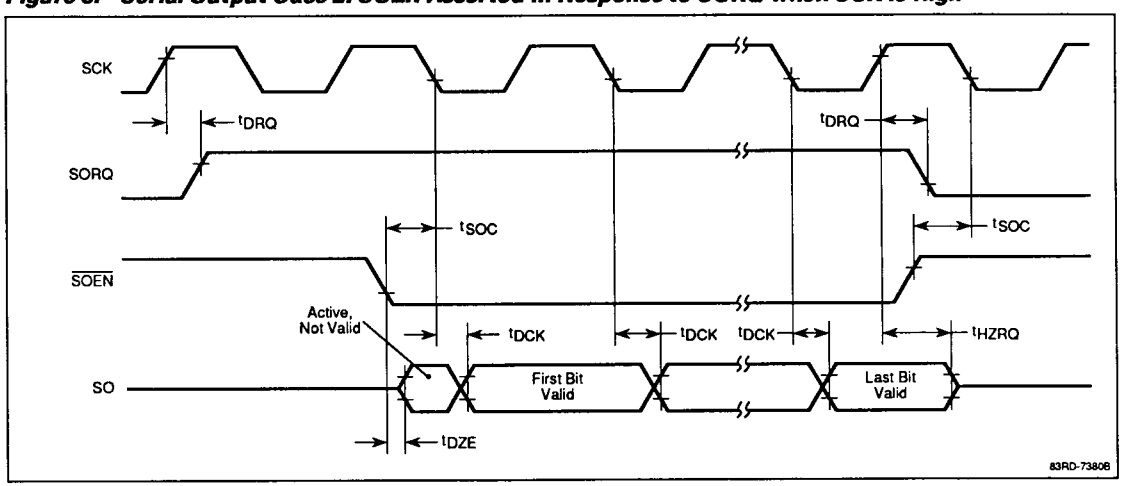
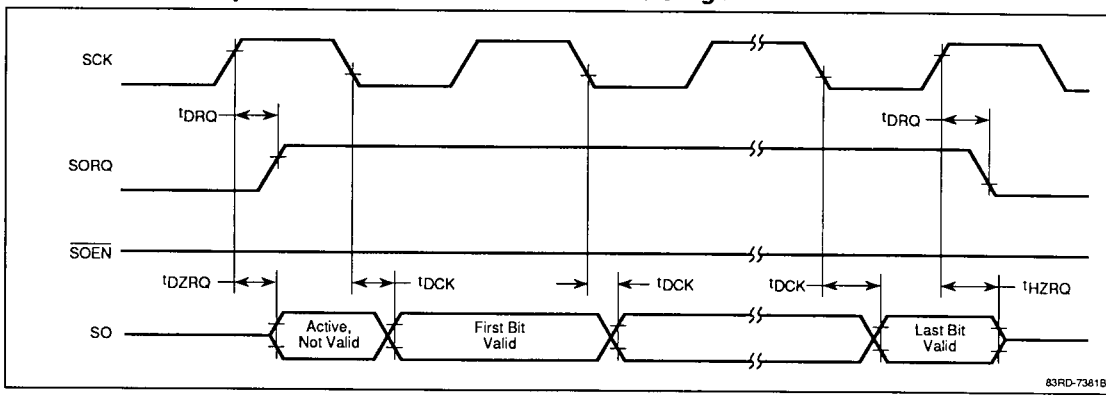


Figure 8. Serial Output Case 2: $\overline{SOEN}$ Asserted in Response to SORQ When SCK Is High



3a

Figure 9. Serial Output Case 3: $\overline{SOEN}$ Active Before SORQ Is High



Serial Output, Case 4A

Avoid releasing $\overline{SOEN}$ in the middle of a transfer (that is, before the last bit is shifted out), since this will stop the output shift operation. When $\overline{SOEN}$ is again asserted, the remainder of the transfer will be shifted out before the next transfer can begin. The next transfer will begin immediately without any indication of the byte/word boundary. If $\overline{SOEN}$ is released while SCK is high (figure 10) at least t_{SOC} before the falling edge of SCK, then SO will go inactive t_{HZE} after $\overline{SOEN}$ is released (which may be before or after the falling edge of SCK).

Serial Output, Case 4B

If $\overline{SOEN}$ is released while SCK is low (figure 11) at least t_{CSO} after the falling edge of SCK, then the next bit will be shifted out t_{DCK} after the falling edge of SCK for useat the subsequent rising edge of SCK. SO will then go inactive t_{HZSC} after this rising edge of SCK.

Note: For all its uses, $\overline{SOEN}$ must not change state within t_{SOC} before or t_{CSO} after the falling edge of SCK; otherwise, the results will be indeterminate.

Serial Input

Serial input timing (figure 12) is much simpler than serial output timing. Data bits are shifted in on the rising edge of SCK if $\overline{SIEN}$ is asserted. Both $\overline{SIEN}$ and SI must be stable at least t_{DC} before and t_{CD} after the rising edge of SCK; otherwise the results will be indeterminate.

Figure 10. Serial Output Case 4A: If $\overline{SOEN}$ Is Released in the Middle of a Transfer During SCK High

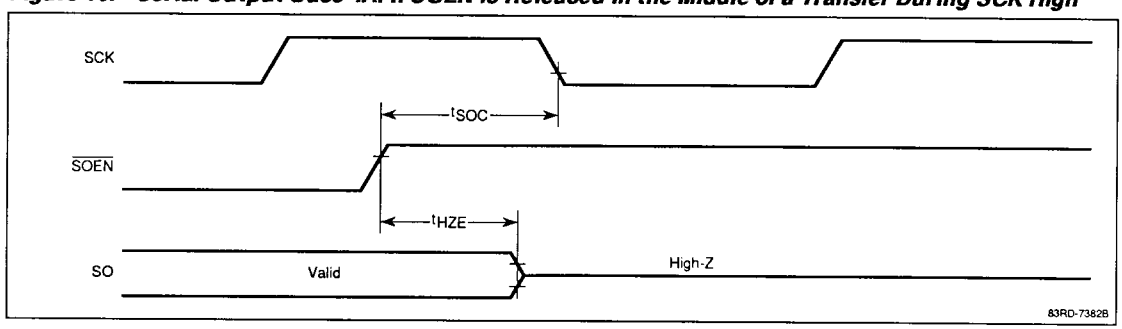


Figure 11. Serial Output Case 4B: If $\overline{SOEN}$ Is Released in the Middle of a Transfer During SCK Low

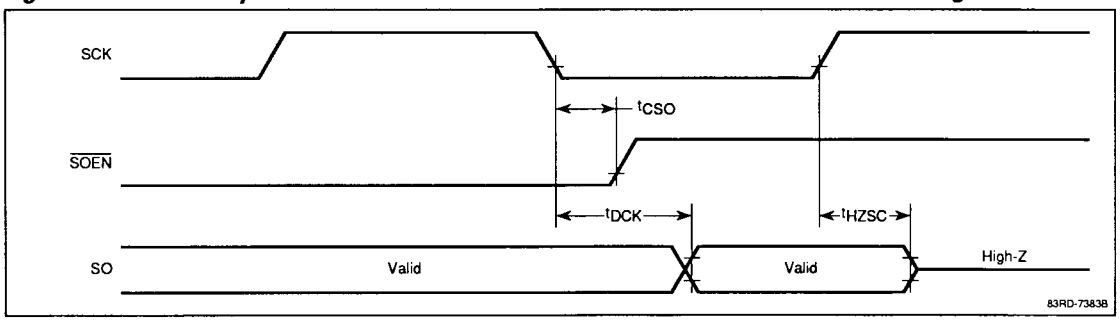
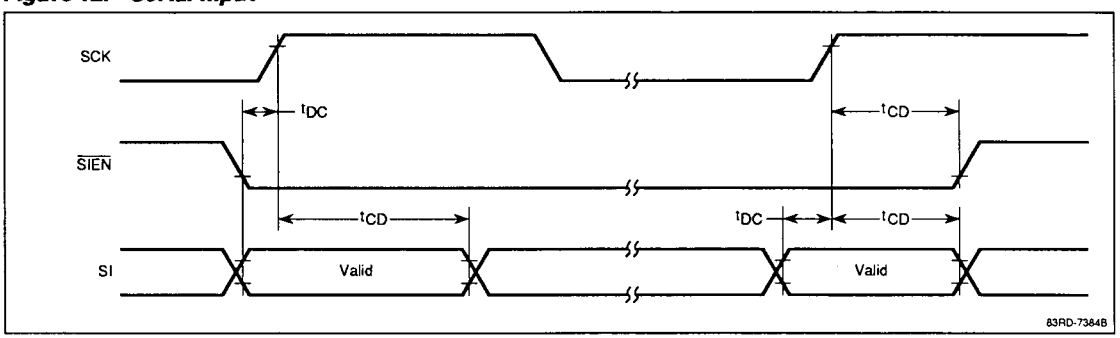


Figure 12. Serial Input



3a

Serial Timing Example

Figure 13 shows serial timing of cascaded SPIs with a common SCK. SO from the first SPI equals SI of the second, and the first SPI's SORQ inverts to become $\overline{SIEN}$ of the second. $\overline{SOEN}$ of the first SPI is always asserted.

When cascading two SPIs in the described configuration, most of the timing involved is directly copied from the case of serial output with $\overline{SOEN}$ always enabled (figure 13). It must be shown that the results will be suitable for the serial input timing of the second SPI.

(1) SORQ(1) rises t_{DRQ} after a rising edge of SCK, and it is inverted (inverter has t_{PHL} delay time) to become $\overline{SIEN}$(2), which must be stable t_{DC} before the next rising edge of SCK. It also must not change until t_{CD} after this first rising edge of SCK as shown by case 2 in figure 8.

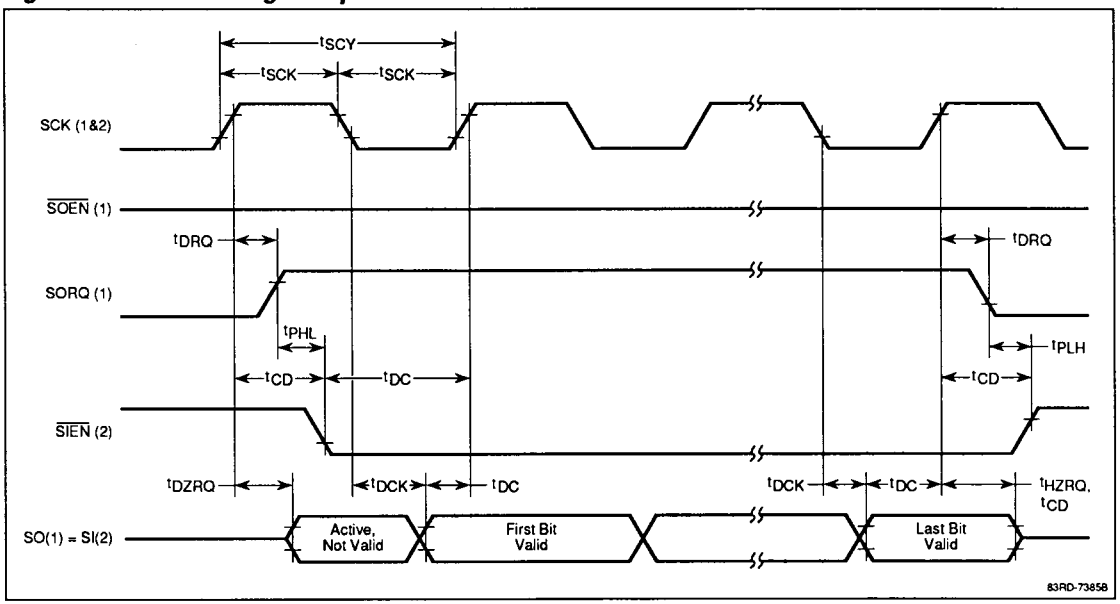
$$t_{DRQ}\ (max) + t_{PHL} + t_{DC}\ (min) \leq t_{SCY}\ (min)$$
$$t_{PHL}\ (max) \leq t_{SCY}\ (min) - t_{DC}\ (min) - t_{DRQ}\ (max)$$
$$\leq 480 - 55 - 150$$
$$\leq 275\ ns\ (readily\ achieved\ by\ 74LS14,\ for\ example)$$

(2) SORQ(1) is released t_{DRQ} after the last useful rising edge of SCK and is inverted (inverter has t_{PHL} delay time) to become $\overline{SIEN}$(2), which must remain stable t_{CD} after the rising edge of SCK.

$$t_{DRQ}\ (min) + t_{PLH}\ (min) \geq t_{CD}\ (min)$$
$$t_{PLH}\ (min) \geq t_{CD}\ (min) - t_{DRQ}\ (min)$$
$$\geq 30 - 30$$
$$\geq 0\ (no\ problem,\ assuming\ causality)$$

Note: This also shows t_{PHL} (min) ≥ 0 for the rising edge of SORQ.

Figure 13. Serial Timing Example



(3) SO(1) is valid t_{DCK} after a falling edge of SCK; since it becomes SI(2), it must be valid t_{DC} before the next rising edge of SCK.

t_{DCK} (max) + t_{DC} (min) ≤ t_{SCK} (min)
150 + 55 ≤ 230
205 ≤ 230 (this condition is satisfied)

(4) SO(1) remains valid t_{HZRQ} after the last useful rising edge of SCK; since it becomes SI(2), it must remain valid t_{CD} after this rising edge of SCK.

t_{HZRQ} (min) ≥ t_{CD} (min)
70 ≥ 30 (this condition is satisfied)

Note: The above calculations may need to be adjusted for rise and fall times, since t_{SCY} and t_{SCK} are measured for midpoints of wave slopes.

μPD77P20 UV ERASABLE EPROM VERSION

Function

The 77P20 operates from a single +5-volt power supply and can accordingly be used in any 77C20A/7720A masked ROM application.

Use of Evakit-7720

The following sections describe electrical conditions that are required for programming the 77P20. However, the Evakit-7720, NEC's hardware emulator development tool for the 77C20A/7720A/77P20, meets the electrical and timing specifications presented below. When the Evakit-7720 is used for programming 77P20, all data transfers and formatting are handled automatically by Evakit's monitor program. Please refer to the Evakit-7720(B) User's Manual for programming procedures.

The information presented below in the sections on Configuration, Operation, and Programming (and the various subsections) is required only for users who do NOT intend to use an Evakit to program the 77P20.

Configuration

Data transfer for programming and reading the internal ROM is partitioned into three bytes for each 23-bit wide instruction location and into two bytes for each 13-bit wide data location. Partitioning of data transfer into and out of the data port is shown in figure 14.

Figure 14. Instruction ROM Format

MSB										LSB
22	21	20	19	18	17	16	15	14	13	12
11	10	9	8	7	6	5	4	3	2	1	0

Instruction ROM

The instruction ROM data is transferred through the data port as a high byte, middle byte, and low byte as shown in figure 15. Bit 7 of the middle byte should be assigned a value of zero. Data is presented to the data port in a bit-reversed format. The LSB through the MSB of an instruction ROM byte is applied to the MSB through the LSB of the data port, respectively.

Data ROM

Figure 16 shows the data ROM format. The data ROM data is transferred through the data port as a low byte and a high byte as shown in figure 17. Bits 0, 1, and 2 of the low byte should be assigned a value of zero. Data is presented to the data port in corresponding order. The MSB through the LSB of a data ROM byte is applied to the MSB through the LSB of the data port, respectively.

Initially and after each erasure, all bits of the 77P20 are in the zero state.

Figure 15. Transfer of Instruction ROM Data

Data Port	7	6	5	4	3	2	1	0
High Byte	15	16	17	18	19	20	21	22
Middle Byte	*	8	9	10	11	12	13	14
Low Byte	0	1	2	3	4	5	6	7

* Set to 0 as dummy data.

Figure 16. Data ROM Format

MSB												LSB
12	11	10	9	8	7	6	5	4	3	2	1	0

Figure 17. Transfer of Data ROM Data

Data Port	7	6	5	4	3	2	1	0
High Byte	12	11	10	9	8	7	6	5
Low Byte	4	3	2	1	0	*	*	*

* Set to 0 as dummy data.

Operating Modes

In order to read or write the instruction or data ROMs, the mode of operation of the 77P20 must be initially set. At the RST trailing edge, the $\overline{RD}$, $\overline{WR}$, and $\overline{CS}$ should be logical zero and the $\overline{DACK}$, A_0, and SI signals should be set to determine the mode of operation accordingly, as set out in table 14.

3a

Table 14. μPD77P20 Operation Mode

$\overline{DACK}$	A_0	SI	
0	0	0	Write mode instruction and data ROM
0	0	1	Read the instruction ROM
0	1	0	Read the data ROM

Once set, the 77P20 will remain in the selected mode. A reset is required to transfer to another mode.

Write Mode

The individual instruction ROM and data ROM bytes are specified by control signals $\overline{RD}$, A_0, SI, and INT as set out in table 15. Before writing the EPROM location, the bytes should be loaded accordingly.

Table 15. Write Mode Specification of ROM Bytes

$\overline{RD}$	A_0	SI	INT	
1	0	0	1	Write instruction byte, high
1	0	1	0	Write instruction byte, middle
1	0	1	1	Write instruction byte, low
1	1	0	0	Write data byte, low
1	1	0	1	Write data byte, high

Read Mode

The instruction ROM and data ROM bytes are specified by the control signals $\overline{RD}$, A_0, SI, and INT as set out in table 16. Reading is accomplished by setting the control signals accordingly.

Table 16. Read Mode Specification of ROM Bytes

$\overline{RD}$	A_0	SI	INT	
0	0	0	1	Read instruction byte, high
0	0	1	0	Read instruction byte, middle
0	0	1	1	Read instruction byte, low
1	0	0	0	Read data byte, high and low

The instruction ROM and data ROM are addressed by the 9-bit program counter and the 9-bit ROM pointer respectively. The PC is reset to 000H and is automatically incremented to the end address 1FFH. The RP is reset to 1FFH and is automaticaly decremented to 000H.

Erasing

Programming can occur only when all data bits are in an erased or low (0) level state. Erase 77P20 programmed data by exposing it to light with wavelengths shorter than approximately 4000 angstroms. Note that constant exposure to direct sunlight or room level fluorescent lighting could erase the 77P20. Consequently, if the 77P20 will be exposed to these types of lighting conditions for long periods of time, mask its window to prevent unintentional erasure.

The recommended erasure procedure for the 77P20 is exposure to ultraviolet light with wavelength of 2537 angstroms. The integrated dose (i.e., UV intensity x exposure time) for erasure should not be less than 15 W·s/cm^2. The erasure time is approximately 20 minutes using an ultraviolet lamp with a power rating of 12,000 μW/cm^2.

During erasure, place the 77P20 within 1 inch of the lamp tubes. If the lamp tubes have filters, remove the filters before erasure.

Programming

Programming of the 77P20 is achieved with a single 50-ms TTL pulse. Total programming time for the 11,776 bits of instruction EPROM and also for the 6630 bits of data EPROM is 26 seconds. Data is entered by programming a high level in the chosen bit locations. Both instruction ROM and data ROM should be programmed since they cannot be erased independently. Both instruction ROM and data ROM programming modes are entered in the same manner.

The device must be reset initially before it can be placed into the programming mode. After reset, the $\overline{WR}$ signal and all other inputs ($\overline{RD}$, $\overline{CS}$/PROG, $\overline{DACK}$, A_0, SI, and INT) should be a TTL low signal t_{RS} prior to the falling edge of RST. $\overline{WR}$ is then held for t_{RH} before being set to a TTL high-level signal. The device is now in a programming mode and will stay in this mode, allowing ROM locations to be sequentially programmed.

Programming Mode of Instruction ROM. Instruction ROM locations are sequentially programmed from address 000H to address 1FFH. The location address is incremented by the application of CLK for a duration of t_{CY}. Data bytes for each location as specified by control signals $\overline{RD}$, A_0, SI, and INT (table 15) are clocked into the device by the falling edge of $\overline{RD}$.

After the three bytes have been loaded into the device, V_{PP} is raised to 21 V ±0.5 V, t_{VS} prior to $\overline{CS}$/PROG transitioning to a TTL high-level signal. V_{PP} is held for the duration of t_{PRPR} plus t_{PRV} before returning to the V_{CC} level. After t_{PRCL}, the instruction ROM address can be incremented to program the next location. Figure 18 shows the programming mode of instruction ROM timing.

Programming Mode of Data ROM. Data ROM locations are sequentially programmed from address 1FFH to address 000H. The location address is decremented by the application of CLK for t_{CY}. The data bytes for each location as specified by control signals $\overline{RD}$, A_0, SI, and $\overline{INT}$ are clocked into the device by the falling edge of $\overline{RD}$.

After the two bytes have been loaded into the device, V_{PP} is raised to 21 V, ±0.5 V t_{VPR} prior to $\overline{CS}$/PROG transitioning to a TTL high-level signal. V_{PP} is held for the duration of t_{PRPR} plus t_{PRV} before returning to the V_{CC} level. After t_{PRCL}, the data ROM address can be decremented to program the next location. Figure 19 shows programming mode of data ROM timing.

Read Mode. A read should be performed to verify that the data was programmed correctly. Prior to entering read mode, the device must be reset.

Read Mode of Instruction ROM. This mode is entered by holding the $\overline{WR}$ signal at a TTL low level with the SI signal at a TTL high level and all other specified inputs ($\overline{RD}$, $\overline{CS}$/PROG, $\overline{DACK}$, A_0, INT) at TTL low levels for t_{CORS} prior to the falling edge of RST. $\overline{WR}$ is then held for t_{RSW} before being set to a TTL high level. The device is now in the instruction ROM read mode and will stay in this mode until reset.

Instruction ROM locations are sequentially read from address 000H through 1FFH. Application of CLK for t_{CY} will increment the location address. The three data bytes will be read as specified by the control signals $\overline{RD}$, A_0, SI, and INT (table 16). Figure 20 shows read mode of instruction ROM timing.

Figure 18. Programming Mode of Instruction ROM

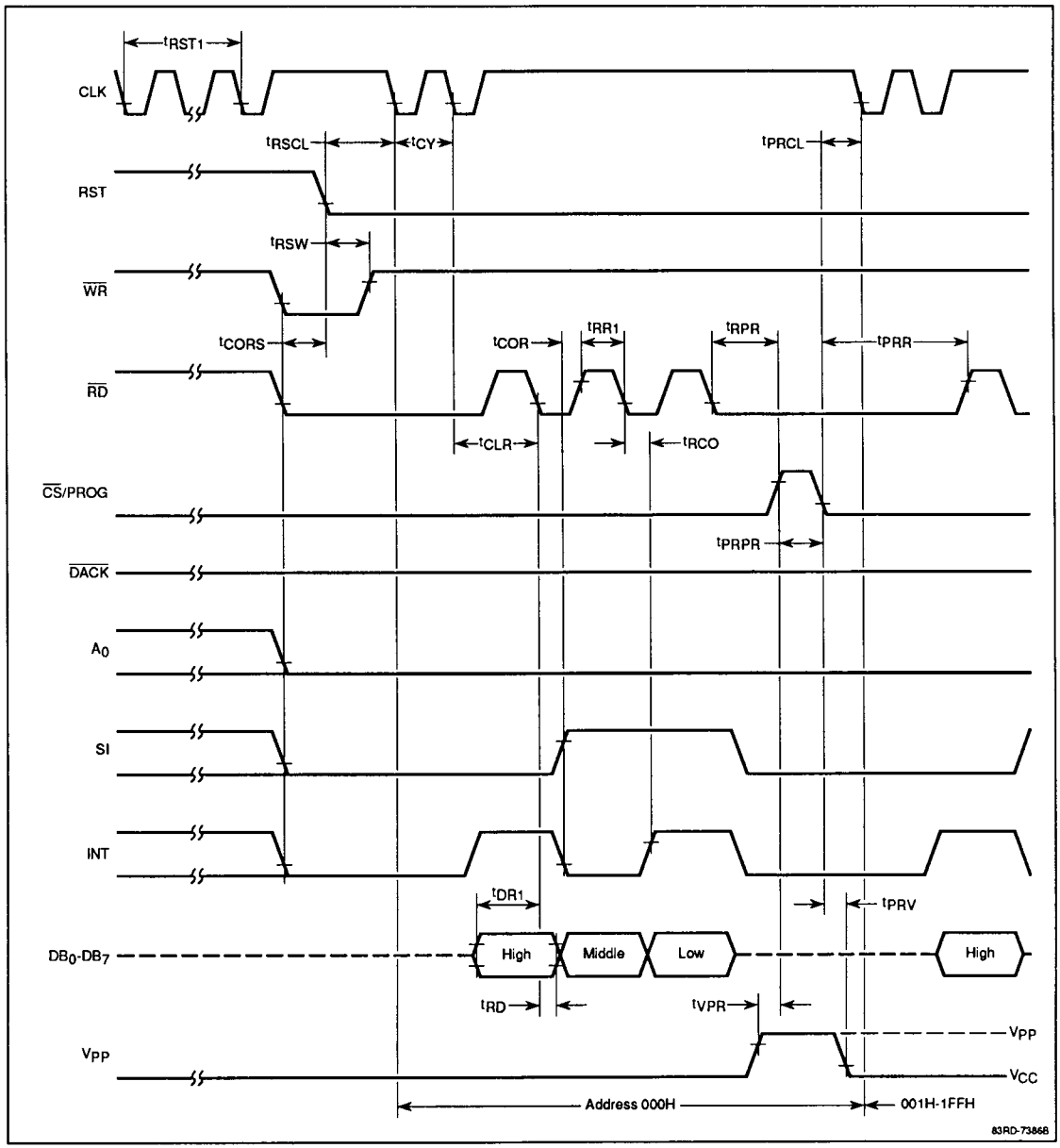


Figure 19. Programming Mode of Data ROM

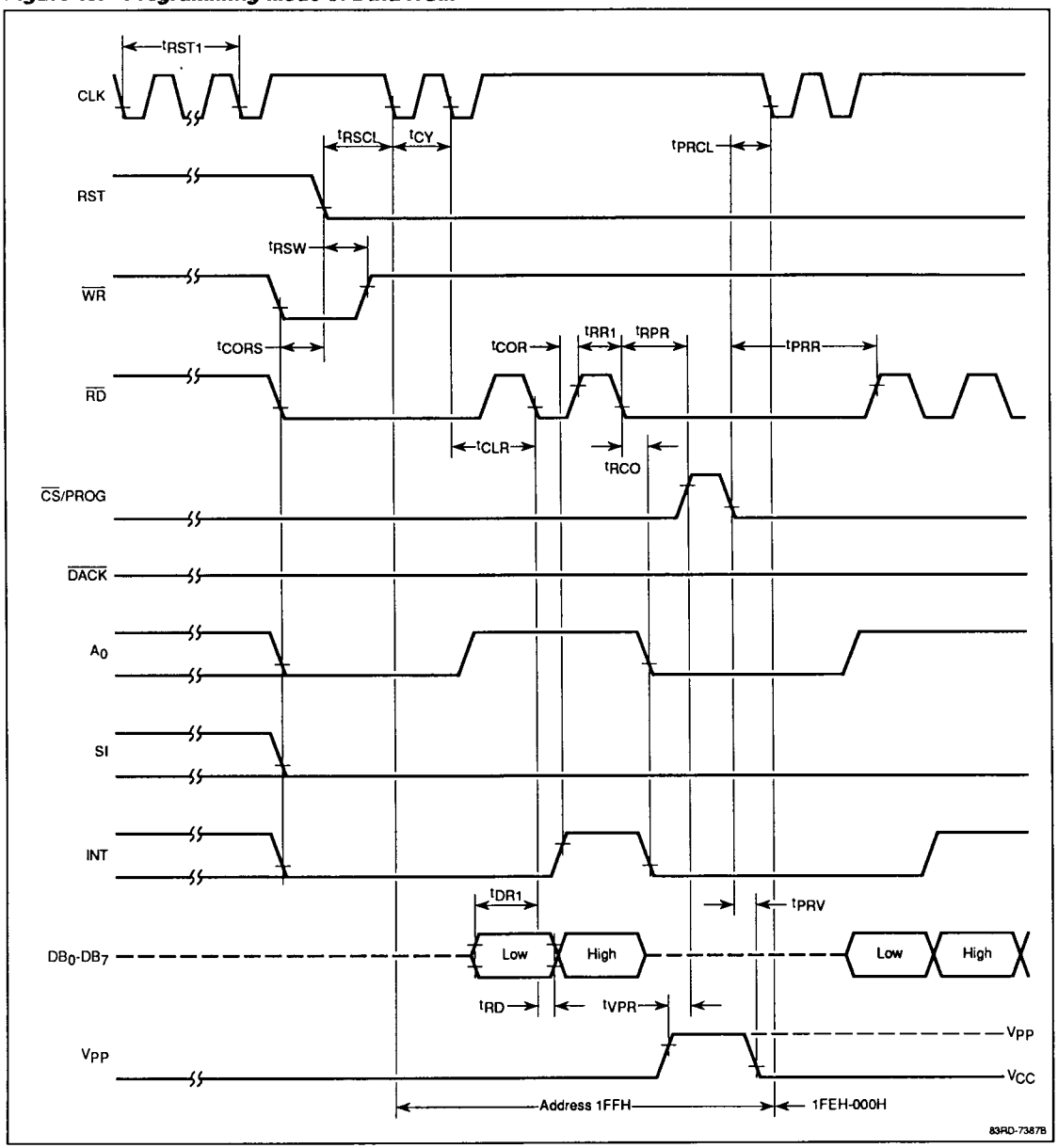


Figure 20. Read Mode of Instruction ROM

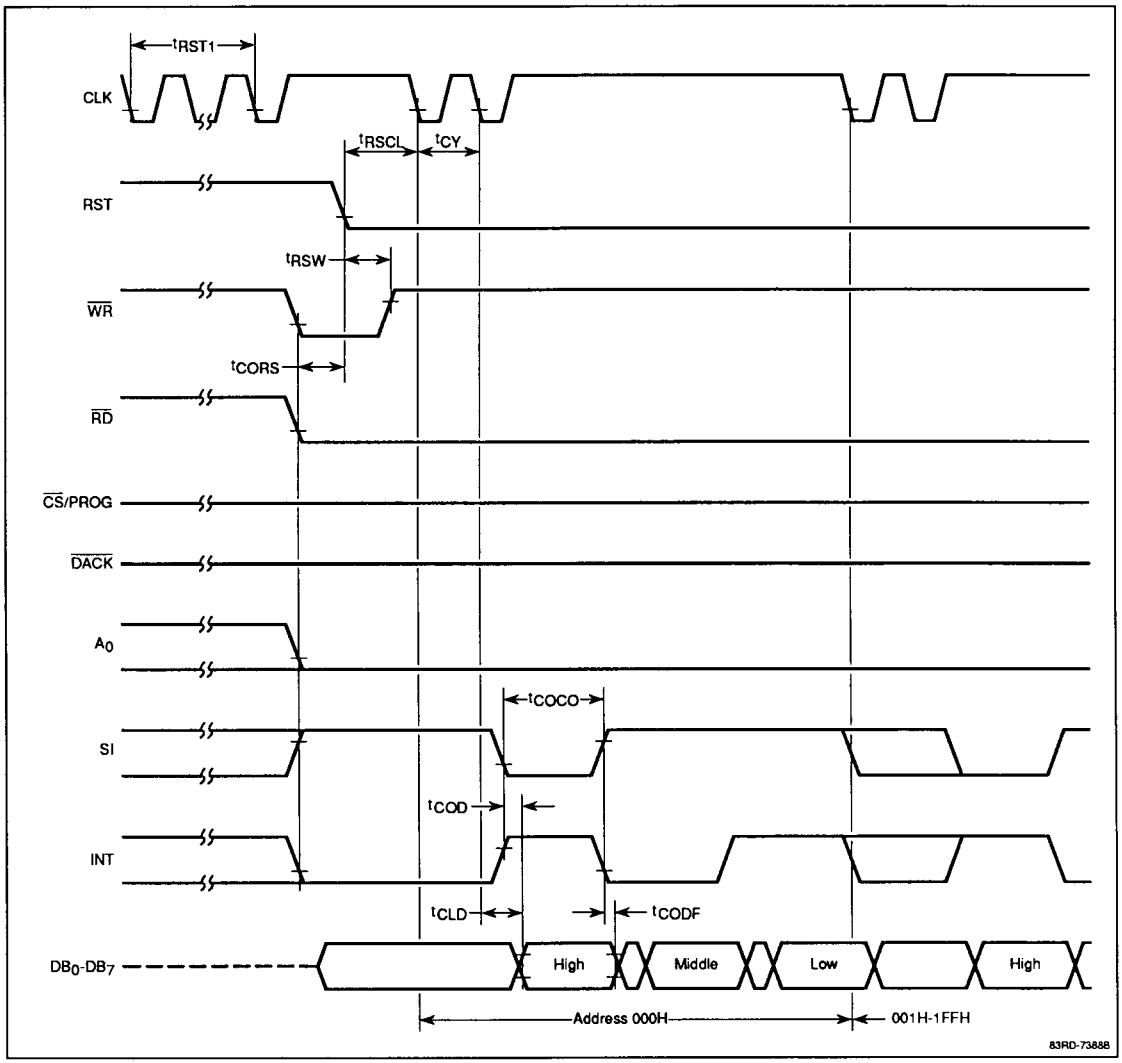
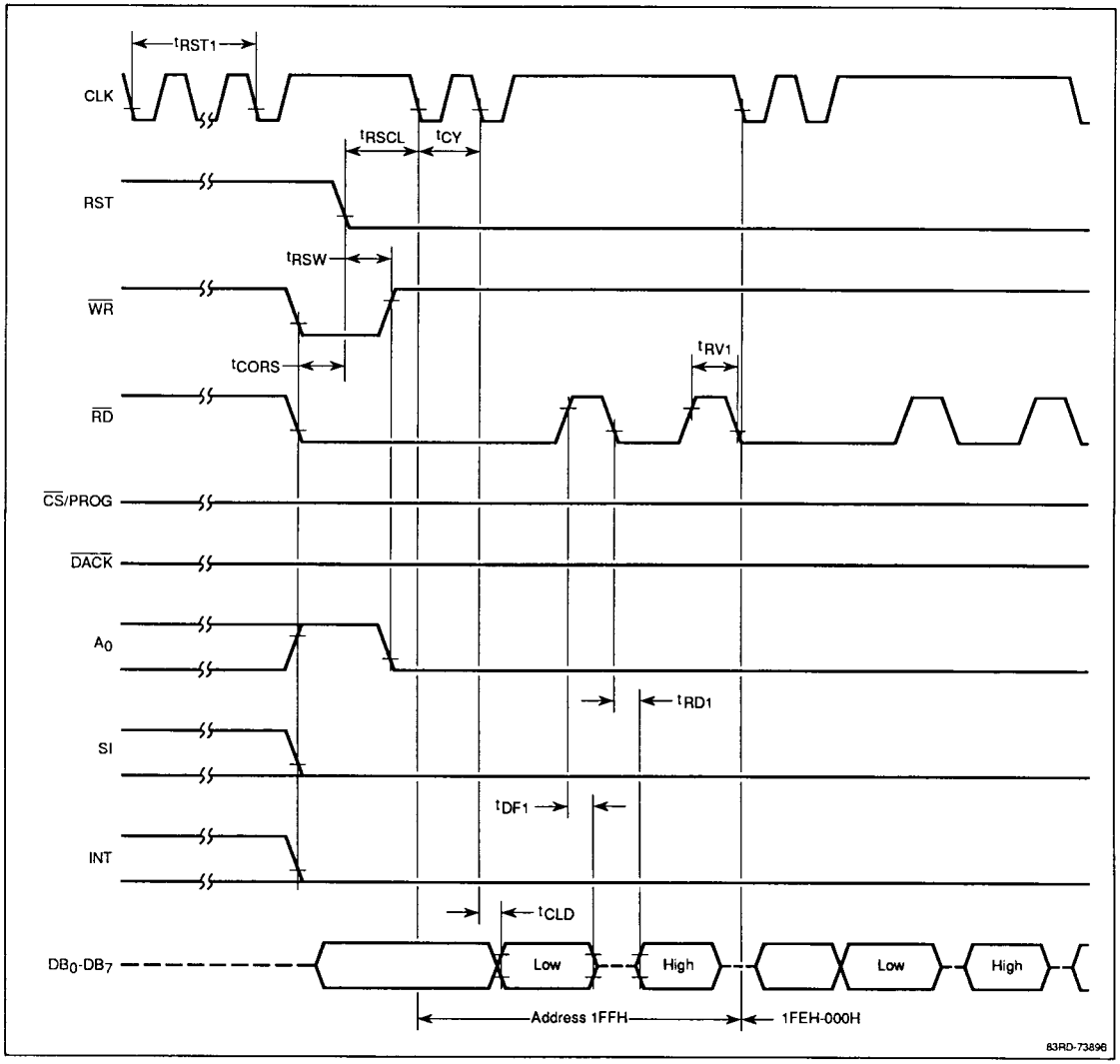


Figure 21. Read Mode of Data ROM



Read Mode of Data ROM. Figure 21 shows read mode of data ROM timing. This mode is entered by holding the $\overline{WR}$ signal at a TTL low level with the A_0 signal at a TTL high level and all other specified inputs ($\overline{RD}$, $\overline{CS}$/PROG, $\overline{DACK}$, SI, INT) at TTL low levels for t_{CORS} prior to the falling edge of RST. $\overline{WR}$ and A_0 are then held for t_{CORS} prior to the falling edge of RST. $\overline{WR}$ and A_0 are then held for t_{RSW} before being set to a TTL high level and TTL low level, respectively. The device is now in the data ROM read mode and will stay in this mode until it is reset.

Data ROM locations are sequentially read from address 1FFH through 000H. Application of CLK for t_{CY} will decrement the location address. After the address has been decremented, the low byte of the current location

will be available at the data port subsequent to a t_{CLD} delay. Application of $\overline{RD}$ will present the high byte t_{RD1} from the falling edge of the $\overline{RD}$ pulse. $\overline{RD}$ is then applied for t_{RV1} to complete reading of the current location.

Read Operation, AC Characteristics

T_A = 25°C ±5°C; V_{CC} = 5 V ±5%; V_{PP} = V_{CC} + 0.25 V max; V_{PP} = V_{CC} − 0.85 V min

Parameter	Symbol	Min	Max	Unit	Conditions
Data access time from CLK	t_{CLD}		1	μs	
Data delay time from SI, IN ↑	t_{COD}		1	μs	
Data flat time from SI, IN ↑	t_{CODF}	0		ns	
SI, INT pulse width	t_{COCO}	1		μs	
$\overline{RD}$ recovery time	t_{RV1}	500		ns	
Data access time from $\overline{RD}$ ↓	$t_{\overline{RD}1}$		150	ns	
Data float time from $\overline{RD}$ ↑	t_{DF1}	10		ns	

3a

Programming Operation, AC Characteristics

T_A = 25°C ±5°C; V_{CC} = 5 V ±5%; V_{PP} = 21 V ±0.5 V

Parameter	Symbol	Min	Typ	Max	Unit	Conditions
CLK cycle time	t_{CY}	240			ns	
CLK setup time to $\overline{RD}$ ↓	t_{CLR}	2			μs	
CLK hold time from RST ↓	t_{RSCL}	6			μs	
CLK hold time from PROG ↓	t_{PRCL}	200			ns	
Control signal set-up time to RST ↓	t_{CORS}	1			μs	
$\overline{WR}$ hold time from RST ↓	t_{RSW}	6			μs	
Data set-up time from $\overline{RD}$ ↓	t_{DR1}	1			μs	
Data hold time from $\overline{RD}$ ↓	t_{RD}	100			ns	
$\overline{RD}$ pulse width	t_{RR1}	1			μs	
SI, INT set-up time from $\overline{RD}$ ↑	t_{COR}	100			ns	
SI, INT hold time from $\overline{RD}$ ↓	t_{RCO}	100			ns	
$\overline{RD}$ set-up time to PROG ↑	t_{RPR}	100			ns	
$\overline{RD}$ hold time from PROG ↓	t_{PRR}	2			μs	
V_{PP} set-up time to PROG ↑	t_{VPR}	2			μs	
V_{PP} hold time from PROG ↓	t_{PRV}	2			μs	
RST pulse width	t_{RST1}	4			t_{CY}	
PROG pulse width	t_{PRPR}	45	50	55	ms	

Operation Mode

The 77P20 may be utilized in an operation mode after the instruction ROM and data ROM have been programmed. Since it was first introduced in 1982, the 77P20 has undergone several mask revisions to improve manufacturability and/or function. And since the purpose of the 77P20 is to run any program that may be programmed in the masked ROM 77C20A/7720A, it is important to know how to determine the step levels and the differences between them.

Step Level

The markings on the μPD77P20 package consist of three lines, as follows:

NEC JAPAN	Manufacturer
D77P20D	Part Number
nnnnXnnnn	Date code

In the date code, "X" identifies the step level of the part. Parts marked with step level K, E, or P should not be used for final system test by customers who are planning to submit code for the masked ROM 77C20A/7720A.

On all other 77P20 step versions, a slight functional change was made, and the change is incorporated in the 77C20A/7720A. The change allows the serial clock (SCK) to run asynchronously with CLK. Specified versions of 77P20 (i.e. K, E, P) and all Evakit-7720s and Evakit-7720Bs (Evaluation Systems for 77C20A/ 7720A/ 77P20) require that SCK run synchronously with CLK.

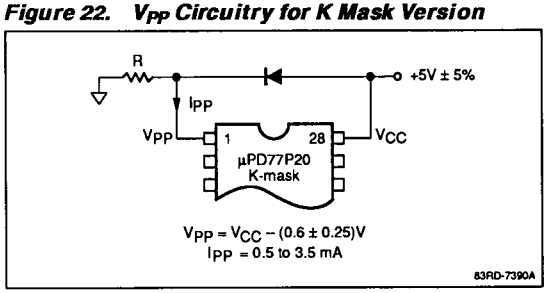
Because this functional change results in a slight change in internal serial timing, it is mandatory that code to be submitted for 77C20A/7720A be verified in customer's system using versions of 77P20 other than those listed above (i.e. K, E, P).

Pin 1 Connection

The K mask version requires that the programming voltage V_{PP} be supplied in a different manner than for all later versions, as shown in figure 22. A silicon junction diode of 0.6 V forward voltage (V_F) should be used. R should be 800 to 1800 Ω to satisfy the V_{PP} and I_{PP} requirements.

In all mask versions other than K, pin 1 must be connected directly to V_{CC}.

Figure 22. V_{PP} Circuitry for K Mask Version



DEVELOPMENT TOOLS

For software development, assembly into object code, and debugging, an absolute assembler and simulator are available. The ASM77 Absolute Assembler and SM77C25 Simulator for analyzing development code and I/O timing characteristics are available for systems supporting CP/M® and CP/M-86®, ISIS-II®, or MS-DOS® operating systems. Additionally, the ASM77 Absolute Assembler is offered in Fortran source code for mini and main frame computer systems.

Once software development is complete, the code can be completely evaluated and debugged in hardware with the Evakit-7720 Evaluation System. The Evakit provides true in-circuit real-time emulation of the SPI for debugging and demonstrating your final system design. Code may be down-loaded to the Evakit from a development system via an RS232 port using the EVA communications program. This program is available in executable form for ISIS-II systems and many CP/M, CP/M-86, and MS-DOS systems. The EVA communications source code is also available for adapting the program to other systems.

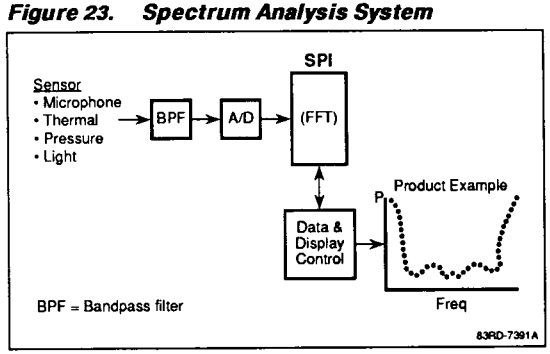
The Evakit also serves to program the 77P20, a full-speed EPROM version of the SPI. The 77P20 may also be programmed using DATA I/O "Unisite" and "2900 Programming Systems." Library routines for common DSP routines such as N-stage IIR (biquadratic) and FIR (transversal filters) are available on disk (free). Other hardware interface test routines as well as a Software Development Took Kit are also available.

Further operational details of the SPI can be found in the μPD77C20A/7720A/77P20 Signal Processing Interface Design Manual. Operation of the SPI development tools is described in the Absolute Assembler User Manual, the Simulator Operating Manual, and the Evakit- 7720 User's Manual.

SYSTEM CONFIGURATION

Figures 23, 24, 25, and 26 show typical system applications for the 77C20A/7720A/77P20 SPI.

Figure 23. Spectrum Analysis System



CP/M and CP/M-86 are registered trademarks of Digital Research Corp. ISIS-II is a registered trademark of Intel Corp. MS-DOS is a registered trademark of Microsoft Corp.

Figure 24. Analog-to-Analog Digital Processing System Using a Single SPI

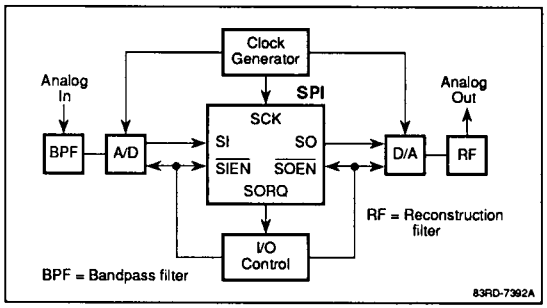


Figure 25. Signal Processing System Using Cascaded SPIs and Serial Communication

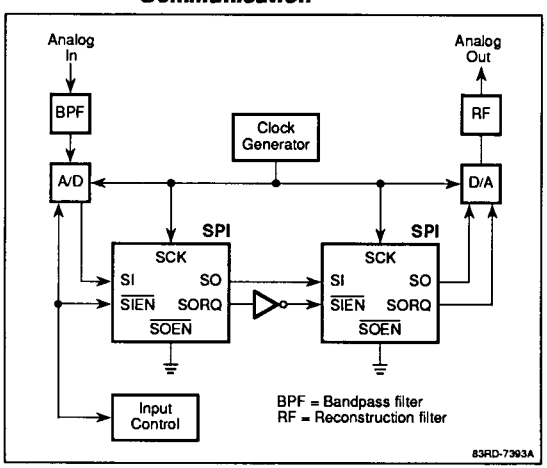
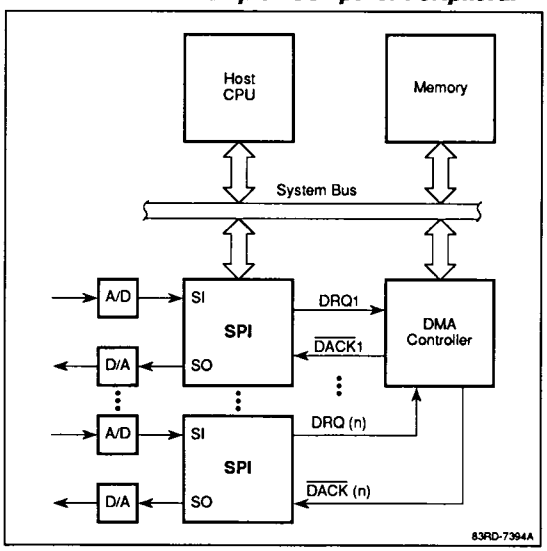


Figure 26. Signal Processing System Using SPIs as a Complex Computer Peripheral



3a