NEC
NEC Electronics U.S.A. Inc.
Microcomputer Division

μPD7720
SIGNAL PROCESSING INTERFACE (SPI)
TECHNICAL MANUAL

μPD7720

FABIO MONTORO
CLEARWATER
FL
USA.

μPD7720 Signal Processing Interface (SPI)
Technical Manual

uPD7720 Technical Manual
TABLE OF CONTENTS

LIST OF TABLES

TABLE NO.

LIST OF FIGURES

FIGURE NO.

1. SUMMARY DESCRIPTION

Fabricated in high-speed NMOS, the uPD7720 SPI is a complete 16-bit microcomputer on a single chip. ROM space is provided for program and data/coefficient storage, while the on-chip RAM may be used for temporary data, coefficients and results. Computational power is provided by a 16-bit Arithmetic/Logic Unit (ALU) and a separate 16 x 16-bit fully parallel multiplier. This combination allows the implementation of a "sum of products" operation in a single 250 nsec instruction cycle. In addition, each arithmetic instruction provides for a number of data movement operations to further increase throughput. Two serial I/O ports are provided for interfacing to codecs and other serially-oriented devices while a parallel port provides both data and status information to conventional uP for more sophisticated applications. Handshaking signals, including DMA controls, allow the SPI to act as a sophisticated programmable peripheral as well as a stand alone microcomputer.

APPLICATIONS

- Speech Synthesis and Analysis
- Digital Filtering
- Fast Fourier Transforms (FFT)
- Dual-Tone Multi-Frequency (DTMF) Transmitters/Receivers
- High Speed Data Modems
- Equalizers
- Adaptive Control
- Sonar/Radar Image Processing
- Numerical Processing (Calculation-intensive applications)

PERFORMANCE BENCHMARKS

Second Order Digital Filter (BiQuad)	2.25 us
SINE/COS of Angles	5.25 us
μ/A Law to Linear Conversion	0.50 us
FFT: 32-Point Complex	0.7 ms
64-Point Complex	1.6 ms

1.1 Features

Fast Instruction Execution - 250 ns/8 MHz CLOCK

16-bit Data Word

Multi-operation Instructions for Optimizing Program Execution

Large Memory Capacities

Program ROM	512 x 23 bits
Data/Coefficient ROM	510 x 13 bits
Data RAM	128 x 16 bits

Fast (250 ns/8 MHz) 16 x 16 bits Parallel Multiplier with 31-bit result

Four-level Subroutine Stack for Program Efficiency

Multiple I/O Capabilities

- Serial - Separate input and output (8 or 16-bit)
- Parallel - 8-bit bus
- DMA

Compatible with most Microprocessors, Including:

- uPD8080
- uPD8085
- uPD8086
- uPD780 (Z80™)

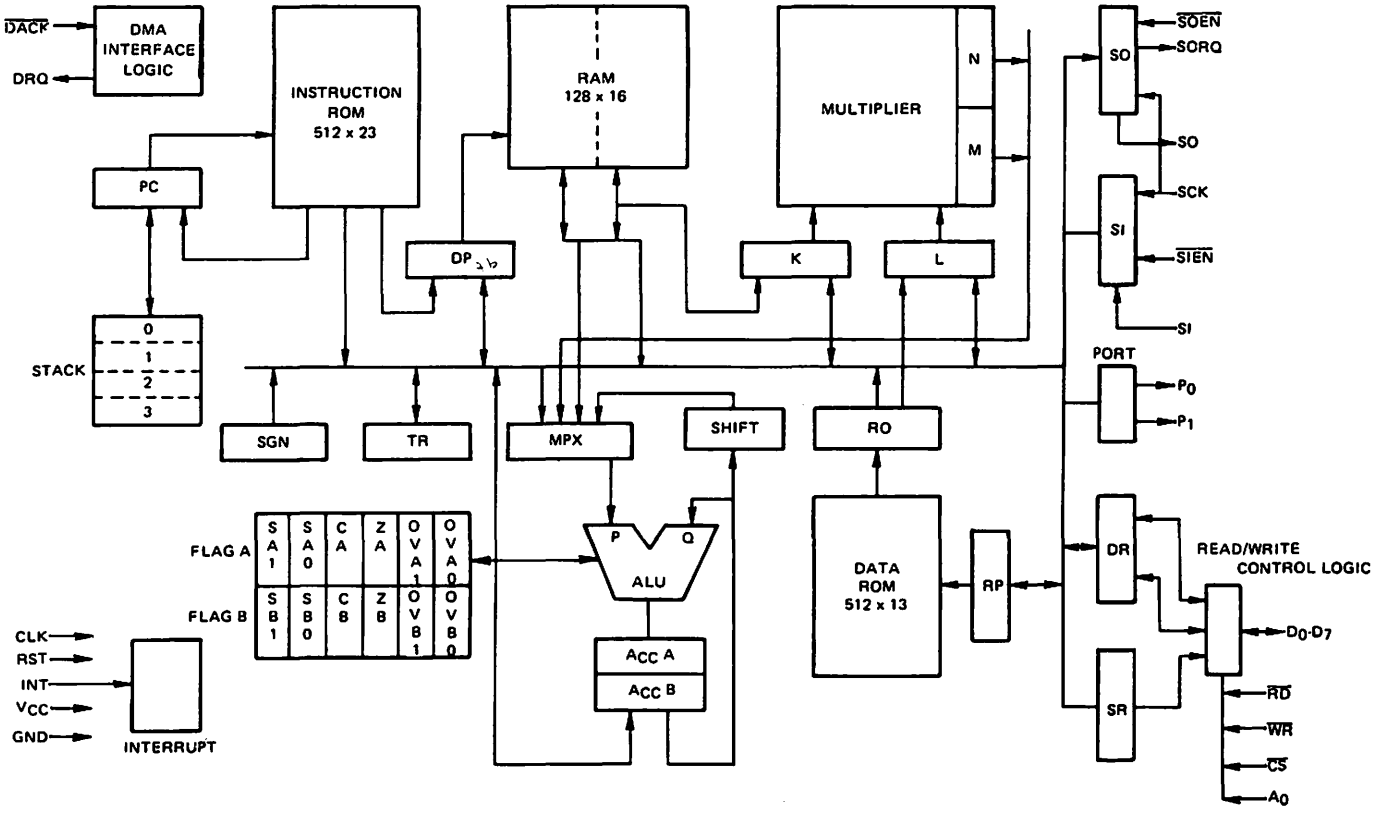
+5V Power Supply

NMOS Technology

28-pin DIP package

™ -- Trademark of Zilog Corporation

1.2 Block Diagram



1.3 Instruction Summary

The SPI has four (4) instructions. Two of these instructions, OP and RT,* are almost identical in that both execute the same operations, but RT also implements a subroutine/interrupt return when it completes its other operations.

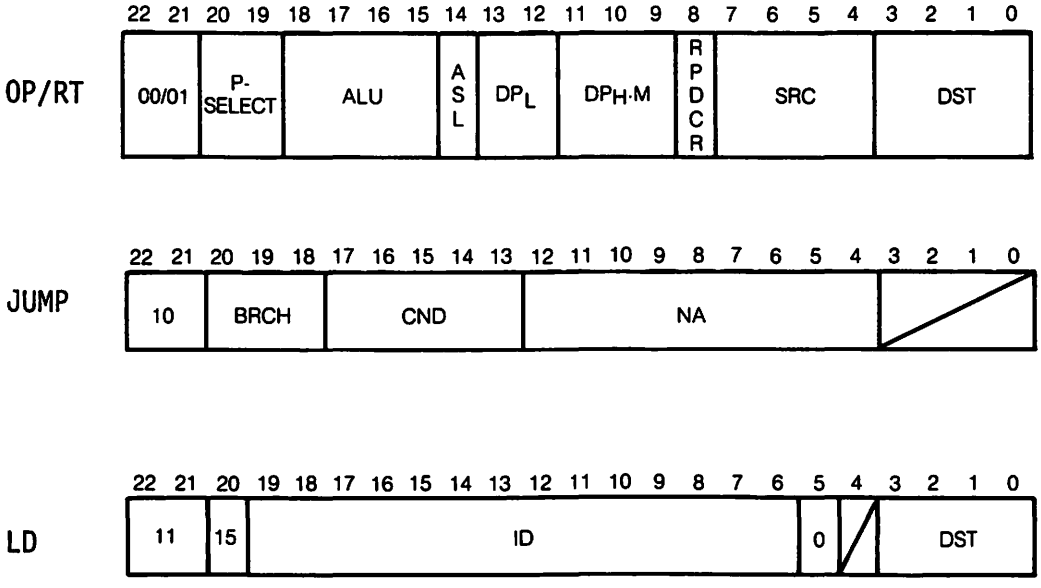
OP and RT can perform the following operations in a single instruction: implement an ALU operation on either accumulator, using an operator from one of four inputs; modify the lower four bits and upper three bits of the data RAM pointer; decrement the data/coefficient ROM pointer, and move data from one register to another. While this is happening, the two 16-bit multiplier input registers are multiplied and their 31-bit product is placed in the multiplier output registers.

* In the assembler, the RT instruction is treated as a variation of the OP instruction, not as a separate instruction.

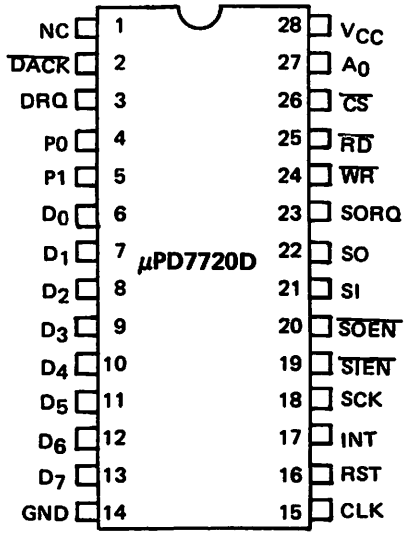
The third instruction is called the Jump instruction, and has three variations - unconditional jump, conditional jump (32 types), and subroutine call.

The fourth and last is the LD instruction which allows you to load a 16-bit immediate value to one of thirteen possible destination (DST) registers.

FIGURE 1.3 INSTRUCTION SUMMARY



2. PIN CONFIGURATION



2.1 Pin Identification

PIN	NAME	I/O	FUNCTION
1	NC	I	No Connection.
2	$\overline{DACK}$	I	DMA Request Acknowledge. Indicates to the μPD7720 that the Data Bus is ready for a DMA transfer. ($\overline{DACK} = \overline{CS} \bullet A_0 = 0$)
3	DRQ	O	DMA Request signals that the μPD7720 is requesting a data transfer on the Data Bus.
4,5	P_0, P_1	O	P_0, P_1 are general purpose output control lines.
6-13	D_0-D_7	I/O Tristate	Port for data transfer between the Data Register or Status Register and Data Bus.
14	GND		
15	CLK	I	Single phase Master Clock input.
16	RST	I	Reset initializes the μPD7720 internal logic and sets the PC to 0.
17	INT	I	Interrupt. A low to high transition on this pin will (if interrupts are enabled by the program) execute a call instruction to location 100H.
18	SCK	I	Serial Data Input/Output Clock. A serial data bit is transferred when this pin is high.
19	$\overline{SIEN}$	I	Serial Input Enable. This line enables the shift clock to the Serial Input Register.
20	$\overline{SOEN}$	I	Serial Output Enable. This pin enables the shift clock to the Serial Output Register.
21	SI	I	Serial Data Input. This pin inputs 8 or 16 bit serial data words from an external device such as an A/D converter.
22	SO	O	Serial Data Output. This pin outputs 8 or 16 bit data words to an external device such as an D/A converter.
23	SORQ	O	Serial Data Output Request. 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.
24	$\overline{WR}$	I	Write Control Signal writes the contents of data bus into the Data Register.
25	$\overline{RD}$	I	Read Control Signal. Enables an output to the Data Port from the Data or Status Register.
26	$\overline{CS}$	I	Chip Select. Enables data transfer with Data or Status Port with $\overline{RD}$ or $\overline{WR}$.
27	A_0	I	Selects Data Register for Read/Write (low) or Status Register for read (high).
28	V_{CC}		+5V Power

3. FUNCTIONAL DESCRIPTION

3.1 Instruction ROM

The 512 x 23 bits of mask-programmable instruction ROM are addressed by a 9-bit Program Counter which can be modified by an external reset, interrupt, call/jump, or return instruction.

3.2 Program Counter

The Program Counter (PC) is a 9-bit binary counter which functions as follows:

PC is incremented by 1 at each instruction fetch.
The contents of the address field (NA field) of any of the following instructions is transferred to PC when executed:

- JMP instruction
- Conditional jump instructions (if the condition is met)
- CALL instruction (and the PC is pushed on to the stack)

The content of the Stack is transferred to PC when an RT instruction (subroutine Return instruction) is executed.
The interrupt address (100H) is transferred to PC whenever an interrupt condition occurs, and the PC is pushed on to the stack.

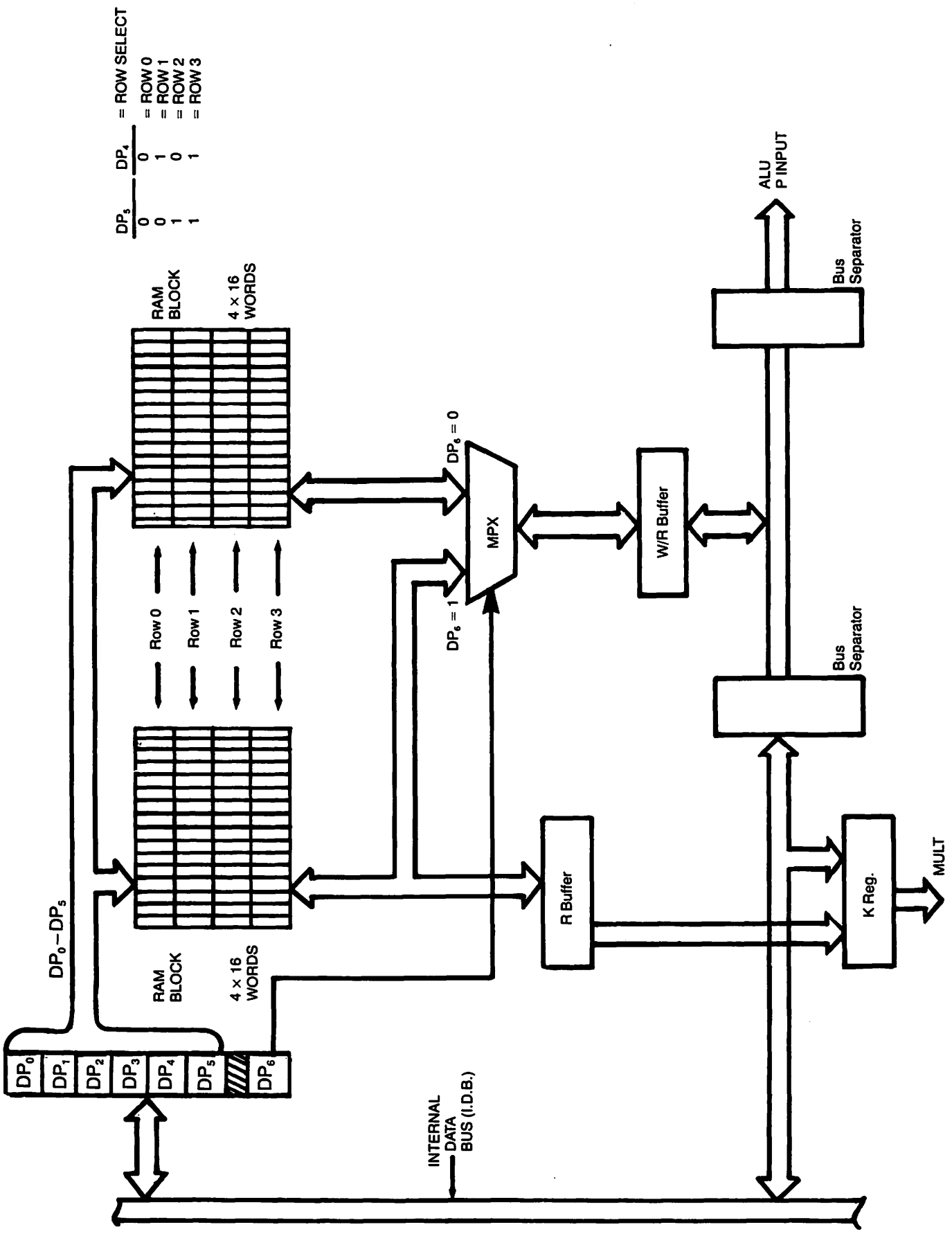
3.3 Stack

The SPI contains a 4-level program stack for efficient program usage and interrupt handling. The stack is a 4 x 9-bit LIFO register (last in first out). It stores the return addresses for subroutines and interrupts. The return address is read out of the stack and transferred to PC when the subroutine Return instruction is executed. You may not nest subroutines (or interrupts) more than four (4) levels because the return addresses are pushed out the bottom of the stack and lost.

3.4 RAM

The data RAM has 128 16-bit words and is addressed through a 7-bit Data Pointer (DP) register. The DP has addressing features that operate simultaneously with arithmetic instructions so that no added time is taken for address modification.

FIGURE 3.1 BLOCK DIAGRAM OF RAM, DP, AND PERIPHERALS



The Data RAM is best thought of as two blocks of memory, each with four rows of 16 words. The DP register's MS bit (DP_6) selects which block will be input/output from the MPX. DP_5 and DP_4 select the row within a block to be accessed and DP_3-DP_0 (DP_L) select the value within the row. In addition to (DP_6 = 1) being able to route its data through the MPX, the high order block may also send its data directly to the K register (Mult input register) via its own bus as part of an OP/RT instruction (@KLM = Destination).

3.5 Data Pointer (DP) Register

The Data Pointer is a seven-bit register that specifies the RAM address. The three higher bits are referred to as DP_H and the four lower bits as DP_L.

When data is loaded from the internal data bus to DP, the lower seven bits are input and the nine higher bits are ignored (IDB is 16-bit bus). When data is output from the DP to the internal data bus, the seven lower bits contain the DP value, and the nine higher bits of the data bus are filled with zeroes.

The Data Pointer can be modified as part of an OP/RT instruction*.

The three higher bits of the DP can be modified by being Xor'ed with the three bits in the DP_HM field of the OP/RT instruction (MØ through M7 in assembly language).

EXAMPLE:

DP/BIT	654	3210
DP =	000	XXXX
M1 = XOR	001	
DP =	001	XXXX

DP/BIT	654	3210
DP =	001	XXXX
M1 = XOR	001	
DP =	000	XXXX

The four lower bits of the DP can be modified by the DP_L field of an OP/RT instruction in the following ways:

*See Page 27, Assembly Language Instruction Format, Arithmetic MOV (OP/RT).

1. increment DP_L - MODULO 16 (0 - 15) = DPINC
2. decrement DP_L - MODULO 16 (15 - 0) = DPDEC
3. clear DP_L = DPCLR
4. no change (NOP) = DPNOP

You can modify both parts of the DP with OP/RT instruction simultaneously, but those operations are ignored if a DP Load from the internal data bus is selected as part of the same instruction (@DP = Destination).

The DP value resulting from modifications as part of an OP/RT instruction is not implemented until the end of the instruction cycle, and therefore does not affect the DP value used as part of that same instruction.

3.6 Data/Coefficient ROM (512 x 13 bits)

The Data/Coefficient ROM is organized in 512 words by 13 bits and is addressed by a 9-bit ROM pointer (RP) Register. You may use all ROM locations except addresses 0 and 1. This ROM is ideal for storing the necessary coefficients, conversion tables, and constants for processing.

The ROM's output buffer (RO) data may be routed two places, directly to the L register (special bus, @KLR=DST) or to the IDB (RO=SRC) as specified by an OP/RT instruction. In either case, the 13 bits are placed in higher order bits of the output word (16-bit word) and the low order 3 bits are always zero.

3.7 RP (ROM Pointer)

The ROM Pointer is a 9-bit register that specifies the Data ROM address. It is used in much the same manner as the DP register.

When data is input to the RP register from the internal data bus, the nine lower bits are stored and the seven higher bits are ignored. When data is output from the RP register to the internal data bus, the RP data fills the lower nine bits of the data bus, and zeroes are output to the seven higher bits.

You can decrement the RP register as part of an OP/RT instruction by using the RPDCR field. The RP value that results from modifications as part of an OP/RT instruction is not implemented until the end of the instruction cycle. The RP modification is ignored if a value is being input from the IDB as part of the same instruction (@RP = DST).

3.8 Multiplier

This is a fully parallel multiplier that uses the secondary Booth algorithm.

It multiplies two 16-bit words (taken from the K and L registers) to produce a 31-bit solution expressed in the two's complementary form (sign bit) + (30-bit data).

Of the 31 bits, the sign bit and the 15 higher bits are output to the M register, and the 15 lower bits are output to the N register (0 is placed in the LSB of the N register). A multiply is done every instruction cycle.

3.9 K Register

The K Register is a 16-bit register that holds either the multiplier or the multiplicand to be input to the multiplier. You can fill the K register with data that is output to the internal data bus or sent via the separate high RAM bus as specified by the destination field of an OP/RT instruction (@K = Destination or @KLM = Destination). You can also output the contents of the K register to the internal data bus (K = source). When data is placed into the K register, it is automatically accessed by the multiplier so that the multiplier solution is available on the next instruction cycle.

3.10 L Register

The L Register is a 16-bit register that holds either the multiplier or the multiplicand to be input to the multiplier. Data output to the internal data bus or data sent via the separate ROM bus can be put in the L register as specified by an OP/RT instruction (@L = Destination or @KLR = Destination). You may output the contents of the L register to the internal data bus (L = source). The Multiplier L register operation is the same as the K register.

3.11 M and N Registers

The M and N registers are the output registers for the multiplier. Output from the multiplier is in 2's complementary form. The sign bit and the 15 higher bits of data are placed in the M register, and the 15 lower bits of data are placed in the N register. A zero is placed in the LSB of the N register. This is shown below in Figure 3.2.

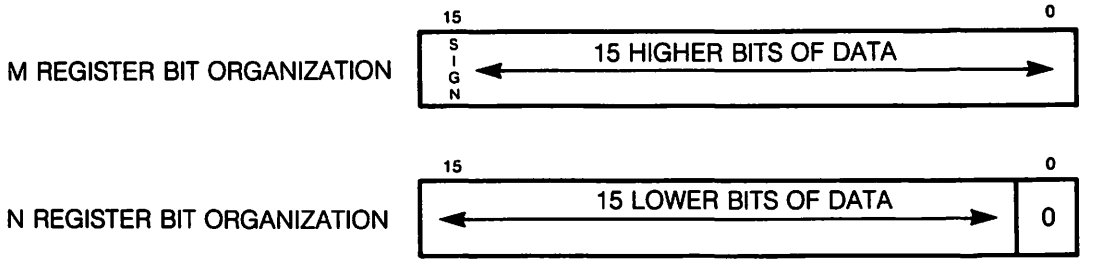


FIGURE 3.2 M AND N REGISTER BIT ORGANIZATION

If both the multiplier and the multiplicand are the maximum negative value (8000H), the multiplier overflows and the output is 80000000H.

The outputs of the M and N registers are connected to the P input MPX of the ALU. No connection to the Internal Data Bus is made from these registers.

3.12 ALU

The ALU is a 16-bit two's complement unit capable of executing sixteen distinct operations on virtually any internal register. The operations are:

NOP	Increment Acc
OR/AND/XOR	Complement Acc
Subtract	1-bit Right Shift
ADD	1-bit Left Shift
Subtract with borrow	2-bit Left Shift
ADD with Carry	4-bit Left Shift
Decrement Acc	8-bit Exchange

The ALU outputs from these operations are stored in the same accumulator that was operated on (ASL).

Note that in the case of a subtract that Q-P is performed (i.e. Acc - X). (See Figure 3.3)

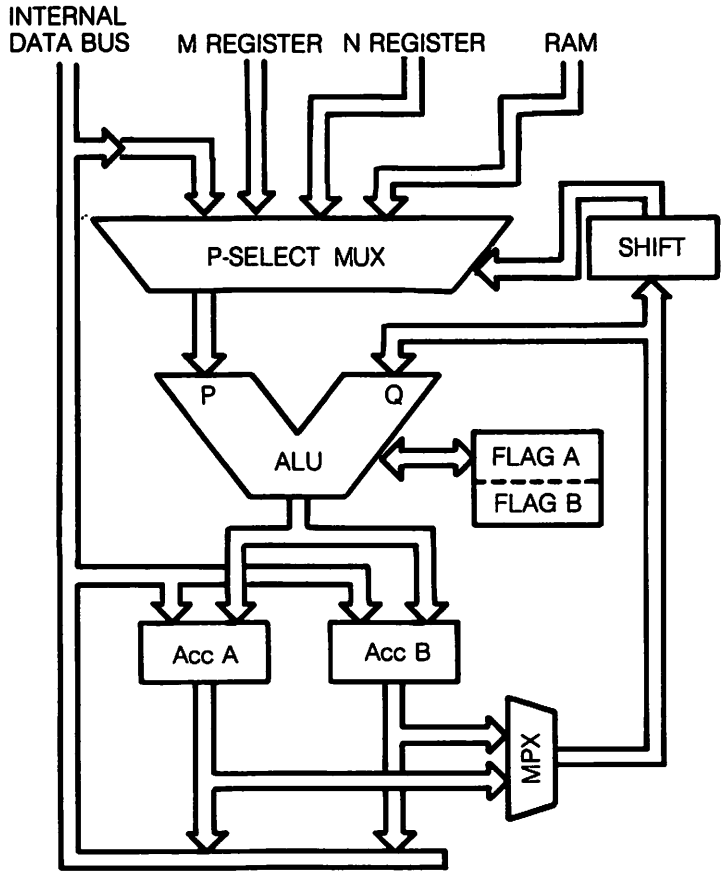


FIGURE 3.3 ALU AND PERIPHERALS BLOCK DIAGRAM

3.13 Accumulators (AccA, AccB)

The accumulators are a pair of 16-bit registers that store the results of ALU operations. Each accumulator has its own set of flags that are updated after each arithmetic instruction except NOP.

Whether the ALU output goes to Acc A or Acc B is specified by the ASL bit of each OP/RT instruction. The contents of both Acc A and Acc B can be output to the internal data bus and can also be input to the Q input of ALU or to SHIFT, as needed by the particular ALU operation.

The output from the internal data bus to Acc A or Acc B, or conversely, from the accumulators to the internal data bus, is controlled by the SRC and DST fields of the OP/RT instruction. You may move data from a source to a destination during an arithmetic operation (see Figure 1.2 - Block Diagram). If the accumulator being used for an ALU operation is specified as a destination (@A or @B) for IDB information in the same instruction, the ALU operation is ignored, and the IDB data is latched to that accumulator.

3.14 SHIFT

You can implement five different shift operations on 16-bit data in Acc A or Acc B.

1) 1-bit Right Shift

The LSB is placed in the Carry bit of the Acc selected by the ASL bit of an OP/RT instruction. The sign bit is copied to the next lower bit. This operation is equivalent to a two's complement division operation.

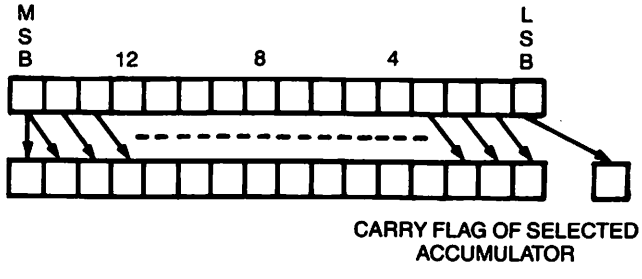


FIGURE 3.4 1-BIT RIGHT SHIFT

2) 1-bit Left Shift

The Carry flag of the Acc not selected by the ASL bit is put into the LSB, and the MSB of the selected Acc goes to its own Carry flag bit.

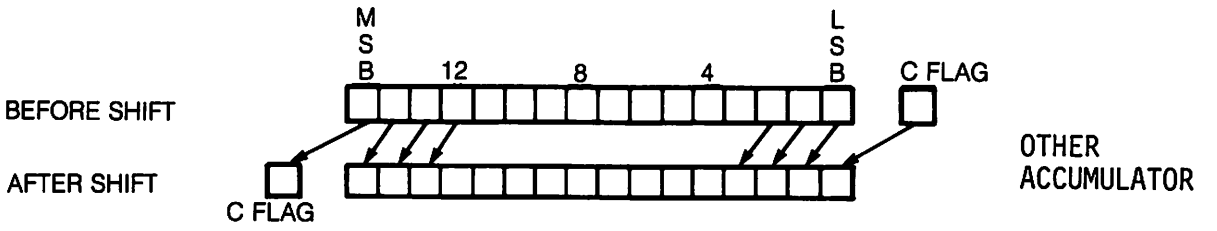


FIGURE 3.5

3) 2-Bit Left Shift

The two lower bits are filled with one's, and the two highest bits are discarded.

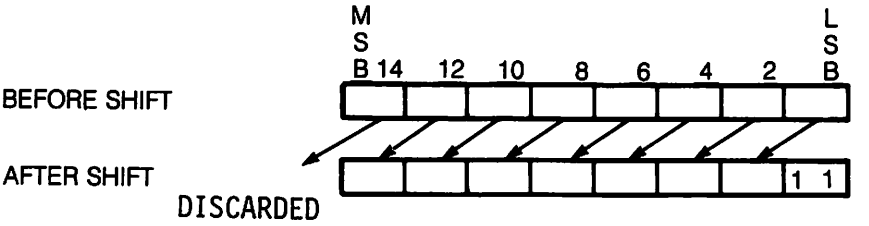


FIGURE 3.6

4) 4-Bit Left Shift

The four lower bits are filled with one's, and the four highest bits are discarded. This operation and the preceding one are ideal for generating RP addresses, because modification of RP as part of the OP/RT instructions is a decrement or NOP.

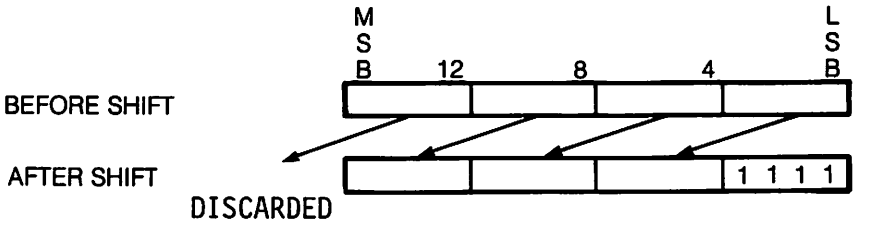


FIGURE 3.7

5) 8-Bit Exchange

The eight higher bits are exchanged with the eight lower bits.

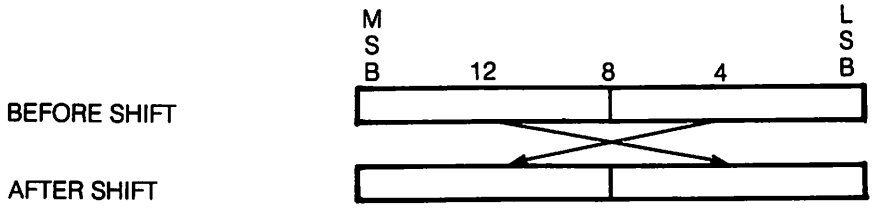


FIGURE 3.8 8-BIT EXCHANGE

3.15 Flag Registers (See ALU function and Flag operation - Table 4.2)

The uPD7720 has two flag registers: Flag A and Flag B. Flag A is a six-bit register that indicates the results and latest status of Acc A. Flag B performs the same function for Acc B. Each flag register consists of the following flag bits shown in figure 3.9:

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

FIGURE 3.9 ACCUMULATOR FLAG BITS

1) CA, CB (Carry)

These flags store the carry of operational results.

Carry from the ALU is stored after one of the following operations: SUB, ADD, SBB, ADC, DEC, or INC.

After a 1-bit Right or 1-bit Left Shift, the LSB or MSB is stored, respectively.

In operations other than those above, this flag is set to zero. The last previous status is unchanged after a NOP.

2) ZA, ZB (Zero)

This flag is set to one if the data stored in the Acc is zero, and is set to zero if the contents of the Acc is nonzero. The last previous status is unchanged after a NOP.

3) SAO, SBO (Sign)

In operations other than NOP, the MSB of the ALU contents is placed in this flag. The last previous status is unchanged after a NOP.

4) OVAO, OVBO (Overflow)

This flag stores the logical XOR of the carry from the 15th bit (MSB (SIGN)) and 14th bit of the Acc after one of the following operations: SUB, ADD, SBB, ADC, DEC, or INC. The last previous status is unchanged after a NOP. This flag is set to zero after any operation other than the previous.

5) OVA1, OVB1 (Overflow)

This flag promotes more efficient processing of the overflow resulting from three consecutive additions.

This flag is set to one if the overflow flag (OVAO, OVBO) was set an odd number of times after the three consecutive additions. It is set to zero if the overflow flag was set an even number of times. If the overflow was set in the order 1-0-1, then:

OVA1 (OVB1) = 1, if SA1 (SB1) = SAO (SBO), or

OVA1 (OVB1) = 0, if SA1 (SB1) is not equal to SAO (SBO)

This applies to SUB, ADD, SBB, ADC, DEC, and INC. The previous status is unchanged after a NOP. This flag is set to zero after operations other than the above.

6) SA1, SB1 (Sign)

This flag, used with OVA1 (OVB1) aids in overflow processing. Direction of an overflow (positive or negative) can be judged with this flag.

If the overflow status (OVA1, OVB1) equals zero as you begin the next ALU operation (SUB, ADD, SBB, ADC, DEC, INC), this flag is set equal to the resultant sign bit (SAO, SBO) after that operation.

If the overflow status (OVA1, OVB1) equals one as you begin the next operation, the value of this flag is unchanged.

3.16 Sign Register (SGN)

When OVA1 (or OVB1) is set, the SA1 (or SB1) bit will hold the corrected sign of the overflow. The SGN Register will use SA1 (SB1) to automatically generate saturation constants 7FFFH(+) or 8000H(-) to permit efficient limiting of a calculated value. The SGN register will hold the current saturation value for up to three consecutive additions in a row.

3.17 Temporary Register (TR)

The TR register is a 16-bit temporary storage register on the internal data bus.

3.18 SR (Status Register)

The Status Register is a 16-bit register that contains the information required to handle data transfers with external devices. Of the 16 bits, only the 8 MSB may be read by an external processor. Any attempt to write into bits that are not defined or under SPI control are ignored.

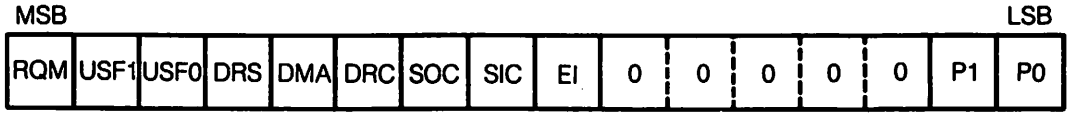


FIGURE 3.10 STATUS REGISTER BITS

1) RQM (Request for Master)

This flag bit is used for data transfers between the data register and the host processor in non-DMA mode.

A read or write from the internal data bus to the data register sets this flag to one. An external 8- or 16-bit read or write resets this flag to zero. The status of this bit remains unchanged when DMA = 1 (DMA mode).

When using DRNF in the SRC field of an OP/RT instruction, this flag is not set, even though data is read from the data register to the internal data bus.

2) USF1, USF0 (User Flags)

These are general purpose flags that can be read by the host processor for user-defined signaling.

3) DRS (Data Register Status)

This bit indicates the status of data transfers when an external device views the data register as a 16-bit register (DRC=0).

The data bus connecting to external devices has only 8 bits; therefore, if the data register is operating in 16 bit mode, data must be transferred in two steps. This bit turns to one after the first transfer, then to zero after the second transfer (low byte then high byte).

This flag remains at zero during 8-bit transfers (DRC=1).

4) DMA (Direct Memory Access)

This bit determines the method by which data can be transferred between an external device and the data register.

When this bit is 1, data transfers are made via DMA using DRQ and $\overline{\text{DACK}}$. Note that $\overline{\text{DACK}}$ is the same as setting $\overline{\text{CS}}$ and AØ = Ø. When this bit is a Ø, parallel data transfer is handed by controlling $\overline{\text{CS}}$ and AØ directly by the controlling device.

5) DRC (Data Register Control)

This bit controls whether the data register is used in double byte (16 bit) or single byte (8 bit) mode. When this bit is 0, the data register is set for sixteen bit transfers. When this bit is 1, the data register is set for eight bit transfers.

6) SOC (Serial Output Control)

This bit specifies the length of serial data to be output from the SO pin. When this bit is 0, all 16 bits of the data word are output. When this is a 1, only 8 bits of the data word are output (the low 8 bits of the SO register).

7) SIC (Serial Input Control)

This bit specifies the length of serial data to be input at the SI pin. When this bit is 0, the data input is 16 bits. When this bit is 1, the data input is 8 bits.

8) EI (Enable Interrupt)

When this bit is 1, interrupts are accepted. When an interrupt is received, this bit is automatically reset to reject subsequent interrupts. When this bit is set to zero, interrupts are not accepted or remembered by the SPI.

9) P1, P0

These bits correspond to the output ports P1 and P0. All values input here are output as they are (1 corresponding to a high output value).

3.19 Parallel I/O Port

The 8-bit parallel I/O port may be used for transferring data or reading the status of the SPI. Data transfer is handled through a 16-bit Data Register (DR) that is software-configurable for single or double byte transfers.

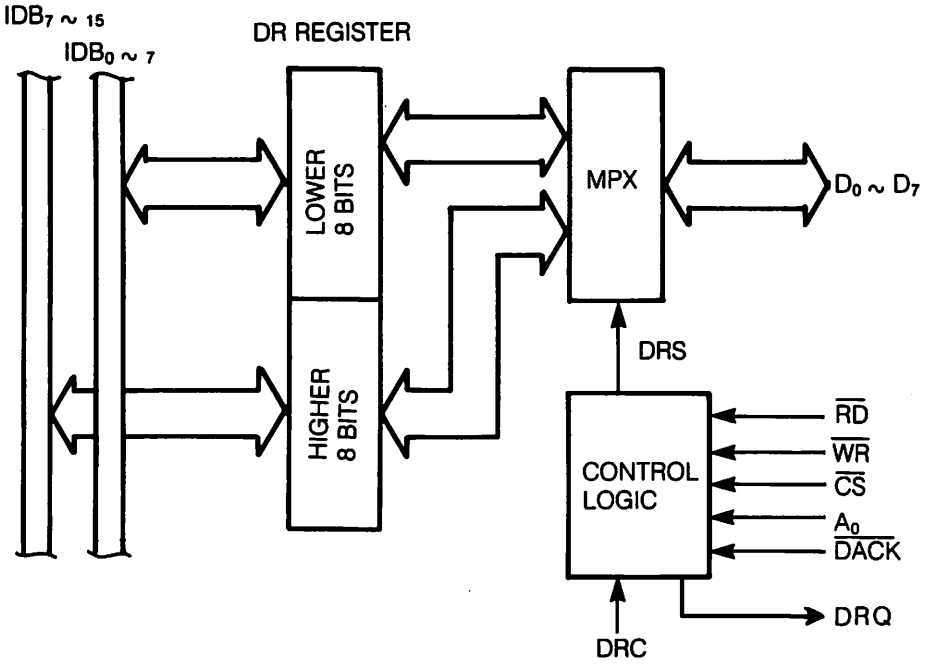


FIGURE 3.11
DATA REGISTER & PERIPHERAL'S BLOCK DIAGRAM

3.20 DR (Data Register)

This 16-bit register is used to transfer data between the SPI and external devices. The data bus, D_0-D_7 is eight bits wide. Sixteen-bit data is transferred in two steps (low byte then high byte) even though a transfer is made only once internally.

If the DRC bit of the status register defines the data register as eight bits, only the lower eight bits of the data register are transferred to an external device.

3.21 Parallel READ/WRITE Operation

The Read/Write control logic transfers data to or from the SPI, depending on the status of the external control signals, $\overline{CS}$, Ao, $\overline{WR}$, $\overline{RD}$, or $\overline{DACK}$. The condition of $\overline{DACK}$ = 0 is equivalent to Ao = $\overline{CS}$ = 0.

Data is sent from the SPI or to the SPI in low byte, high byte order.

Whether the eight MSBs or eight LSBs are being sent/received is denoted by the status of the DRS bit of the status register.

$\overline{CS}$	A_0	$\overline{WR}$	$\overline{RD}$	Operation
1	X	X	X	Internal operation is not affected: D_0-D_7 are kept under a high impedance
X	X	1	1	
0	0	0	1	Data of D_0-D_7 are latched to DR register*
0	0	1	0	Contents of DR register are output to D_0-D_7*
0	1	0	1	Inhibited
0	1	1	0	8 higher bits of SR register are output to D_0-D_7
0	X	0	0	Inhibited

*Whether 8 higher bits or lower bits of DR register are assigned depends on the status of the DRS bit of the SR register.

TABLE 3.1 R/W CONTROL LOGIC

3.22 DMA Interface Logic

DMA data transfers are controlled by DRQ and $\overline{DACK}$, and may take place only when the DMA bit of the status register is set to 1.

1) DRQ Operation

DRQ is a DMA request for the host processor or the DMA controller to begin the DMA transfer. When data is transferred between the data register and the internal data bus, DRQ is set to 1 and output. DRNF in the SRC field of an OP/RT instruction prevents DRQ from being set even if a transfer from the data register has been made.

When DRC = 0, DRQ is reset at the second fall of $\overline{\text{DACK}}$ to read/write data (16 bit).

When DRC = 1, DRQ is reset at the only fall of $\overline{\text{DACK}}$ (8 bit).

Note: The RQM bit is not affected by data transfers in DMA mode.

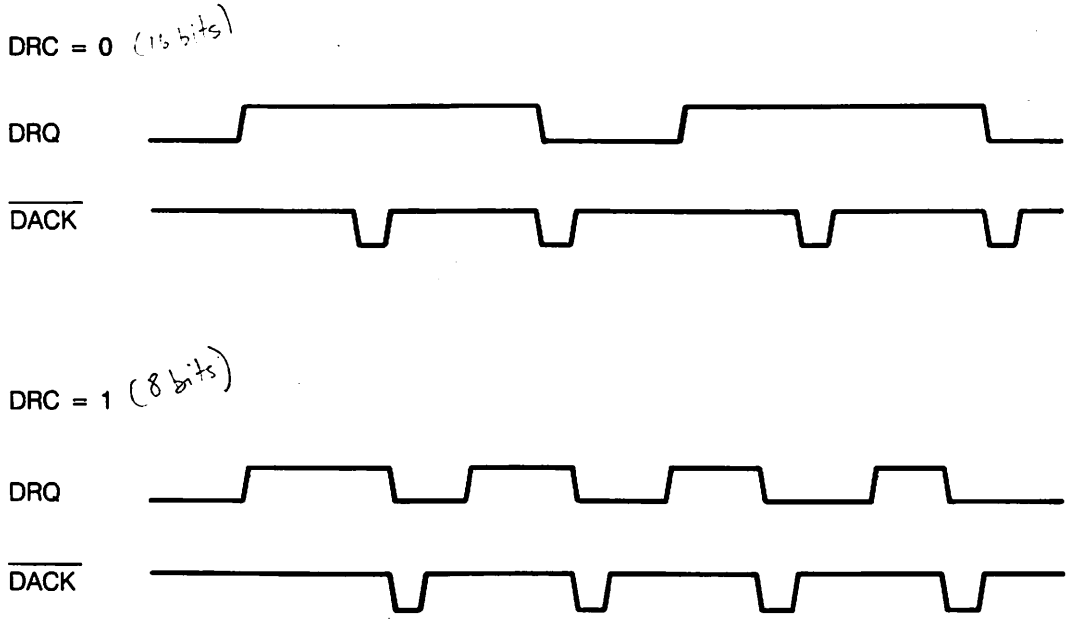


FIGURE 3.12 DRQ AND $\overline{\text{DACK}}$ TIMING

3.23 SI Register (Serial Input)

The SI register is the register in which serial input data is latched. It performs the following functions:

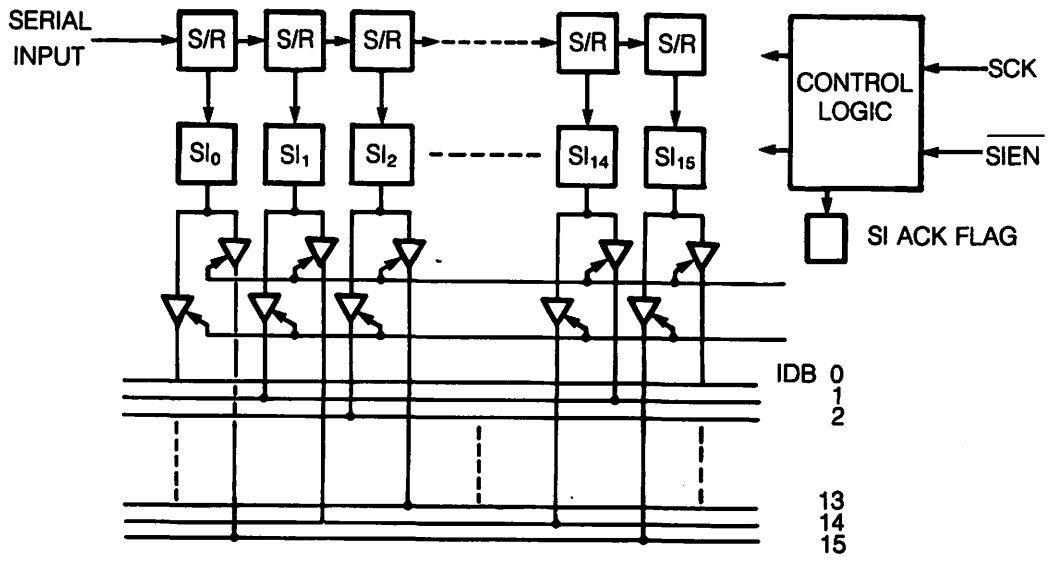


FIGURE 3.13 SERIAL INPUT BLOCK DIAGRAM

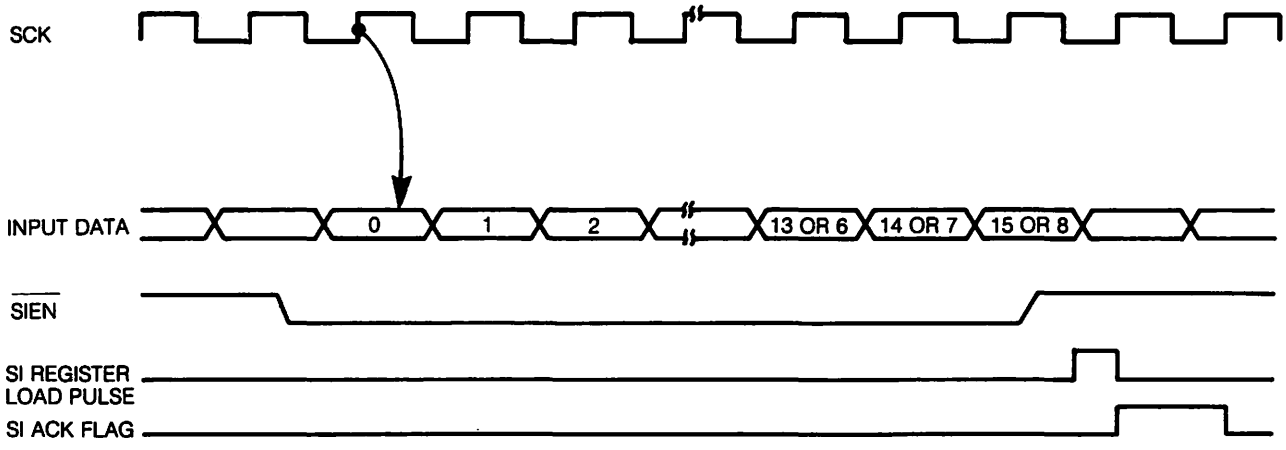


FIGURE 3.14 SERIAL INPUT TIMING

1) Serial data should be synchronized with the shift clock, such that the data may be latched at the rising edge of the shift clock.

2) Serial data is converted to parallel data by an internal shift register.

3) Shift register data is transferred to the SI register when the number of bits shifted equals the length specified by the SIC bit of the status register.

4) An internal flag (SIACK flag) is set when the data is latched to the SI register, and reset when data is read. There are conditional jump instructions that test the status of SIACK.

5) The parallel data can be read out of the SI register into the internal data bus by an OP/RT instruction (SIM or SIL = SOURCE).

6) As the parallel data is read out of the SI register, the SIACK flag is reset.

7) Two field specifications of the OP/RT instructions are available to read any parallel data from the SI register to the internal data bus. One specification outputs the SI register's MSB to the MSB of the IDB (Normal order); the other instruction outputs the LSB to the MSB (bit reversed order).

8) If the SIC bit of the status register is 1 (8 bit mode) and data is shifted in MS bit first, the SIM source specification should be used to load the data to the IDB in normal order. Zeros will be placed in the high 8 bits of the IDB.

9) The input of serial data to the shift register and the output of parallel data from the SI register may be performed independently. This allows you to input serial data of 8 or 16 bits consecutively.

10) If data is not read from the SI register before the next length of bits has been shifted in, the previous data in the register will be lost.

3.24 SO Register (Serial Output)

The SO register outputs serial data. It performs the following functions:

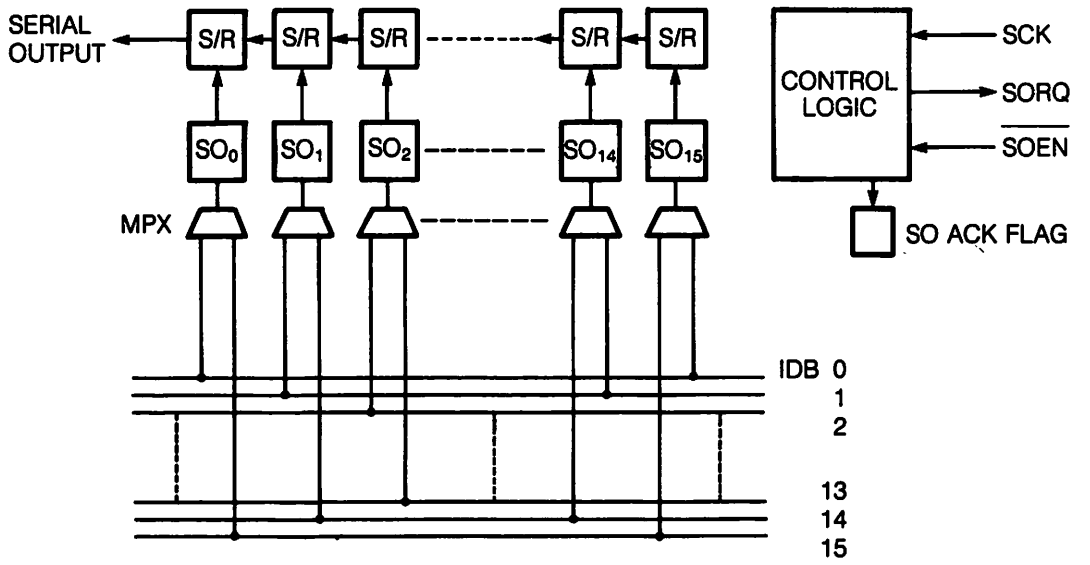


FIGURE 3.15 SERIAL OUTPUT BLOCK DIAGRAM

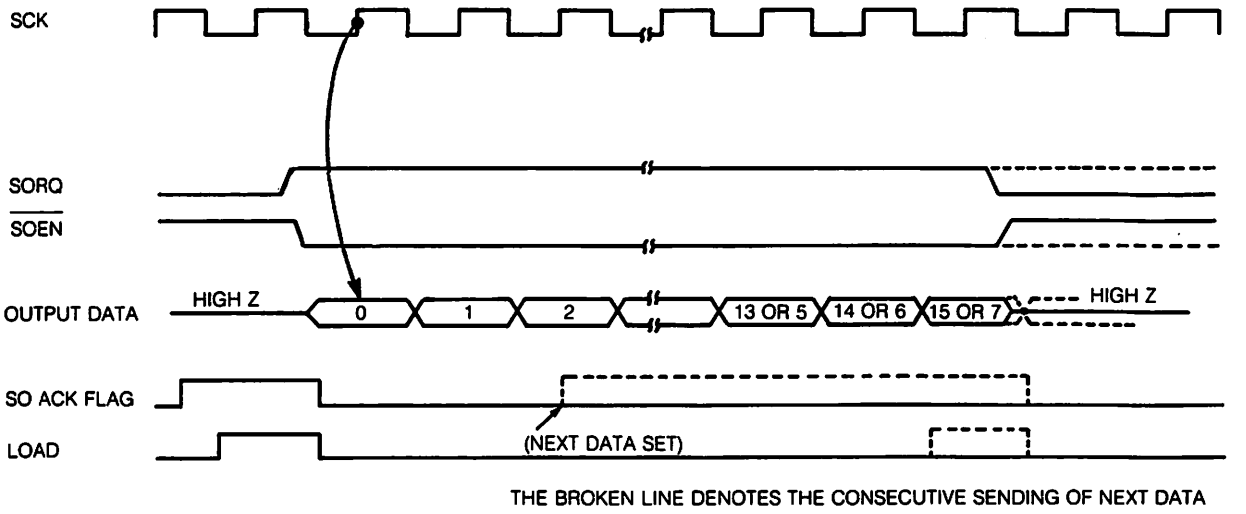


FIGURE 3.16 SERIAL OUTPUT TIMING

1) Data being output is loaded in the SO register (16 bits parallel) from the internal data bus.

2) An internal flag (SOACK flag) is set when data is written to the SO register.

3) Output data is transferred to the shift register from the SO register. If the shift register is busy, the data transfer is held until it is available.

4) SORQ is set to 1 when the data is transferred to the S/R informing the external device that data is available.

5) The serial data transfer starts if $\overline{SOEN}$ is input after data is sent to the shift register, i.e. if SORQ = 1.

6) The SO pin is held high if SORQ = 1 and $\overline{SOEN}$ = 1 or SORQ = 0 and $\overline{SOEN}$ = 0.

7) When output data is transferred from the SO register to the shift register, the SOACK flag is reset, indicating that the SO register is ready for the next word of data.

8) After sending the amount of bits specified by SOC, and if no data is available in the SO register, SORQ is set to Ø, instructing the system to wait for next serial output data.

9) Serial data is clocked out synchronously with the falling edge of the SCK, and is held until after the rising edge of SCK.

10) Like the SI register, you have the choice of sending the SO register data in normal or bit reversed order.

11) If the SOM destination is used to write data to the SO register, the MS bit of the 16 bit word written into the register is shifted out first, regardless of whether the SO register is in 8 or 16 bit mode.

3.25 Interrupts

Interrupts are processed in the following manner:

1) If the EI bit of the status register is 1, when a rising edge is sensed on the INT pin, the present instruction is finished. A NOP and JMP instruction to location (1ØØH) are implemented.

2) The return address is pushed onto the stack (during NOP).

3) All rising edges of interrupt line are ignored if the EI bit is set to zero (which occurs after an interrupt).

4) Interrupt will reset the EI bit to zero and must be set under program control to accept another interrupt.

5) After an interrupt is accepted, the INT pin should be reset low before the next interrupt; otherwise, only the first interrupt is accepted, even if the EI bit is high (edge sensitive).

4. INSTRUCTIONS

The SPI has three types of instructions, all one word and composed of 23 bits. You can identify each instruction by the code in the OP field (Bits 22 & 21).

4.1 OP/RT Instruction

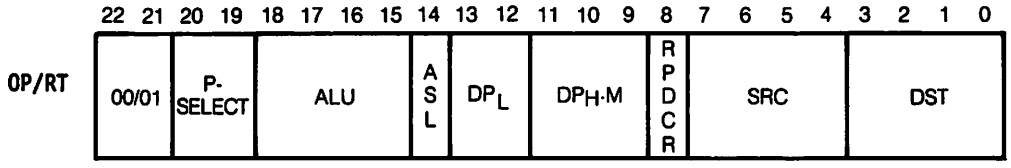


FIGURE 4.1 OP/RT INSTRUCTION FORMAT

FIGURE 4.2 ASSEMBLY LANGUAGE INSTRUCTION FORMAT

ARITHMETIC MOV ("OP/RT")

IN PARALLEL:

* TRANSFER DATA VIA THE INTERNAL DATA BUS

* ADD PRESENT CONTENTS OF MULTIPLIER OUTPUT REGISTER M TO ACC

* DATA RAM AND ROM POINTER MODIFICATION

* SUBROUTINE RETURN

EXAMPLE:

```
* OP    MOV     @ TR, MEM   /* RAM(DP) ⟶ TR                          DESTINATION, SOURCE */
       ADD     ACCA, M     /* M + ACCA ⟶ ACCA                       P-SELECT, ASL */
       DPINC               /* INCREMENT DATA RAM POINTER LOW (DPL)  DPL */
       RPDEC               /* DECREMENT DATA ROM POINTER (RP)       RPDEC = 1 */
       M1                  /* MODIFY RAM DPH                        DPMH = ØØ1 */
       RET                 ; /* SUBROUTINE (OR INTERRUPT) RETURN     OP CODE = Ø1 */
```

*NOTE: 1) ALL 6 FUNCTIONS EXECUTE IN ONE INSTRUCTION CYCLE.

2) ALL DP & RP MANIPULATION IS DONE AT END OF INSTRUCTION CYCLE.

3) THIS IS ACTUALLY AN RT INSTRUCTION BECAUSE RET IS USED. WITHOUT RET THIS WOULD BE AN OP INSTRUCTION.

This instruction performs the operations specified by the eight fields and the two bit OP code. It is used for arithmetic operations, data transfers, and subroutine returns.

When this is an OP instruction (OP Field-00), the program counter holds the current address plus one as the next address. When this is an RT instruction (OP Field-01), the program counter is set with the value at the top of the LIFO stack at the end of the instruction cycle.

4.1.1 P-Select Field

This field selects the source for P input of the ALU. This input may come from RAM, the internal data bus, the M Register, or the N Register for most ALU operations. For the INC, DEC, Complement Acc, and Shift operations, either an immediate value or SHIFT is used in the P input.

	P-SELECT FIELD		
MNEMONIC	D_{20}	D_{19}	INPUT
RAM	0	0	RAM
IDB	0	1	Internal Data Bus
M	1	0	M Register
N	1	1	N Register

TABLE 4.1 P-SELECT FIELD

4.1.2 ALU Field

This field specifies the ALU function according to the table below. Note that all results from an ALU operation are left in the accumulator that was operated on (ASL bit).

Mnemonic	ALU Field D_{18}	D_{17}	D_{16}	D_{15}	ALU Function	Flag A / Flag B	Flags Affected 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	‡	Ø	‡	Ø	Ø
AND	0	0	1	0	AND		X	‡	Ø	‡	Ø	Ø
XOR	0	0	1	1	Exclusive OR		X	‡	Ø	‡	Ø	Ø
SUB	0	1	0	0	Subtract		‡	‡	‡	‡	‡	‡
ADD	0	1	0	1	ADD		‡	‡	‡	‡	‡	‡
SBB	0	1	1	0	Subtract with Borrow		‡	‡	‡	‡	‡	‡
ADC	0	1	1	1	Add with Carry		‡	‡	‡	‡	‡	‡
DEC	1	0	0	0	Decrement ACC		‡	‡	‡	‡	‡	‡
INC	1	0	0	1	Increment ACC		‡	‡	‡	‡	‡	‡
CMP	1	0	1	0	Complement ACC (1's Complement)		X	‡	Ø	‡	Ø	Ø
SHR1	1	0	1	1	1-bit R-Shift		X	‡	‡	‡	Ø	Ø
SHL1	1	1	0	0	1-bit L-Shift		X	‡	‡	‡	Ø	Ø
SHL2	1	1	0	1	2-bit L-Shift		X	‡	Ø	‡	Ø	Ø
SHL4	1	1	1	0	4-bit L-Shift		X	‡	Ø	‡	Ø	Ø
XCHG	1	1	1	1	8-bit Exchange		X	‡	Ø	‡	Ø	Ø

‡ May be affected, depending on the results
– Previous status can be held
Ø Reset
X Indefinite

TABLE 4.2 ALU FUNCTION AND FLAG OPERATION

1) NOP (No Operation)

No operation is made by the ALU. The Acc Flags are unchanged. All of the ALU functions that follow are equivalent to a NOP if the accumulator being operated on is used as a destination for a move as part of the same instruction (@A or @B = DST).

2) OR/AND/XOR

These operations are executed between the input selected by the P-Select field and one of the accumulators. The ASL bit chooses the Acc to be operated on.

3) SUB (Subtract)

This operation subtracts the input selected in the P-Select field from the specified accumulator and leaves the result in that accumulator. The borrow input to the lowest bit is zero (Acc - P-Select input).

4) ADD

This operation adds the input selected in the P-Select field to the specified accumulator. The carry input to the lowest bit is zero.

5) SBB (Subtract with Borrow)

This operation subtracts the input selected in the P-Select field from the specified accumulator. The Borrow (Carry) flag of Flag register not selected by the ASL bit is input as a borrow to the lowest bit. For example, if Acc A is selected, the Borrow (Carry) of Flag B is used as the borrow.

6) ADC (Add with Carry)

This operation adds the input selected in the P-Select field to the specified accumulator. The Carry flag of the Flag register not selected by the ASL is input as a carry to the lowest bit. For example, if Acc A is selected, the Carry (CB) of Flag B is added.

7) DEC (Decrement Acc)

This operation subtracts one from the accumulator value specified by the ASL bit.

8) INC (Increment Acc)

This operation adds one to the accumulator value specified by the ASL bit.

9) CMP (Complement Acc)

This operation takes the one's complement of the accumulator value specified by the ASL bit.

10) SHR1 (1-bit Shift Right)

This operation shifts the contents of the specified accumulator one bit to the right. A sign bit is entered in the MSB-1 (bit 14) and the LSB is set to the carry flag of the Acc selected. For example, if Acc A is selected, the LSB of Acc A before the shift is input to the carry flag of A (CA).

11) SHL1 (1-bit Shift Left)

This operation shifts the contents of the specified accumulator one bit to the left. The Carry of the Flag register not selected is input to the LSB after the shift. For example, if Acc A is selected, the carry flag of B (CB) is input to the LSB. The MSB (bit 15) before the shift is set to its own carry flag.

12) SHL2 (2-bit Shift Left)

This operation shifts the contents of the specified accumulator two bits to the left. The two LSBs after the shift are set to 1. The two MSBs before the shift are discarded.

13) SHL4 (4-bit Shift Left)

This operation shifts the contents of the specified accumulator four bits to the left. The four LSBs after the shift are set to 1. The four MSBs before the shift are discarded.

14) XCHG (8-bit Exchange)

This operation exchanges the eight higher bits and eight lower bits of a selected accumulator.

4.1.3 ASL (Accumulator Select) Bit

This bit specifies whether Acc A or Acc B is used for the ALU operation.

The same Acc is specified for both ALU operation and result. Acc A is specified if this bit is set to zero, and Acc B is specified if this bit is set to one.

When Acc A is selected, Flag register A is also selected and Flag register B is selected when Acc B is selected.

This bit is ignored if NOP is specified in the ALU field.

If Acc specified by this bit is also specified in the DST field, this bit is ignored.

TABLE 4.3 ACCUMULATOR SELECT FIELD

Mnemonic	ASL Field D_{14}	Acc Selection
ACCA	0	Acc A
ACCB	1	Acc B

4.1.4 DP_L Field

This field specifies the modification of the four lower bits (DP_L) of the data pointer. The DP_L value, changed by the operation, is valid for the next instruction, and specifies the RAM address under that instruction. The table below shows the possible operations.

TABLE 4.4 DATA POINTER LOW FIELD

Mnemonic	DP_L Field D_{13}	D_{12}	DP_3-DP_0
DPNOP	0	0	No Operation
DPINC	0	1	Increment DP_L
DPDEC	1	0	Decrement DP_L
DPCLR	1	1	Clear DP_L

This field is ignored if DP is specified in the DST field. If DP is specified in the SRC field, the value of DP before change is output to the internal data bus.

The counter used for modification is a module counter (ØH comes after ØFH if DPINC).

4.1.5 DP_HM (DP_H Modify) Field

This field modifies the three higher bits (DP_H) of the data pointer by XORing it with the three bits of this field. The modified DP_H value is valid for the next instruction, and specifies the RAM address of that instruction.

TABLE 4.5 DATA POINTER HIGH MODIFICATION FIELD

Mnemonic	DP_H-M Field			Exclusive OR
	D_{11}	D_{10}	D_9	
* M0	0	0	0	(DP_6 DP_5 DP_4) ¥ (0 0 0)
M1	0	0	1	DP_6 DP_5 DP_4 ¥ (0 0 1)
M2	0	1	0	DP_6 DP_5 DP_4 ¥ (0 1 0)
M3	0	1	1	DP_6 DP_5 DP_4 ¥ (0 1 1)
M4	1	0	0	DP_6 DP_5 DP_4 ¥ (1 0 0)
M5	1	0	1	DP_6 DP_5 DP_4 ¥ (1 0 1)
M6	1	1	0	DP_6 DP_5 DP_4 ¥ (1 1 0)
M7	1	1	1	DP_6 DP_5 DP_4 ¥ (1 1 1)

* NO CHANGE

As shown in Table 4.5, DP6, DP5, and DP4 are XORed with the value in this field, and the result is input into DP6-4. If you do not wish to modify the DP_H value, you must set each bit of this field to zero (MØ). This field is ignored if DP is specified in the DST field. The DP value before the change is output to the internal data bus if DP is specified in the SRC field.

4.1.6 RPDCR Bit (RP Decrement)

When this bit is a one, it decrements the value in the RP register. The decreased value is valid for the next instruction. The output of the data ROM that corresponds to the decreased value can be read on the next instruction. This bit is ignored if the RP register is specified in the DST field. If this

bit is set to one and the RP register is specified in the SRC field, the value before the decrement is output to the internal data bus.

No change is made to the RP register if this bit is set to zero.

TABLE 4.6 ROM POINTER DECREMENT FIELD

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

4.1.7 SRC Field (Source)

This field specifies the register from which the data that is placed on the IDB comes.

Mnemonic	SRC Field D_7	D_6	D_5	D_4	Specified Register
NON	0	0	0	0	NO Register
A	0	0	0	1	ACC A (Accumulator A)
B	0	0	1	0	ACC B (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 Data No Flag ①
SR	1	0	1	0	SR Status
SIM	1	0	1	1	SI Serial in MSB ②
SIL	1	1	0	0	SI Serial in LSB ③
K	1	1	0	1	K Register
L	1	1	1	0	L Register
MEM	1	1	1	1	RAM

TABLE 4.7 - SOURCE FIELD SPECIFICATIONS

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 in goes to LSB, last bit to MSB (bit reversed).

The registers you may use for this field are shown in table 4.7. The contents of the specified register is output to the internal data bus. If you specify RAM in this field, you cannot specify it again in the DST field. Data cannot be transferred from RAM to RAM. If you specify an accumulator, the value before an ALU operation is output to the internal data bus.

4.1.8 DST Field (Destination)

The value placed on the IDB (SRC) is latched to the register specified in this field.

TABLE 4.8 DESTINATION FIELD SPECIFICATIONS

Mnemonic	DST Field				Specified Register
	D_3	D_2	D_1	D_0	
@NON	0	0	0	0	NO Register
@A	0	0	0	1	ACC A (Accumulator A)
@B	0	0	1	0	ACC B (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
@SOL	1	0	0	0	SO Serial Out LSB ①
@SOM	1	0	0	1	SO Serial Out MSB ②
@K	1	0	1	0	K (Mult)
@KLR	1	0	1	1	IDB → K ROM → L ③
@KLM	1	1	0	0	Hi RAM → K IDB → L ④
@L	1	1	0	1	L (Mult)
@NON	1	1	1	0	NO Register
@MEM	1	1	1	1	RAM

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 $DP_6 = 1$ (i.e., 1, DP_5, DP_4,-DP_0) is placed in K register. IDB is placed in L.

If you specify the same accumulator in both the DST field and ASL bit, the ASL bit is ignored and the ALU performs a NOP. If you specify the DP or RP register in this field, any modification specified by the DPH, DPL, or RPDCR fields is ignored.

Please note that @KLM and @KLR destinations latch data from a special bus to the K or L registers (respectively), as well as the value on the IDB into the L or K multiplier input register. This enables both multiplier input registers to be loaded as part of the same instruction.

The LD instruction uses this field to specify where to load its immediate data.

4.2 JP Instruction

The JP instruction contains three fields (other than the OP code), and it may take one of three forms: unconditional jump, conditional jump, or subroutine call.

FIGURE 4.3 JP INSTRUCTION FORMAT

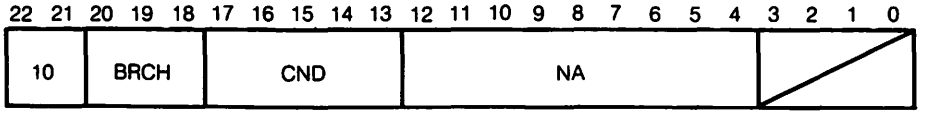


FIGURE 4.4

JP ASSEMBLY LANGUAGE INSTRUCTION FORMAT

```
EXAMPLE:  JSAØ          OVFLOW          ; /*    JUMP TO OVFLOW IF */
                                        /*    THE SIGN BIT OF ACCUMULATOR */
                                        /*    A IS "Ø" */
```

4.2.1 BRCH Field (BRCH)

This field specifies which of the three forms is to be executed. The instruction specifications are shown in Table 4.9.

BRCH FIELD				
MNEMONIC	D_{20}	D_{19}	D_{18}	FUNCTION
JMP	1	0	0	Unconditional Jump
CALL	1	0	1	Subroutine Call
*	0	1	0	Conditional Jump

*Mnemonic of the CND field is used.

TABLE 4.9 BRANCH FIELD

1) Unconditional Jump

The address in the NA field is transferred to PC when this instruction is executed. The CND field is ignored.

2) Conditional Jump

If the condition specified in the CND field is true, the address in the NA field is transferred to PC when this instruction is executed. If the condition is false, PC equals PC + 1.

3) Subroutine Call

The address in the NA field is transferred to PC, and the address (PC + 1) is pushed to the stack. The CND field is ignored.

4.2.2 CND Field (Condition)

This field specifies the condition that must be true for a conditional jump to be executed. All possible conditional jump instructions and their mnemonics are shown in Table 4.10.

TABLE 4.10 CONDITION FIELD

	CND FIELD					
MNEMONIC	D_{17}	D_{16}	D_{15}	D_{14}	D_{13}	CONDITION
JNCA	0	0	0	0	0	CA = 0
JCA	0	0	0	0	1	CA = 1
JNCB	0	0	0	1	0	CB = 0
JCB	0	0	0	1	1	CB = 1
JNZA	0	0	1	0	0	ZA = 0
JZA	0	0	1	0	1	ZA = 1
JNZB	0	0	1	1	0	ZB = 0
JZB	0	0	1	1	1	ZB = 1
JNOVA0	0	1	0	0	0	OVA0 = 0
JOVA0	0	1	0	0	1	OVA0 = 1
JNOVB0	0	1	0	1	0	OVB0 = 0
JOVB0	0	1	0	1	1	OVB0 = 1
JNOVA1	0	1	1	0	0	OVA1 = 0
JOVA1	0	1	1	0	1	OVA1 = 1
JNOVB1	0	1	1	1	0	OVB1 = 0
JOVB1	0	1	1	1	1	OVB1 = 1
JNSA0	1	0	0	0	0	SA0 = 0
JSA0	1	0	0	0	1	SA0 = 1
JNSB0	1	0	0	1	0	SB0 = 0
JSB0	1	0	0	1	1	SB0 = 1
JNSA1	1	0	1	0	0	SA1 = 0
JSA1	1	0	1	0	1	SA1 = 1
JNSB1	1	0	1	1	0	SB1 = 0
JSB1	1	0	1	1	1	SB1 = 1
JDPL0	1	1	0	0	0	DP_L = 0
JDPLF	1	1	0	0	1	DP_L = F (HEX)
JNSIAK	1	1	0	1	0	SI ACK = 0
JSIAK	1	1	0	1	1	SI ACK = 1
JNSOAK	1	1	1	0	0	SO ACK = 0
JSOAK	1	1	1	0	1	SO ACK = 1
JNRQM	1	1	1	1	0	RQM = 0
JRQM	1	1	1	1	1	RQM = 1

4.3.2 DST Field (Destination)

This field specifies the register to which the data in the ID field is loaded. The registers that can be used are the same as those for the DST field of the OP/RT instruction (see Table 4.8). Note that @KLM and @KLR can be used with this instruction allowing a load of both multiplier input registers.

5. Timing

The uPD7720 SPI operates by a single-phase clock applied to the CLK pin. The maximum frequency is 8 MHz which gives a 250 ns instruction cycle.

5.1 Serial Data Timing

The serial data I/O shift clock (SCK) can be input asynchronously with the system clock (CLK). Refer to sections 3.23 (SI), 3.24 (SO), and 7 for detailed information on serial I/O timing (MAX frequency is 2 MHz).

5.2 Reset (RST) Timing

Each RST input must be continued over 3 clock cycles to initialize the system. The RST input initializes the following to zero:

1) PC
2) Flags
3) SR Register
4) DRQ
5) SORQ
6) Serial bit count

5.3 Interrupt

The interrupt input is rising edge sensed. For a rising edge to be accepted as a valid interrupt, the internal interrupt facilities must be enabled prior to that rising edge (EI status bit = 1). Interrupt must be held high for at least 8 clock cycles after the rising edge.

EXAMPLE 1

ASSEMBLY LANGUAGE

INSTRUCTION EXAMPLES

```
1)  MYSELF:   JNSIAK    MYSELF                  ; /* WAIT FOR SERIAL */
                                                  /* INPUT WORD */

2)  MYSELF:   JSOAK     MYSELF                  ; /* WAIT TO OUTPUT THE */
                                                  /* NEXT SERIAL WORD */

3)            OP        MOV       @KLR, SIM       /* PLACE ROM VALUE POINTED TO */
                                                    /* BY RP IN MULTIPLIER INPUT */
                                                    /* REGISTER L */
                                                    /* PLACE SERIAL INPUT */
                                                    /* REGISTER VALUE IN MULT. */
                                                    /* INPUT REGISTER K */
                        ADD       ACCA, RAM       /* ADD RAM VALUE POINTED TO */
                                                    /* BY THE DP TO ACCA */
                        DPINC                     /* INCR DP LOW 4 BITS */
                        RPDEC                   ; /* DECR RP (ROM POINTER) */

4)            OP        MOV       @MEM, A         /* MOV ACCA TO RAM */
                                                    /* POINTED TO BY DP */
                        XOR       ACCA, IDB       /* XOR VALUE ON IDB */
                                                    /* (ACCA IS ON IDB) WITH */
                        DPINC                   ; /* ACCA (CLEAR ACCA & FLAG A) */
                                                    /* INCR DPL */
```

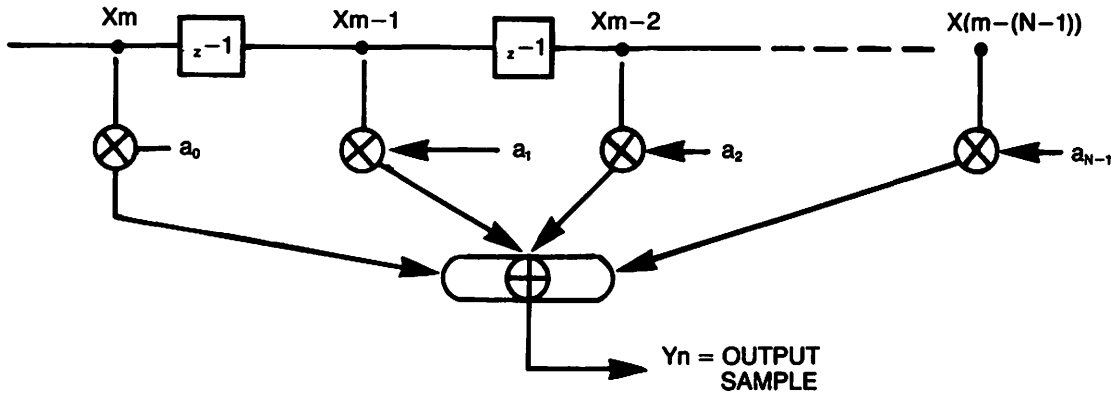
NOTE: ASSEMBLER PLACES NO OPERATION VALUES IN FIELDS NOT SPECIFIED.

EXAMPLE #2

FIR FILTER

This is an example of an FIR-type digital filter. This example should familiarize you with the features of the SPI and its assembly language. This routine is not useful for all applications, but it demonstrates some of the SPI's unique and powerful features.

TRANSVERSAL (FIR) FILTER



EQUATION — $Y_m = \sum_{n=0}^{63} a_n X_{m-n}$

- 64 taps
- All coefficients stored in ram (programable).
- All 31 bits of multiplier output used.

MEMORY MAP

DP_H \ DP_L	ØØØØ	ØØØ1		111Ø	1111	
000	Xm			Xm-14	Xm-15	DATA
001	Xm-16			Xm-	Xm-31	
010	Xm-32			Xm-46	Xm-47	
011	Xm-48			Xm-67	Xm-63	
100	a_0			a_{14}	a_{15}	COEFF.
101	a_{16}			a_{30}	a_{31}	
110	a_{32}			a_{46}	a_{47}	
111	a_{48}			a_{62}	a_{63}	

$DP_6 = Ø$ (DATA rows)

$DP_6 = 1$ (COEFF. rows)

PROGRAM EXAMPLE #3

FIR MAIN PROGRAM

```
ORG             1ØØH                    ; /* PROGRAM AT INTERRUPT LOCATION*/

                                        /* INITIALIZE */

LDI             @DP, ØØØØH              ; /* SET DP = 000/0000 */
OP      MOV     @NON, A                   /* CLEAR ACCA */
        XOR     ACCA, IDB               ; /* XOR ACCA WITH ACCA */
OP      MOV     @NON, B                   /* CLEAR ACCB */
        XOR     ACCB, IDB               ;
JSIAK   $                               ; /* WAIT FOR INPUT DATA */
OP      MOV     @MEM, SIM               ; /* GET INPUT VALUE FROM THE */
                                          /* SERIAL INPUT REGISTER*/

                                        /* FIRST TAP */

OP      MOV     @KLM, MEM               ; /* PLACE a  IN K (MULT. INPUT)*/
                                          /* PLACE X  IN L (MULT. INPUT)*/
                                                  o       o

OP      MOV     @TR, L                    /* SAVE X  IN TR TO MOVE */
        ADD     ACCB, N                 ; /* IT DOWN ONE DELAY (MEM LOC) - */
                                          /* ADD LOW WORD MULT OUTPUT */

OP      ADC     ACCA, M                   /* ADD HIGH WORD TO ACCA */
        DPINC                           ; /* WITH CARRY FROM LOW */
                                          /* INCR LOW 4 BITS OF DP */
                                          /* DP = 000/0001 NXT INSTRUCTION */


                                        /* OTHER TAPS */

CAL     TRFIL                           ; /* DO ALL THE TAPS LEFT */
                                          /* TO DO IN THIS ROW*/
                                          /* DP = 000/0000*/
CAL     TRFIL                           ; /* DO ALL TAPS IN THIS ROW*/
                                          /* DP = 001/0000*/

OP      M2                              ; /* XOR DPH WITH Ø1Ø*/
                                          /* DP MXT INSTR = 010/000 */

CAL     TRFIL                           ; /* DO ROW DP = 010 0000*/

CAL     TRFIL                           ; /* DO ROW DP = 011 0000*/
JNSOAK  $                               ; /* WAIT FOR OUTPUT REGISTER CLEAR*/

                        /* - 16 BIT ACCURACY SOLUTION IN ACCA -*/
MOV             @SOM, A                 ; /* MOV RESULT TO SO REGISTER */
LDI             @SR, Ø8H                ; /* ENABLE INT*/
OP              RET                     ; /* RETURN TO MAIN PROGRAM */
```

FIR SUBROUTINE

```
TRFIL:
        OP      MOV   @KLM, MEM        ; /* PLACE a_m & Xm IN MULTIPLIER*/
                                          /* INPUT REGISTERS K & L*/

        OP      MOV   @MEM, TR            /* PLACE DATA IN MEMORY LOCATION*/
                ADD    ACCB, N          ; /* ONE DELAY FROM PREVIOUS LOCATION*/
                                          /* ADD LOW WORD MULT.*/

        OP      MOV   @TR, L              /* SAVE DATA TO PLACE IN DELAYED*/
                ADC    ACCA, M            /* LOCATION*/
                DPINC                   ; /* ADD WITH CARRY (B)*/
                                          /* HIGH MULT WORD*/
                                          /* INCR DP_L*/

        JDPLO   TRDIL                   ; /* TEST FOR DP_L = 0000 (B)*/
                                          /* IF SO HAVE FINISHED*/
                                          /* A ROW GO TO RET*/
        JMP     TRFIL                   ; /* NOT FINISHED NXT*/
                                          /* TAP*/

TRDIL:
        OP      M1                        /* XOR DP_HL WITH ØØ1B (CHANGE ROW)*/
                RET                     ; /* DP = AAA | ØØØØ */
```

6. TYPICAL SYSTEM CONFIGURATIONS

The uPD7720 SPI is a single chip microcomputer. However, it is also designed to operate as a complex peripheral to a microcomputer.

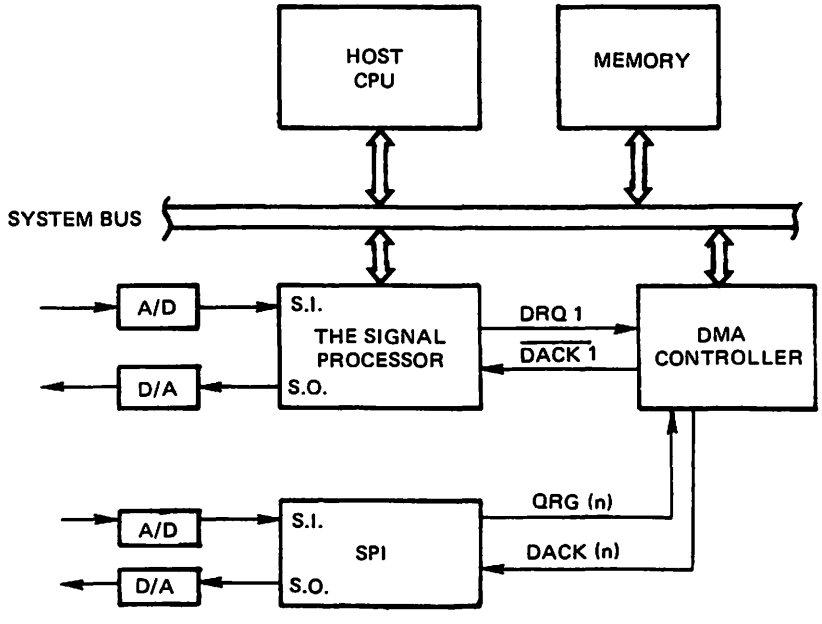
Three configurations are shown in Figures 6.1-2-3. The first is with the SPI operating as a complex peripheral, inputting and outputting data serially as well as communicating on a microcomputer system bus.

The second example shows two uPD7720s in a cascaded configuration. The SPIs could also be attached to a microcomputer bus if desired, but it is not shown in this example. This type of configuration is good for systems in which data rates exceed the capability of one SPI to process the allocated function completely within the available time.

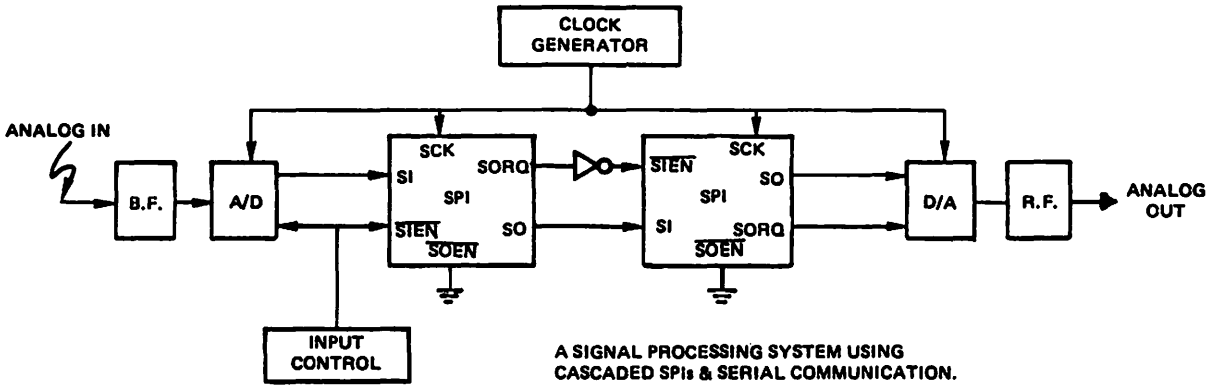
The last configuration is an application example.

EXAMPLE #A

A SIGNAL PROCESSING SYSTEM USING SPI(s) AS A COMPLEX COMPUTER PERIPHERAL

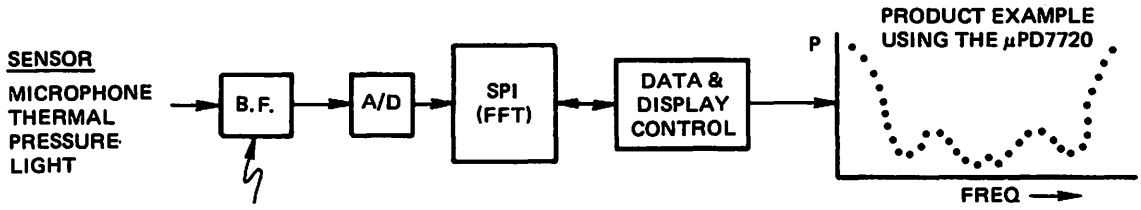


EXAMPLE #B



A SIGNAL PROCESSING SYSTEM USING CASCADED SPIs & SERIAL COMMUNICATION.

EXAMPLE #C



SPECTRUM ANALYSIS SYSTEM

7. ELECTRICAL AND TIMING SPECIFICATIONS

ABSOLUTE MAXIMUM RATINGS*

Voltage (V_{CC} Pin) . . . . . -0.5 to +7.0 Volts ①
Voltage, Any Input (V_I) . . . . . -0.5 to +7.0 Volts ①
Voltage, Any Output (V_o) . . . . . -0.5 to +7.0 Volts ①
Operating Temperature (T_{OPT}) . . . . . -10°C to +70°C
Storage Temperature (T_{STG}) . . . . . -65°C to +150°C

Note: ① With respect to GND.

*COMMENT: 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.

$T_a = 25°C$

DC CHARACTERISTICS

$T_a = -10 \sim +70°C$, $V_{CC} = +5V \pm 5\%$

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT	CONDITION
Input Low Voltage	V_{IL}	-0.5		0.8	V	
Input High Voltage	V_{IH}	2.0		V_{CC} +0.5	V	
CLK Low Voltage	$V_{\phi L}$	-0.5		0.45	V	
CLK High Voltage	$V_{\phi H}$	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} = 0V
Input Load Current	I_{LIH}			10	μA	$V_{IN} = V_{CC}$
Output Float Leakage	I_{LOL}			-10	μA	V_{OUT} = 0.47V
Output Float Leakage	I_{LOH}			10	μA	$V_{OUT} = V_{CC}$
Power Supply Current	I_{CC}		180	280	mA	

CAPACITANCE

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT	CONDITION
CLK, SCK Input Capacitance	C_ϕ			20	pF	f_c = 1 MHz
Input Pin Capacitance	C_{IN}			10	pF	
Output Pin Capacitance	C_{OUT}			20	pF	

AC CHARACTERISTICS

$T_a = -10 \sim +70°C$, $V_{CC} = +5V \pm 5\%$

PARAMETER	SYMBOL	MIN	TYP	MAX	UNIT	CONDITION
CLK Cycle Time	ϕCY	122		2000	ns	①
CLK Pulse Width	ϕD	60			ns	
CLK Rise Time	ϕR			10	ns	①
CLK Fall Time	ϕF			10	ns	①
Address Setup Time for $\overline{RD}$	t_{AR}	0			ns	
Address Hold Time for $\overline{RD}$	t_{RA}	0			ns	
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	②
DRQ Delay	t_{AM}			150	ns	
$\overline{DACK}$ Delay Time	t_{DACK}	1			ϕD	②
SCK Cycle Time	t_{SCY}	480		DC	ns	
SCK Pulse Width	t_{SCK}	230			ns	
SCK Rise/Fall Time	t_{RSC}			20	ns	①
SORQ Delay	t_{DRQ}	30		150	ns	$C_L = 100$ pF
$\overline{SOEN}$ Setup Time	t_{SOC}	50			ns	
SOEN Hold Time	t_{CSO}	30			ns	
SO Delay from SCK = LOW	t_{DCK}			150	ns	
SO Delay from SCK with SORQ ↑	t_{DZRQ}	20		300	ns	②
SO Delay from SCK	t_{DZSC}	20		300	ns	②
SO Delay from SOEN	t_{DZE}	20		180	ns	②
$\overline{SOEN}$ to SO Floating	t_{HZE}	20		200	ns	②
SCK to SO Floating	t_{HZSC}	20		300	ns	②
SO Delay from SCK with SORQ ↓	t_{HZRQ}	70		300	ns	②
$\overline{SIEN}$, SI Setup Time	t_{DC}	55			ns	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	

① Voltage at measuring point of timing 1.0V and 3.0V

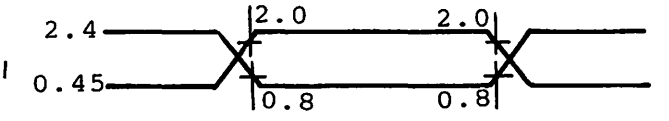
② Voltage at measuring point of AC Timing

$V_{IL} = V_{OL} = 0.8V$

$V_{IH} = V_{OH} = 2.0V$

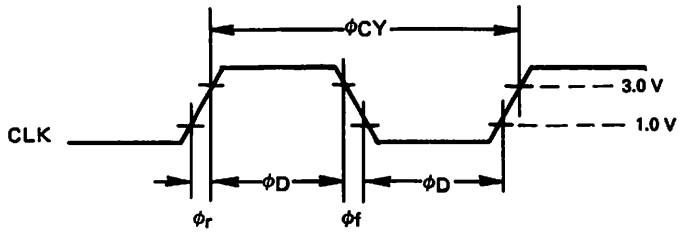
Input Waveform of AC Test (except CLK, SCK)

2.4	2.0	2.0
0.45	0.8	0.8

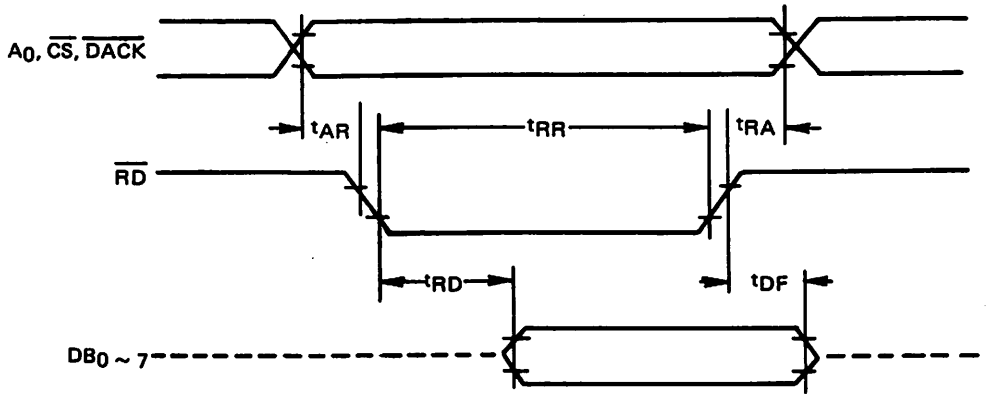


TIMING WAVEFORMS

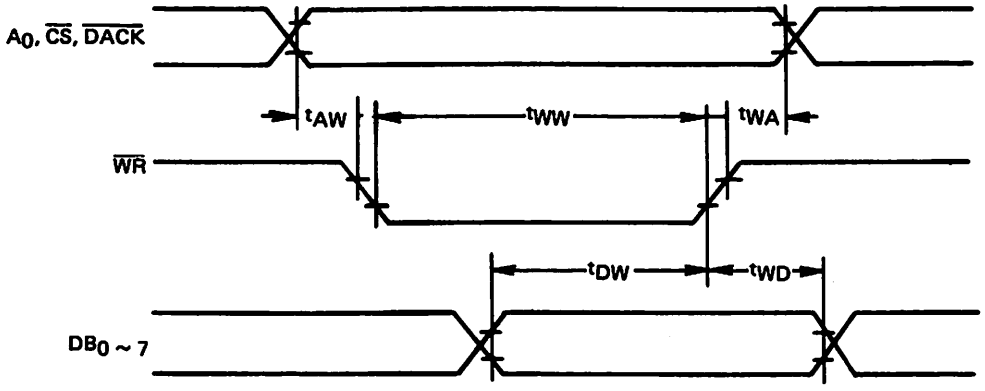
CLOCK



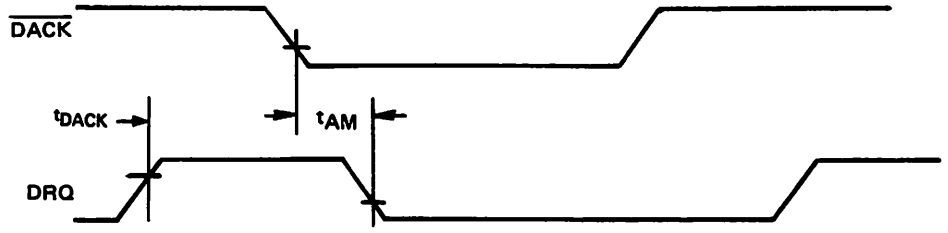
READ OPERATION

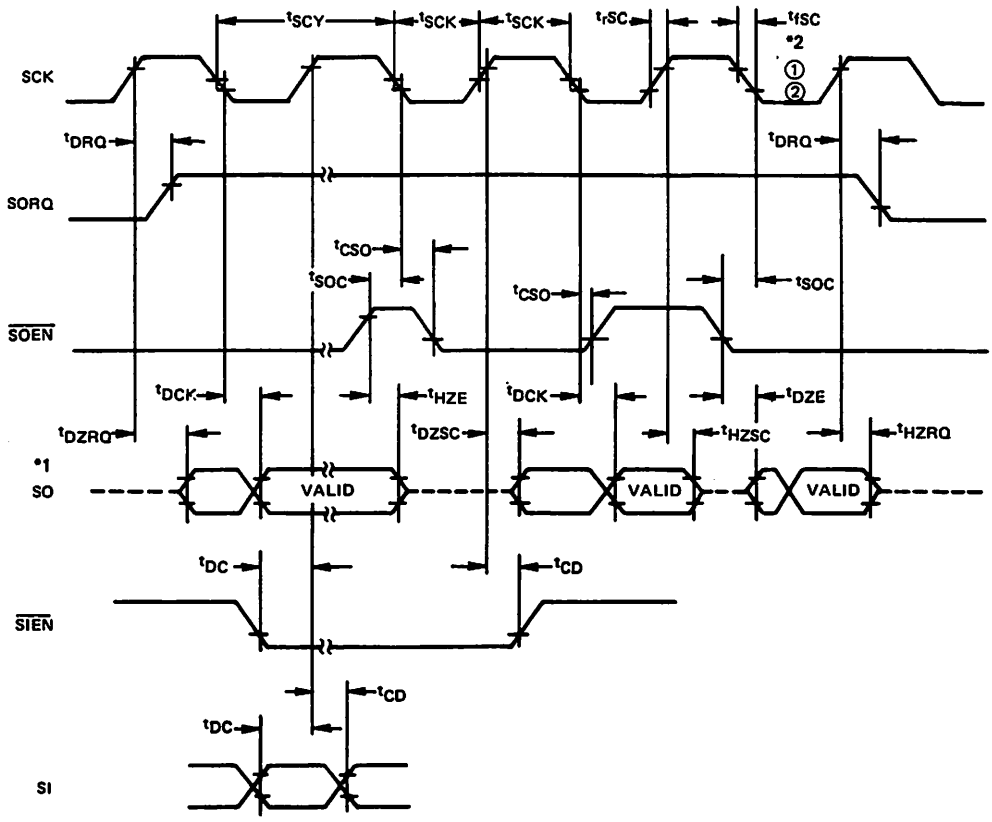


WRITE OPERATION



DMA OPERATION





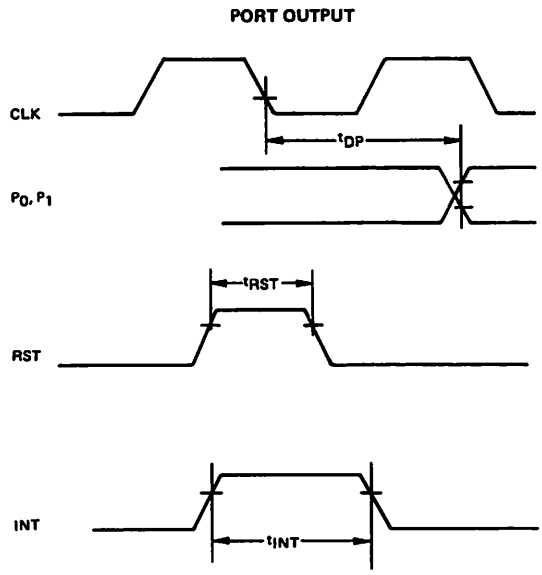
* 1: For SO timing, the data at rising edge of SCK is valid and the other data is invalid. In set-up hold time of data for SCK, the most strict specifications are the following.

$$\text{set-up} = t_{SCK} - t_{DCK}$$

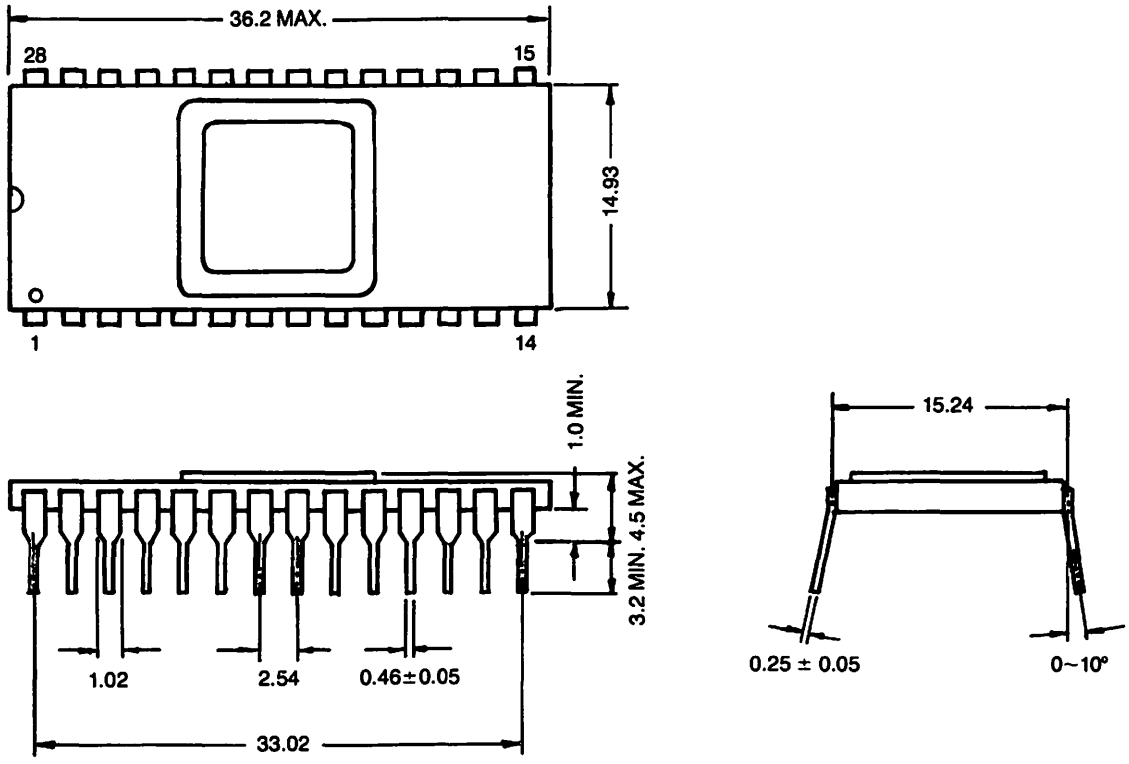
$$\text{hold} = t_{HZRQ}$$

* 2: Voltage at measuring point of t_{rsc} and t_{fsc} for SCK timing

① 3.0V ② 1.0V



Package Dimensions



4.2.3 NA Field (Next Address)

This field specifies the address to which the program vectors.

4.3 LD Instruction

FIGURE 4.5 LD INSTRUCTION FORMAT

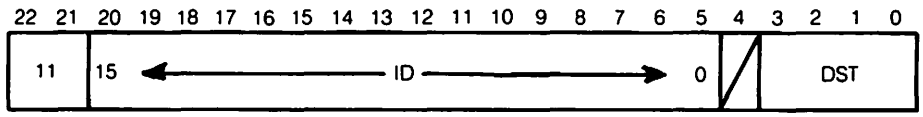


FIGURE 4.6 LD ASSEMBLY LANGUAGE INSTRUCTION FORMAT

LOAD IMMEDIATE DATA VALUE ON TO INTERNAL DATA BUS AND STORE INTO REGISTER OR MEMORY ON INTERNAL DATA BUS

```
EXAMPLE:   LDI      @A,ØØØØH          ;    ØØØØH TO ACCUMULATOR A
```

The LD instruction is composed of two fields other than the OP code, and is executed in the following manner:

1) Data is output from the ID field, over the internal data bus, to the register specified in the DST field.

2) The 0 bit of the ID field is output to the 0 bit of the internal data bus and bit 15 to bit 15 of the bus.

4.3.1 ID Field (Immediate Data)

This field contains the data to be transferred to the register specified in the DST field.

NEC
NEC Electronics U.S.A. Inc.
Microcomputer Division

One Natick Executive Park
Natick, Massachusetts 01760
TEL 617-655-8833
TWX 710-386-2110

REGIONAL SALES OFFICES

EASTERN REGION
275 Broadhollow Road, Route 110
Melville, NY 11747
TEL: 516-293-5660
TWX: 510-224-6090

MIDATLANTIC REGION
2000 Grosvenor Century Plaza, Suite 333
10632 Little Patuxent Parkway
Columbia, MD 21044
TEL: 301-730-8600
TWX: 710-862-2868

MIDWESTERN REGION
5105 Tollview Drive, Suite 190
Rolling Meadows, IL 60008
TEL: 312-577-9090
TWX: 910-233-4332

NORTHEASTERN REGION
21-G Olympia Avenue
Woburn, MA 01801
TEL: 617-935-6339
TWX: 710-348-6515

NORTHWESTERN REGION
20480 Pacifica Drive, Suite E
Cupertino, CA 95014
TEL: 408-446-0650
TWX: 910-338-2085

OHIO VALLEY REGION
19675 West Ten Mile Road
Southfield, MI 48075
TEL: 313-352-3770
TWX: 810-224-4625

SOUTHWESTERN REGION
1940 West Orangewood Avenue, Suite 205
Orange, CA 92668
TEL: 714-937-5244
TWX: 910-593-1629

SOUTH CENTRAL REGION
16475 Dallas Parkway, Suite 290
Dallas, TX 75248
TEL: 214-931-0641
TWX: 910-860-5284

SOUTHEASTERN REGION
Vantage Point Office Center, Suite 209
4699 North Federal Highway
Pompano Beach, FL 33064
TEL: 305-785-8250
TWX: 510-956-9722

Fabio Montoro

7720TM-1-82-1K

$15.00