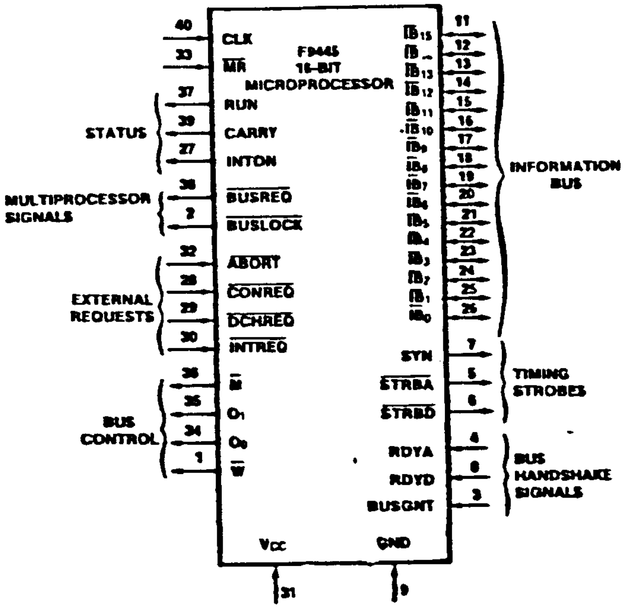
DESCRIPTION

The ICD9445 is a 16-bit microprocessor implemented in CMOS. Sophisticated pipeline architecture gives the ICD9445 very fast execution times. The processor has eight program accessible registers and the capability of directly addressing 128k bytes of memory. It can address 62 I/O devices, handle 16 levels of priority interrupt, and perform fast directmemory access. It has control lines to provide operator-console functions and has an on-chip self-test program. It is supported with extensive software including editors, debuggers, macroassembler, relocating loader, real-time executives, interactive multi-user operating systems and many other utilities. It executes many languages including Fortran, Pascal, C, BASIC, COBOL, etc.

- Parallel architecture for high throughput
- Directly addresses up to 128K bytes of memory with 11 addressing modes
- Eight program-accessible registers (AC0, AC1, AC2, AC3, SP, FP, PC, PSW)
- Versatile instruction set including memory reference, ALU, I/O, Stack, Multiply/Divide, and Floating Point Assist (Scale/Normalize) instructions with 8-bit byte, 16-bit word or 32-bit double word data
- Multi-processing
- Flexible operator control and self-test
- Operation with single clock up to 24 MHz
- TTL Input/Output structure with CMOS internal circuits and bi-directional signals
- 40-pin DIP needing a single +5 V power supply

Pin Functions

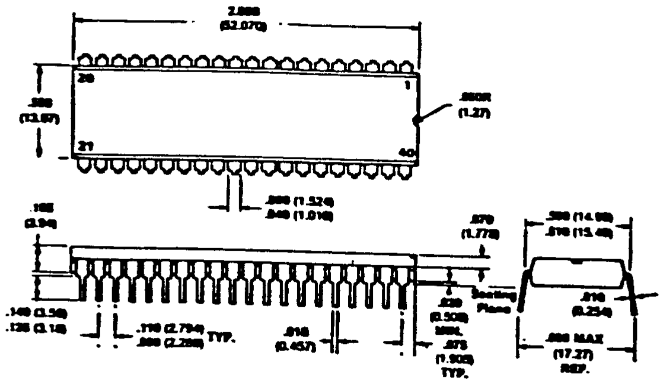


Absolute Maximum Ratings
Beyond these ratings useful life of the device may be impaired.

Storage Temperature	−65 to +150°C
Ambient Temperature Under Bias	−55 to +125°C
V_{CC} Pin Potential to Ground Pin	−0.5 to +6.0 V
Input Voltage (dc)	−0.5 to +5.5 V
Input Current (dc)	−20 to +5 mA
Output Voltage (Output HIGH)	−0.5 to +5.5 V
Outout Current (dc) (Output LOW)	+20 mA

Package Information

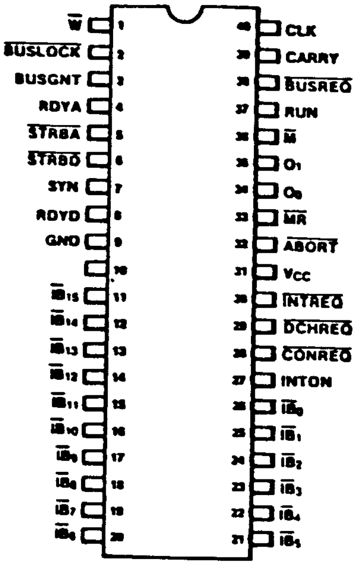
40-Pin Plastic Dual-In-Line



Notes

All dimensions are in inches (bold) and millimeters (parentheses).
Pins are tin-plated kovar.
Package material is plastic.

Connection Diagram
40-Pin DIP (Top View)



$\overline{\text{DCHREQ}}$, Pin 29 — Data Channel Request — Active LOW input; initiates data-channel cycles while LOW after current instruction.

$\overline{\text{CONREQ}}$, Pin 28 — Console Request — Active LOW input; initiates a console operation after current instruction.

$\overline{\text{INTREQ}}$, Pin 30 — Interrupt Request — Active LOW input; initiates entry to interrupt procedure, if interrupts are enabled.

$\overline{\text{ABORT}}$, Pin 32 — Abort — Active LOW input; initiates abort sequence in the current microcycle.

Power

V_{CC}, Pin 31 — Power Supply —'Requires +5 V.

GND, Pin 9 — Ground.

Register Set

The 9445 has eight user-accessible registers including seven 16-bit registers and a program status word (PSW). Then PSW (see *Figure 3)* contains the following four flags: carry (bit 0), 32Kw (bit 1), trap enable (bit 2), and overflow (bit 15). The carry flag (C) indicates the state of the carry bit during arithmetic and logic operations. The 32Kw flag indicates whether the processor is operating in the 32K word ("1") or 64K word ("0") mode. The trap enable/disable flag (ETRP) indicates whether the trap instruction is enabled ("1") or disabled ("0"). The overflow flag (V) indicates twos-complement overflow in arithmetic operations.

In addition, there is an interrupt-on (INTON) flag. The CPU responds to interrupt requests from external I/O devices when the flag is set ("1"). When it is clear ("0"), all interrupt requests are ignored by the CPU. The state of the flag can be altered by the Interrupt-Enable or Interrupt-Disable instruction.

The seven 16-bit registers are: the program counter (PC) that sequences the execution of instructions, four general-purpose accumulators (AC0 through AC3), the stack pointer (SP) and the frame pointer (FP). The program counter sequences the execution of instructions. It holds the address of the next instruction to be executed and is automatically incremented to fetch instructions from consecutive memory locations. A Skip, Jump, Jump-to-Subroutine, Trap instruction, an interrupt generated by an I/O device or an Abort can alter the sequential execution of instructions.

The four accumulators serve as source and destination registers for 16-bit arguments in arithmetic-and-logic instructions. These instructions process the contents of the source accumulators and a base value for the carry flag and store the 16-bit result in the destination accumulator. The associated carry and overflow flags are set or cleared depending on the result of the ALU operation and the vase value of carry. Accumulators AC2 and AC3 also serve as index registers during memory reference instructions. In addition, AC3 functions as a subroutine linkage register, and the pair AC0 and AC1 are used as a 32-bit register in the multiply/divide, the normalize and parametric double-shift instructions.

The other two 16-bit registers serve as temporary storage and as the stack pointer (SP) and frame pointer (FP) in the stack manipulation instructions. The stack pointer contains the address of the top of the stack, i.e. the last word "pushed" onto the stack which is also the first word that may be "popped." The frame pointer contains the address of the highest location in a block of five words on the stack, a "frame," containing program status information used to return from a subroutine

The frame pointer is updated by the Save and Return instructions which are intended to be the first and last instructions, respectively, executed by a subroutine. When a Jump-to-Subroutine instruction is executed, the value CA + 1 is stored in AC3 (CA is the address of the instruction currently being executed). The Save instruction then pushes five key words onto the stack in the following order: first, the contents of AC0; second, the contents of AC1; third, the contents of AC2; fourth, the value of FP before the Save; and fifth, the contents of AC3 and the value of the carry bit in 32K word mode only. At this point, SP points to the top of the frame (which is the current top of the stack), and that address becomes the new value of FP. This new value of FP is also placed in AC3. When a Return instruction is executed, the five words stored in the frame referenced by FP are used to restore accumulators AC0 through AC2 to their values at the time preceding the Save. FP is restored to its previous value (pointing to the last previously saved five-word frame) and PC is loaded with the return address which had been placed in AC3 by the previous Jump-to-Subroutine and pushed onto the stack by the previous Save. The restored value of FP is also placed in AC3 by the Return instruction.

Information may also be moved between SP or FP and any of the four accumulators by the instructions MTFP, MFFP, MTSP, and MFSP without affecting the source register of the move or any of the registers not specified with the instruction. This allows setting up multiple stacks whose pointers are saved in main memory when not in use.

All inputs and outputs, except for bi-directional signals (data bus), are TTL.

Information Bus

$\overline{\text{IB}}_0$ through $\overline{\text{IB}}_{15}$, Pins 11 through 26 — 16-bit Bus — Active LOW bidirectional; $\overline{\text{IB}}_0$ is most significant bit; address valid with $\overline{\text{STRBA}}$ strobe; data valid with $\overline{\text{STRBD}}$ strobe; 3-state during data-channel and non-bus cycles.

Timing and Status

SYN, Pin 7 — Synchronize Output — Clock that is active every cycle; may be used for external synchronization of memory and I/O control.

$\overline{\text{STRBD}}$, Pin 6 — Data Strobe — Active LOW output; active only during memory, I/O, console, or data-channel cycles; used as strobe for data.

RDYD, Pin 8 — Data Ready — Active HIGH input; used to synchronize external devices with the F9445 during data transfer; a LOW level halts the processor.

$\overline{\text{STRBA}}$, Pin 5 — Strobe Memory Address Register — Active LOW output; active only during normal memory cycles; not active during write portion of read-modify-write cycles (DSZ, ISZ, STB instructions and auto-increment/decrement addressing modes); used as strobe for external address register; active on I/O cycles when I/O instruction is output onto bus.

RDYA, Pin 4 — Address Ready — Active HIGH input; maintains address on bus when LOW.

$\overline{\text{M}}$, Pin 36 — Memory or I/O Function — Active LOW output.
O_1, Pin 35 — Memory or I/O Function — Active HIGH output.
O_0, Pin 34 — Memory or I/O Function — Active HIGH output; these pins indicate the type of bus transfer as shown in the following table.

	$\overline{\text{M}}$	O_1	O_0	Function
Memory	0	0	0	Instruction Fetch
	0	0	1	Operand
	0	1	0	Indirect Address
	0	1	1	Address Save on interrupt, abort, and trap
I/O	1	0	0	Input or Output
	1	0	1	Data Channel Acknowledge
	1	1	0	Read Console Code
	1	1	1	Console Data

During machine cycles that do not use the bus, the $\overline{\text{M}}$ and O lines will be "111." $\overline{\text{BUSREQ}}$ and the bus strobes are inactive in these cycles.

$\overline{\text{W}}$, Pin 1 — Write Output — Indicates direction of data flow; HIGH indicates a read or input operation; LOW indicates a write or output operation; 3-state during data-channel cycles and short cycles.

RUN, Pin 37 — Run Status — Active HIGH output; LOW when in halt state.

CARRY, Pin 39 — Carry Status — Active HIGH output; copy of carry bit.

INTON, Pin 27 — Interrupt-On Status — Active HIGH output; copy of Interrupt-On flag; HIGH when interrupts enabled.

CLK, Pin 40 — Clock Input — Single-phase clock; positive-edge triggered.

Arbitration

$\overline{\text{BUSREQ}}$, Pin 38 — Bus Request — Active LOW output; indicates that a bus cycle is required; useful in multi-microprocessor system.

BUSGNT, Pin 3 — Bus Grant — Active HIGH input; used for multi-microprocessor operation; a LOW level inhibits address output

$\overline{\text{BUSLOCK}}$, Pin 2 — Bus Lock — Active LOW open collector output; set during read portion of read-modify-write cycles (on DSZ, ISZ, STB, and auto-increment/decrement), reset during write portion of those cycles; used in multi-microprocessor system.

Service Request

The order of priority of requests and interrupts, from highest to lowest, is as follows: $\overline{\text{MR}}$, $\overline{\text{ABORT}}$, $\overline{\text{DCHREQ}}$, Stack Overflow Interrupt, $\overline{\text{INTREQ}}$, and $\overline{\text{CONREQ}}$.

$\overline{\text{MR}}$, Pin 33 — Master Reset — Active LOW input; a LOW level causes the processor to enter a wait state after completing the next full cycle. On the completion of the RESET sequence the state of the 9445 is as follows: RUN is cleared, interrupts are disabled, traps are enabled, and the processor is in the 32K mode.

Addressing Ranges and Modes

The 9445 memory reference instructions support two address ranges and a variety of addressing modes. These modes include direct/indirect addressing which may be absolute, PC-relative, or indexed by AC2 or AC3. Additional addressing modes include auto-increment, auto-decrement, and address via stack and frame pointers. The two address ranges in which the 9445 can operate are 128K byte (64K word) or 64K byte (32K word) logical address space. The 9445 master resets to the 64K byte (32K word) address range. The 128K byte (64K word) address range can be enabled or disabled under program control.

64K Byte (32K Word) Addressing Mode

After the master reset is activated or the D64K instruction is executed, the 9445 operates in the 64K byte (32K word) address range. In this mode of operation, it uses 15-bit addresses to fetch up to 32K words from the memory and uses either the least-significant sixteenth bit to select high or low byte of the word in the byte instructions or the most-significant sixteenth bit to specify the remaining 15 bits of the word as an indirect address in multi-level indirect address instructions.

Fig. 2 9445 Register Model

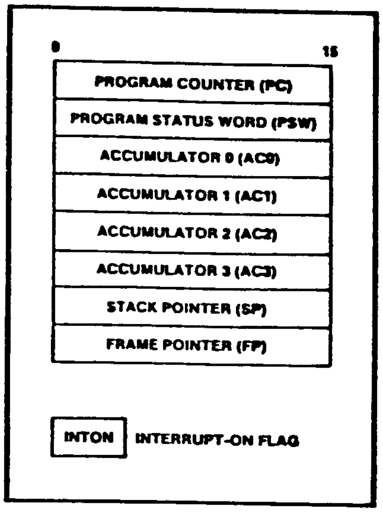
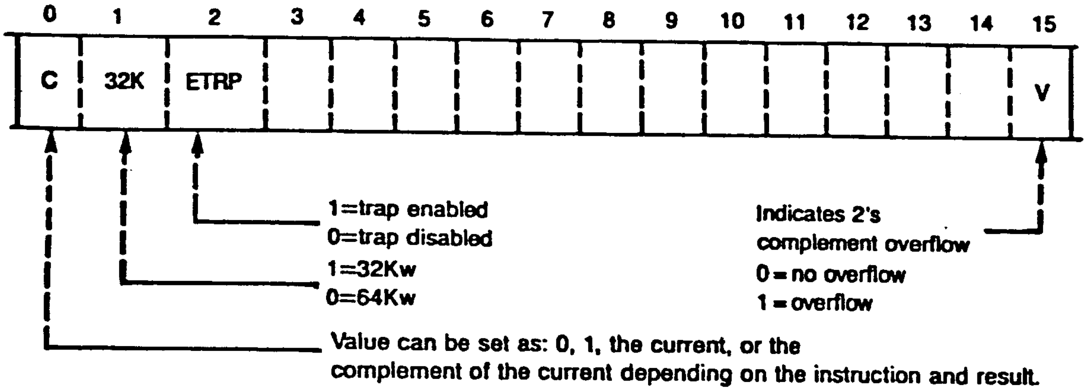


Figure 3 9445 Program Status Word (PSW) (1 of 2)



1=trap enabled
0=trap disabled

1=32Kw
0=64Kw

Value can be set as: 0, 1, the current, or the complement of the current depending on the instruction and result.

Indicates 2's complement overflow
0 = no overflow
1 = overflow

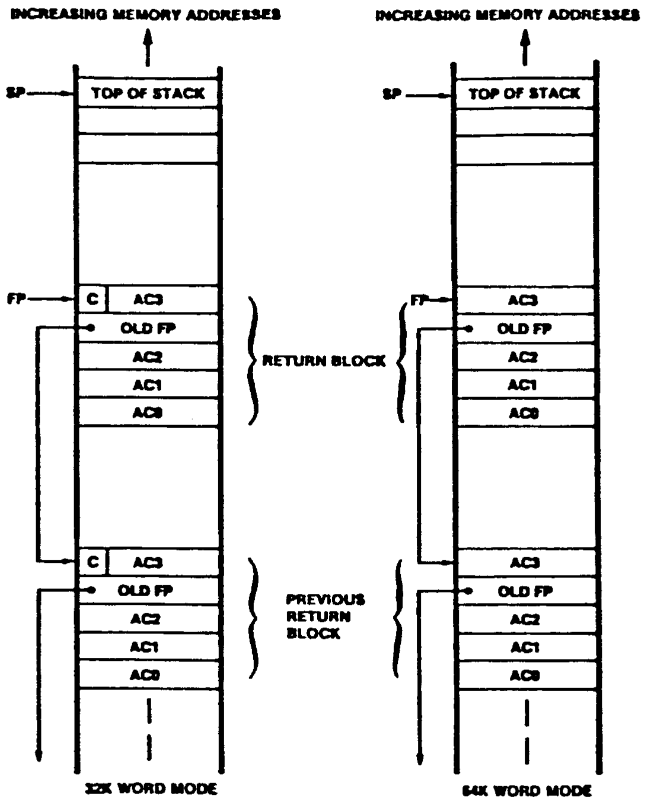
Note: Interrupt-On flag is not a part of the PSW. It can only be tested with SKIP instruction.
1 = responds to interrupt requests, 0 = ignores interrupt requests.

Figure 3 9445 Memory Map

APL start 64K word mode	177777
...	
APL start 32K word mode	077777
...	
Top of page zero memory	000377
...	
ABORT/TRAP vector	000047
ABORT/TRAP address save	000046
Auto-decrement location	000037
↕	000036
	000035
	000034
	000033
	000032
	000031
Auto-decrement location	000030
Auto-increment location	000027
↕	000026
	000025
	000024
	000023
	000022
	000021
Auto-increment location	000020
...	
Stack Fault vector	000003
	000002
Interrupt vector	000001
Interrupt/Stack Fault save address	000000

All addresses are octal numbers.

Fig. 4 Data Organization in a Stack (LIFO) (2 of 2)



In the load-Byte (LDB) and Store-Byte (STB) instructions, a 16-bit accumulator is specified as the byte pointer. The most significant 15 bits of the byte pointer are treated as the logical address of the word containing the byte which the least significant bit specifies, selecting the high (if "0") or low (if "1") byte of the word.

The remaining memory reference instructions specify effective addresses of 16-bit words via various addressing modes described below:

Page Zero — In this mode the instruction provides an 8-bit absolute address to access the first 256 words (page zero) of memory.

PC Relative — In this mode the instruction provides an 8-bit twos-complement signed number that is added to the program counter to access 128 locations below and 127 locations above the address specified in the program counter.

Indexed by AC2 (or AC3) In these two modes the instruction provides an 8-bit twos-complement signed number that is added to AC2 (or AC3) to access 128 locations below and 127 locations above the address specified in the accumulator.

The memory reference instruction may specify any of the above four memory addressing modes to be either direct or indirect. For direct addressing, the effective address computed using the eight address bits of the instruction is the final address of the target word to be stored or retrieved.

For indirect addressing, the effective address computed from the eight address bits of the instruction is used to fetch a 16-bit word that supplies the address of the target word. If the most significant bit of this word is "0," the 15 least significant bits provide the address of the target word. However, if the most significant bit of this word is "1," this specifies a further level of indirect address. In that case, the 15 least significant bits refer to the address of another word which could provide the final address of the target, depending on whether its most significant bit is "0" or "1". Thus, multiple levels of indirect addressing continue until a word is fetched with a most significant bit of "0." These multiple levels of indirect addressing may also occur during the processing of program interrupts if the most significant bit of the vector is "1." Such multiple levels of indirect addressing are only allowed in the 32K word address mode.

The next two types of addressing modes are the auto-increment and auto-decrement modes. When locations 20 through 27 (octal) are indirectly addressed, the auto-increment mode is activated: the contents of the specified location are first incremented and stored back and this new value is treated as the effective address (which can, in turn, be either direct or indirect). Locations 30 through 37 (octal) are used as auto-decrement locations in a similar manner.

The last type of addressing is stack addressing in which the address of the memory reference is derived from the stack pointer.

128K Byte (64K Word) Addressing Range

After the E64K instruction is executed, the 9445 starts operating with the 128K byte (64K word) addressing range. In this range, the 9445 uses 16-bit addresses to fetch up to 64K words from the memory and supports all the eleven addressing modes described previously. However, only one level of indirect addressing is allowed — the one specified in the instruction — since with 16-bit addresses there are no bits available in the words fetched to indicate further indirect addressing.

The byte pointer is also different in the 128K byte (64K word) case compared to the 64K byte (32K word) case. The 64K word range byte pointer is 17 bits wide and is composed of the carry flag and the 16-bit accumulator specified in the LDB or STB instruction. The value of the least-significant bit of the 17-bit word selects the high (if "0") or low (if "1") byte of the word the to be loaded or stored. The carry flag is the most significant bit of the 17-bit word.

Instruction Set

The 9445 has fixed-length instructions, each of which is 16 bits long and divided into several fields. The fields are used to specify the operation code and other related actions, to define conditions and specify the CPU registers containing arguments, to define I/O device codes, and to provide the displacements for the calculation of effective addresses of memory locations.

The whole instruction set can be divided into five broad groups:

Memory Reference Instructions
Arithmetic-and-Logic Instructions
Stack Manipulation Instructions
I/O Instructions
Control Instructions

The Memory Reference instructions modify the contents of memory locations, alter program execution sequence, and move operands between the accumulators and memory locations. The contents of accumulators and the carry and overflow flags are processed by the Arithmetic-and-Logic instructions. The Stack instructions manipulate the registers and the memory in stack-associated operations. The I/O instructions effect data transfers between the accumulators and I/O devices. The Control instructions modify or interrogate the state of the CPU and operator console, performing such actions as controlling the status of the interrupt-on flag and reading the status of the console switch register.

The 9445 instruction set is shown on the following pages. The assembly-language format of each instruction is shown on the left, followed by the name of the instruction and a symbolic description of its action. The corresponding bit pattern for each instruction is shown on the right side of the page.

Following each group of instructions are assembly-language mnemonics and binary representations for optional instruction parts (within square brackets) and accumulator codes, to be inserted at the indicated places in the instructions. Items enclosed in parentheses represent contents of the memory locations addressed by that item.

The required separator, indicated in the assembly-language formats by a square (□), may be entered as any number of combination of space, tab characters or a comma for the macro-assembler; the separator must be a single space for the PEPBUG-45 program.

Input/Output Operations

Input/output devices can transfer data to the 9445-based microcomputer via:

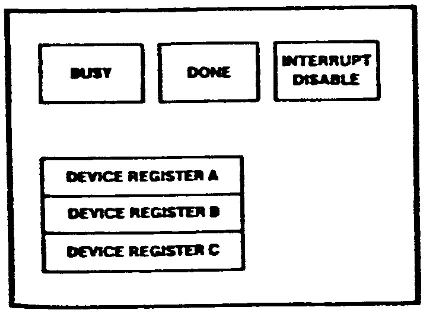
Programmed I/O using the I/O instructions of the 9445,

Memory-mapped I/O using the load/store instructions of the 9445, or

Direct memory access via data-channel transfers.

For programmed I/O, the device consists of up to three (minimum one) bidirectional 16-bit device registers, denoted as A, B, and C, and three 1-bit flags: Busy, Done and Interrupt Disable (see *Figure 5)*. the 2-bit status word composed of Busy and Done represents one of up to four possible states of the device, viz. idle, busy, partially done and completely done (refer to *Device Status Flags* subsection). The 9445 I/O instructions allow data transfers between any of the accumulators (AC0 through AC3) and any of the device registers (A through C), and can test and set the Busy, Done and Interrupt-Disable flags.

Fig. 6 I/O Device Model



The 9445 can transfer the contents of any accumulator to an I/O device by executing a Data-Out instruction. It can load data from an I/O device into any accumulator by executing a Data-In instruction. To test the status of an I/O device, the 9445 can execute a Skip-On-Status instruction. The I/O cycle has the same timing as the memory cycle (see *Figures 15* and *16)*. Features of the I/O cycle are:

- 250 ns (at 24 MHz system clock) minimum cycle time
- Cycle time can be extended using RDYA, RDYD
- I/O instruction is output at address time
- $\overline{\text{STRBA}}$ is used to latch the I/O instruction
- $\overline{\text{STRBD}}$ is used to strobe the data
- O lines indicate the type of cycle as follows:

	$\overline{M}$	O_1	O_0	$\overline{W}$
I/O Input Execute	1	0	0	1
Instruction Fetch	0	0	0	1
I/O Output Execute	1	0	0	0
Interrupt Save	0	1	1	0

- The I/O devices can interrupt the normal flow of the program by using the common interrupt request line

Instruction Decode

An I/O instruction int he 9445 system comprises several fields as shown in *Figure 7*. This format accommodates data transfers between a CPU accumulator and any one of up to three bidirectional registers in any one of 62 I/O devices. Bits 10 through 15 are coded to represent device codes. The all "1s" device code, 77 octal, is reserved for CPU control instructions and should not be assigned to any unique I/O device; for similar reasons, device code 1 is also reserved. By software convention, device code 0 is also not used.

Fig. 7 Input/Output Instruction Fields

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Op Code = 011			AC Address		Type of Transfer			Control		Device Code					

Bits 3 and 4 specify the address of any accumulator involved in an I/O instruction. When no accumulator is involved, both bits are ignored. The function bits 5, 6, and 7 define the I/O operation to be performed. Bits 8 and 9 control or test the status of the device busy and done flags.

The eight standard I/O instructions were listed previously in the *Instruction Set* section. The No-Input/Output (NIO) instruction is a "no data transfer" instruction that can be used to set the busy and done flags as required, by attaching the appropriate flag-setting mnemonic. The 9445 executes a "dummy" data out transfer. The status of a device's busy and done flags is tested by executing a Skip (SKP) instruction that causes a specific I/O device to put its busy and done flag states on lines $\overline{IB}_0$ and $\overline{IB}_1$ respectively of the common information bus. If the flag state satisfies the condition specified by the busy/done flag-testing mnemonic appended to SKP,the CPU skips the next instruction. The remaining six standard I/O instructions first move data between an accumulator and any one of the device registers, A, B, or C. After the transfer is completed, the busy/done flags are set as specified in the I/O instruction.

There are three I/O instructions that are common to all I/O devices: Interrupt-Acknowledge, Mask-Out, and Clear-I/O Devices. The device code for these three instructions is 77 (octal).

When the 9445 executes the I/O instruction, the $\overline{M}$ and O lines will indicate an I/O operation ("100"). The O lines are valid on the rising edge of SYN. The device address (bits 10–15) must be decoded by each device on the I/O bus. Transfers of information to and from the 9445 are timed with $\overline{STRBD}$ in the same way as the memory cycle.

At the address time, the 9445 outputs the I/O instruction on the information bus. This can be used to generate I/O signals on systems without an I/O controller. $\overline{STRBA}$ is generated and can be used to latch the I/O instruction externally. The interrupt-disable, busy and done flags organize interrupt-driven program-controlled I/O operations. The CPU controls the interrupt-disable flag. Both the CPU and the device can control the busy and done flags.

Device Status Flags

Interrupts from a device are disabled when the interrupt-disable flag of the device is set to "1." Interrupts are enabled when the flag is clear. Interrupt requests are generated whenever the device sets the done flag.

During programmed I/O the busy and done flags define the status of the device for the CPU. The busy and done flag states are coded to represent the indicated device conditions, as follows.

Busy	Done	Device State
0	0	Device idle
1	0	Device busy
0	1	Device completely done
1	1	Device partially done

The sequence of I/O transactions is normally dictated by the speed at which the device can communicate with the CPU. If the CPU operates at a higher speed than a device, the user program should enter a wait loop between each I/O transaction with the device. During execution of the loop, the program repeatedly tests the busy or done flag to determine when the device is ready for the next I/O operation.

During an output operation, one instruction stores data in the desired device register and places the device in the busy state. The program then enters a wait loop which terminates when the device has cleared busy and set done to signal readiness for the next output operation.

To initiate an input transaction, the device sets the done flag. The user program tests this flag and if set it issues one instruction that reads data from the appropriate device register and places the device in the busy state. The program then enters a wait loop which terminates when the device has cleared busy and set done to indicate that it has the next data ready.

Interrupts

The interrupt request, INTREQ, line is common to all I/O devices. When the device completes an I/O operation, it should set the done flag. Concurrently, if the device is enabled to interrupt, it should assert the active LOW on the INTREQ line. The processor responds to the interrupt request after completing execution of the current instruction. It then clears the interrupt-on flag so no further interrupts can be started, saves PC (which points to the next instruction) in location 0, and simulates a "jump-indirect-through-location-1" instruction to jump to the interrupt service routine. Location 1 should contain the address of the interrupt service routine or an indirect address to the routine. The 9445, when interrupted, can check for the source of the interrupt in two ways:

It can test the state of the done flags in various devices, one by one, by executing Skip-on-Done instructions; or

It can test the state of the I/O devices by executing the Interrupt-Acknowledge instruction, causing the highest priority device that had sent an interrupt request to respond by placing its device code on bits 10 through 15 of the Information Bus.

Since several devices can request interrupt simultaneously, device priority may be established in a daisy-chain fashion by a physical connection of a serially propagated signal, Interrupt Priority. The first device requesting an interrupt and having its Interrupt-Priority-In line HIGH has priority, and it answers the Interrupt-Acknowledge instruction, at the same time blocking the propagation of the interrupt-priority signal by putting its Interrupt-Priority-Out line in a LOW state.

The interrupt-priority signal is generated in the device having the highest priority. The 9445 can disable the interrupt system in each I/O device by placing a mask on the information bus while executing the Mask-Out instruction.

Each device is assigned a specific bit in the mask. When that bit is "1," the interrupt system is disabled. A "0" in that bit enables the device.

After servicing a device, the routine should restore the preinterrupt states of the accumulators and carry, turn on the interrupts, and jump to the interrupted program. The instruction that enables the interrupt sets the interrupt-on flag (INTON), but the flag has no effect until the next instruction begins. Thus, following the instruction that turns the interrupt back on, the processor always executes one more instruction (assumed to be the return to the interrupted program) before another interrupt service can start.

The interrupt request input INTREQ is negative-level sensitive and is synchronized in the processor. Externally, interrupt requests may be latched with the leading edge of SYN. The interrupt service routine should issue an I/O instruction to clear the done flag of the interrupting device before re-enabling interrupts. The clearing of the DONE flag should in turn clear the interrupt request within the device.

The 9445 recognizes two other types of interrupts:

Abort Interrupt — This is activated by the active LOW of the ABORT input.The processor responds by:

Aborting the instruction being executed,
Storing the address of the aborted instruction in location 46 (octal), and
Jumping indirect through location 47 (octal).

Stack Overflow Interrupt — This is an internal interrupt caused when the stack overflows; i.e., when a stack operation (PSHA, PSHF, PSHR, SAVE, TOPW) writes over a 256 decimal word-page boundary (mod 400 octal). This interrupt is of higher priority than the external interrupt (INTREQ); the processor responds, at completion of the current instruction by:

Clearing the interrupt-on flag (to "0"),
Storing the updated program counter in location 0, and
Jumping indirect through location 3 (octal).

The interrupt-save cycle follows the interrupt. It can be externally detected by the code "011" on the O lines and used, for example, to switch an external mapper such as the 9444, to nonmapped mode.

The order of priority of requests and interrupts, from highest to lowest, is as follows: MR, ABORT, DCHREQ, Stack Overflow Interrupt, INTREQ, and CONREQ.

Type of Interruption	Save address	Vector
Interrupt Request	0	1
Stack Fault	0	3
Abort	46	47
TRAP	46	47

Program interruption save addresses and vectors
(all addresses in octal)

Data Channel

The data channel has three methods of operation

Data-channel cycle with 9445 timing the memory,
Data-channel cycle with external memory timing, and
Autonomous-bus cycle using bus arbitration scheme.

The sequence of events during a data-channel cycle is as follows:

1. $\overline{\text{DCHREQ}}$ is set.
2. 9445 responds by setting $\overline{M}$, O_1, and O_0 to "101" and $\overline{\text{BUSREQ}}$ to "0" (Active). This is recognized externally as Data Channel Acknowledge and can be used to reset $\overline{\text{DCHREQ}}$ if it is the last data-channel cycle required. W is Tri-stated at the same time so that it can be driven by the data channel device.
3. 9445 3-states the bus and sends $\overline{\text{STRBA}}$.
4. The external logic must supply an address at this time. The address time can be extended with RDYA.
5. 9445 outputs $\overline{\text{STRBD}}$.
6. The controller transmits or receives the data-channel data and responds with RDYD, concluding the cycle.

Console Operation

Console operation allows examination and modification of the 9445 internal registers without executing programs in main memory. This is very useful for system diagnostics even when the memory or I/O part of the microcomputer system is not fully functional.

Upon request for console operation, the processor will execute one of a number of console operations depending on a console code on the Information Bus
This facilitates the connection of an external console for monitoring and test purposes. The following sequence is used to execute a console operation:

1. $\overline{\text{CONREQ}}$ is set LOW.
2. The processor finishes the current instruction.
3. The processor sets the $\overline{M}$ and O lines to "110" (Console code In).
4. In response to the $\overline{M}$ and O lines being set to "110," the console logic supplies a code on the Information Bus corresponding to the desired operation, which is selected onto the bus with $\overline{\text{STRBD}}$.
5. The console logic resets $\overline{\text{CONREQ}}$. Even though $\overline{\text{STRBA}}$ is not active in this cycle, the 9445 can be stalled by RDYA.
6. The processor executes the console operation.
7. The processor may read or write data from the console switches or console lamps. In this case, the $\overline{M}$ and O lines are set to "111" (console data). In most cases, the processor halts after the console operation by entering a Wait state. The exceptions are Continue and APL.

Console logic can be implemented in three levels of simplicity:

No Console Code — If a $\overline{\text{CONREQ}}$ is generated and no console code supplied, the default bus value ("0") will cause the processor to execute APL. This sets the PC to −1, then starts normal execution. This is the minimal console operation required. See *Figure 4* for APL starting addresses.

Limited Console Operation — A subset of operations can be arranged with a 2-bit console code. These operations are APL, Test, Continue, and Halt.

Full Console Operation — A 9-bit code (see *Table 1*) defines the full set of console operations. Single-Step is not implemented directly, but can be arranged using Continue: first, the Continue operation is specified; after the first instruction is fetched, a new $\overline{\text{CONREQ}}$ is generated and the operation is changed to Halt.

Self-Test

The response of the 9445 to the SELF-TEST console function is to perform the following operations:

1. AC1 <— −1
2. AC0 <— 0
3. AC2 <— −AC1 (1)
4. AC0 & AC1 <— AC1 × AC2 (−1)
5. BR <— AC2 (1)
6. PC <— BR + 1 (2)
7. BR <— AC0 (−1)
8. (PC) <— BR ((2) <— −1)
   ((PC) is the memory location pointed at by the PC)

Successful operation of the self-test results in a value of −1 being written into memory location 2. Any other value or a write to any other location indicates a failure of the self-test.

Table 1 Console Codes

Information Bus (IB)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
—	—	—	—	—	0	0	—	—	—	—	—	—	—	—	—	AUTOMATIC PROGRAM LOAD
—	—	—	—	—	1	0	—	—	—	—	—	—	—	—	—	CONTINUE
—	—	—	—	—	1	1	—	—	—	—	—	—	—	—	—	STOP
—	—	—	—	—	0	1	0	0	0	—	—	—	—	—	—	SELF TEST
—	—	—	—	—	0	1	0	0	1	—	—	—	—	—	—	EXAMINE NEXT MEMORY
—	—	—	—	—	0	1	0	1	1	—	—	—	—	—	—	DEPOSIT NEXT MEMORY
0	—	0	0	0	0	1	1	0	0	—	—	—	—	—	—	EXAMINE AC0
0	—	0	0	1	0	1	1	0	0	—	—	—	—	—	—	EXAMINE AC1
0	—	0	1	0	0	1	1	0	0	—	—	—	—	—	—	EXAMINE AC2
0	—	0	1	1	0	1	1	0	0	—	—	—	—	—	—	EXAMINE AC3
0	—	1	0	0	0	1	1	0	0	—	—	—	—	—	—	EXAMINE SP
0	—	1	0	1	0	1	1	0	0	—	—	—	—	—	—	EXAMINE FP
1	—	—	—	—	0	1	1	0	0	—	—	—	—	—	—	EXAMINE PC
—	—	—	—	—	0	1	1	0	1	—	—	—	—	—	—	EXAMINE MEMORY
0	—	0	0	0	0	1	1	1	0	—	—	—	—	—	—	DEPOSIT AC0
0	—	0	0	1	0	1	1	1	0	—	—	—	—	—	—	DEPOSIT AC1
0	—	0	1	0	0	1	1	1	0	—	—	—	—	—	—	DEPOSIT AC2
0	—	0	1	1	0	1	1	1	0	—	—	—	—	—	—	DEPOSIT AC3
0	—	1	0	0	0	1	1	1	0	—	—	—	—	—	—	DEPOSIT SP
0	—	1	0	1	0	1	1	1	0	—	—	—	—	—	—	DEPOSIT FP
1	—	—	—	—	0	1	1	1	0	—	—	—	—	—	—	DEPOSIT PC
—	—	—	—	—	0	1	1	1	1	—	—	—	—	—	—	DEPOSIT MEMORY

1 = Low on corresponding bit IB_n
0 = High on corresponding bit IB_n
— = Don't care

Bus Arbitration

The 9445 contains three signals that allow more than one processor to share a common bus:

$\overline{\text{BUSREQ}}$ — This is LOW at the beginning of every cycle in which the 9445 requires use of the bus.

BUSGNT — When LOW, it is used to halt tha processor indicating the bus is unavailable.

$\overline{\text{BUSLOCK}}$ — This indicates that the current bus cycle and the following bus cycle from the processor must not be interrupted by a cycle from another processor.

The $\overline{\text{BUSLOCK}}$ signal has two purposes. One purpose is to prevent the external memory address register from being overwritten during those instructions that rely on the address remaining in this register. The other purpose is to provide a method of synchronizing separate software tasks using a standard semaphore system. An external arbiter is required to determine which processor has access to the bus.

Fig. 13 ALU Cycle Timing

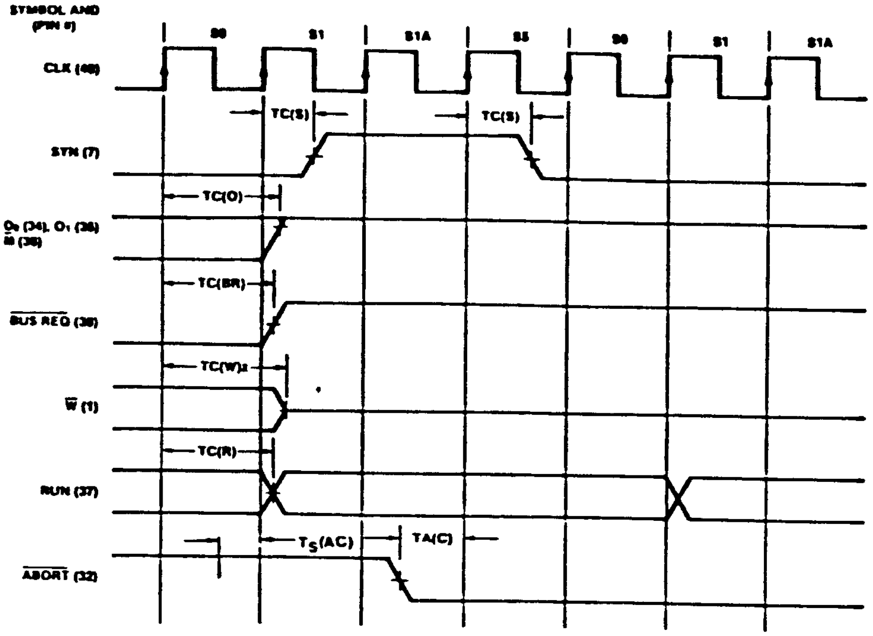


Fig. 14 Minimum Memory Cycle Timing

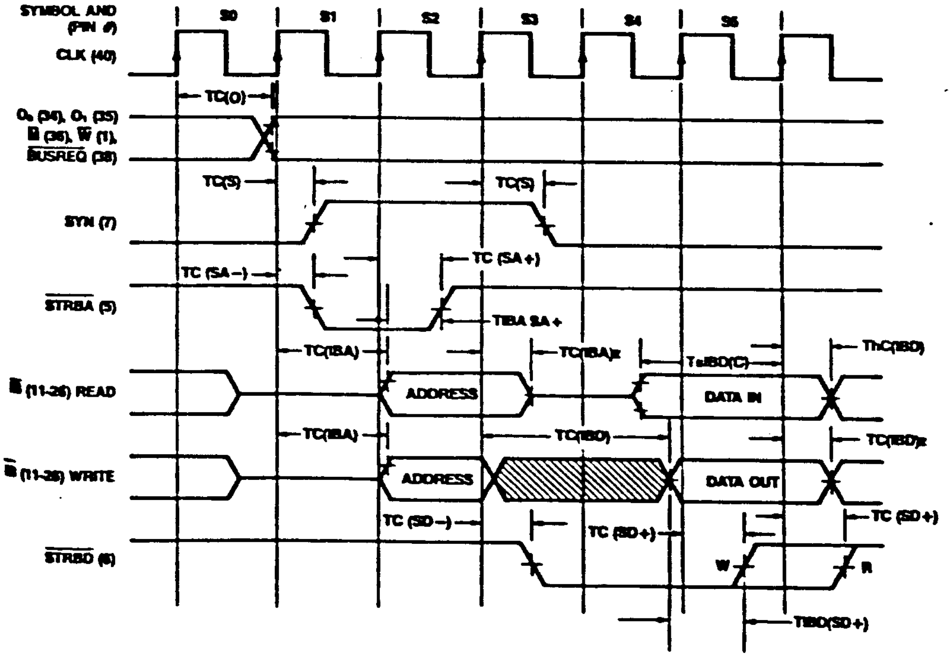


Fig. 15 Extended Memory Cycle Timing

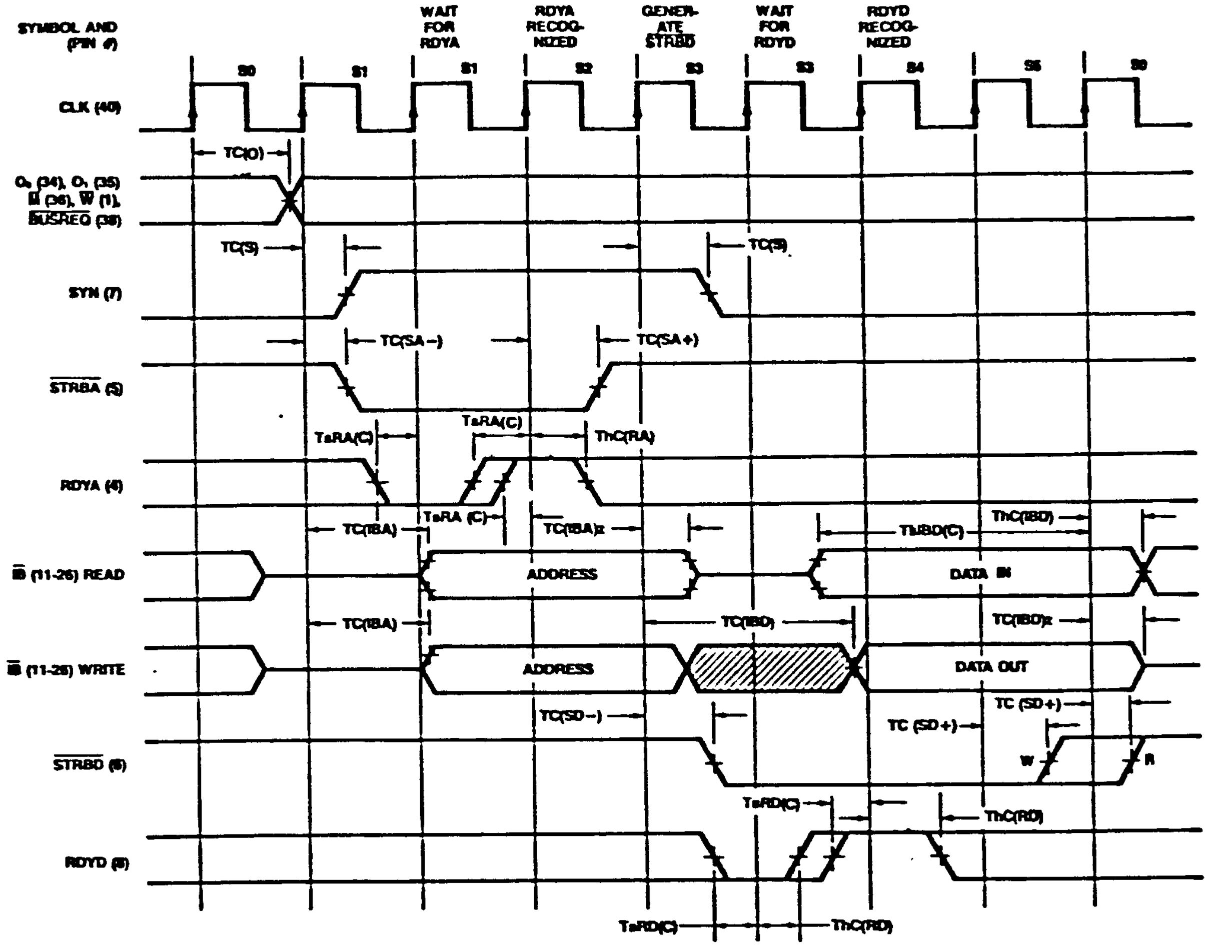
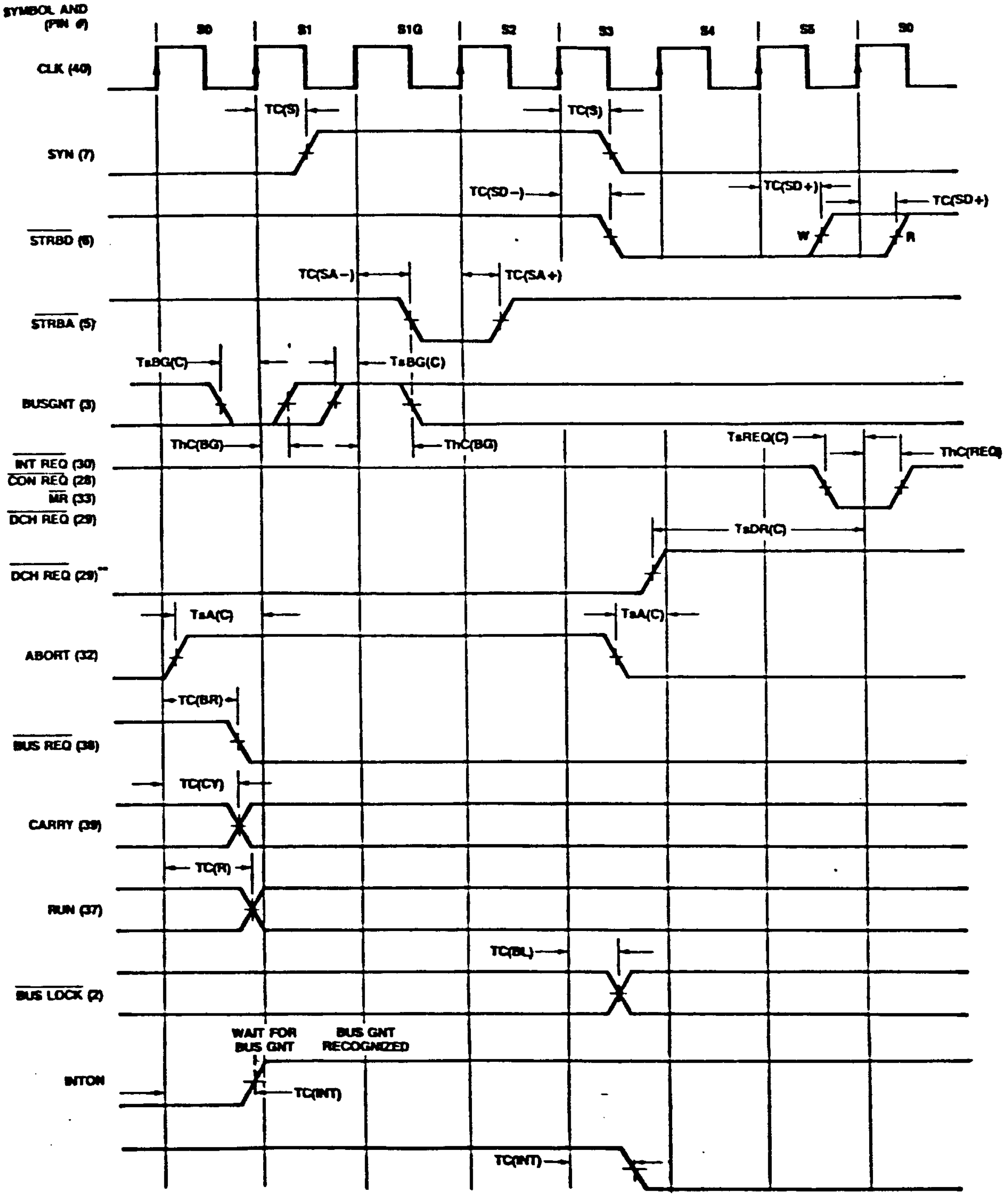


Fig.16 Bus and Status Control Timing



Guaranteed Operating Ranges

Supply Voltage (V_{CC})			Case Temperature
Min	Typ	Max	
4.75 V	5.0 V	5.25 V	0 to 70°C

DC Characteristics

(Over guaranteed operating ranges unless otherwise noted.) I_{INJ}(min) = 3.75 mA: I_{INJ}(max) = 425 mA

Symbol	Characteristic	Min	Typ	Max	Unit	Test Conditions
V_{IH}	Input HIGH Voltage	2.0			V	Guaranteed Input HIGH Voltage
V_{IL}	Input LOW Voltage			0.8	V	Guaranteed Input LOW Voltage
V_{CD}	Input Clamp Diode Voltage		–0.9	–1.5	V	V_{CC} = Min, I_{IN} = –18 mA, I_{INJ} = Min
V_{OH}	Output High Voltage; RUN, CARRY, INTON, SYN, $\overline{STRBD}$, $\overline{BUSREQ}$, $\overline{STRBA}$, O_0, O_1, $\overline{M}$	2.4	3.4		V	V_{CC} = Min, I_{OH} = –400 µA, I_{INJ} = Min
V_{OH}	Output HIGH Voltage; $\overline{IB}_{0-15}$, $\overline{W}$	2.4	3.4		V	V_{CC} = Min, I_{OH} = –1.0 mA, I_{INJ} = Min
V_{OL}	Output LOW Voltage		0.25	0.5	V	V_{CC} = Min, I_{OL} = 8.0 mA, I_{INJ} = Min
I_{IH}	Input HIGH Current; $\overline{DCHREQ}$, $\overline{INTREQ}$, CLK, $\overline{MR}$, RDYA, RDYD, $\overline{ABORT}$, $\overline{CONREQ}$, BUSGNT		2.0	4.0	µA	V_{CC} = Max, V_{IN} = 2.7 V, I_{INJ} = Min
I_{IH}	Input HIGH Current; $\overline{IB}_{0-15}$ (3-state)		5.0	100	µA	V_{CC} = Max, V_{IN} = 2.7 V I_{INJ} = Min
I_{IH}	Input HIGH Current; All Inputs			1.0	mA	V_{CC} = Max, V_{IN} = 5.5 V, I_{INJ} = Min
I_{IL}	Input LOW Current; All Inputs		–0.21	–0.4	mA	V_{CC} = Max, V_{IN} = 0.4 V, I_{INJ} = Min
I_{OZH}	Output OFF State (High Impedance) Current; $\overline{IB}_{0-15}$, 0			100	µA	V_{CC} = Max, V_{IN} = 2.4 V, I_{INJ} = Min
I_{OZL}	Output OFF State (High Impedance) Current; $\overline{IB}_{0-15}$		–210	–400	µA	V_{CC} = Max, V_{OUT} = 0.4 V, I_{INJ} = Min
I_{OZL}	Output OFF State (High Impedance) Current; $\overline{W}$			–100	µA	V_{CC} = Max, V_{OUT} = 0.5 V, I_{INJ} = Min
I_{OSH}	Output Short Circuit Current; All Outputs Except $\overline{BUSLOCK}$	–15		–100	mA	V_{CC} = Max, V_{OUT} = 0.0 V I_{INJ} = Min (Note 2)
I_{LOH}	Output Leakage; $\overline{BUSLOCK}$			1.0	mA	V_{CC} = Min, V_{OH} = 5.25 V, I_{INJ} = Min
I_{CC}	Supply Current		200	350	mA	V_{CC} = Max, I_{INJ} = Min
V_{INJ}	Injector Voltage	.5	1.3	1.7	V	I_{INJ} = 400 mA

AC Characteristics — Commercial

INPUT SIGNALS:
amplitude = 0 to 3 V;
rise and fall time = 6 ns,
measurements are taken at the 1.5 V level

Clock pulse width (CPW) = 45% to 55% of CPRD
Clock period (CPRD) = 1/Frequency
TMEM = 5 CPRD − TC(IBA) − T_sIBD(C)
All times in nanoseconds

	Max Clock Frequency
9445-16A	16 MHz
9445-18A	18 MHz
9445-20A	20 MHz
9445-24A	24 MHz

T_c = 0 to +70°C V_{CC} = 5 V ±5%		9445-16, 9445-18		9445-20		9445-24	
Symbol	**Propagation Delay From Clock**	**Min**	**Max**	**Min**	**Max**	**Typ**	
TC(O)	$\overline{M}$, O_0, O_1		70		70	48	
TC(S)	SYN		50		50	30	
TC(W)	$\overline{W}$		105		105	60	
TC(W)z	$\overline{W}$ to TRI-Z		125		125	66	
TC(IBA)	$\overline{IB}_{0\text{-}15}$ ADDRESS		85		85	40	
TC(IBA)z	$\overline{IB}_{0\text{-}15}$ ADDR TRI-Z		★ 60		★ 55	40	
TC(SA−)	$\overline{\text{STRBA}}$ LOW		48		48	25	
TC(SA+)	$\overline{\text{STRBA}}$ HIGH		50		50	25	
TC(IBD)	$\overline{IB}_{0\text{-}15}$ DATA		110		110	60	
TC(IBD)z	$\overline{IB}_{0\text{-}15}$ DATA TRI-Z		60		60	40	
TC(SD−)	$\overline{\text{STRBD}}$ LOW		36		36	25	
TC(SD+)	$\overline{\text{STRBD}}$ HIGH		50		50	25	
TC(BL)	$\overline{\text{BUS LOCK}}$		65		65		
TC(R)	RUN		100		100	70	
TC(CY)	CARRY		80		80	50	
TC(INT)	INTON		105		105		
TC(BR)	$\overline{\text{BUSREQ}}$		65		65	40	
Symbol	**Setup Time Before Clock Hold Time After Clock**	**Setup Min**	**Hold Min**	**Setup Min**	**Hold Min**	**Setup Typ**	**Hold Typ**
TsRA(C), ThC(RA)	RDYA	15	15	15	15		
TsRD(C), ThC(RD)	RDYD	15	15	15	15		
TsIBD(C), ThC(IBD)	$\overline{IB}_{0\text{-}15}$ DATA IN	★ 100	32	★ 90	32	50	25
TsREQ(C), ThC(REQ)	$\overline{\text{INTREQ}}$, $\overline{\text{DCHREQ}}$, $\overline{\text{CONREQ}}$	25	30	25	30		
TsMR(C), ThC(MR)	$\overline{\text{MR}}$	25	50	25	50		
TsDR(C), ThC(DR)	$\overline{\text{DCHREQ}}$ OFF	130	—	130	—		
TsA(C), ThC(A)	$\overline{\text{ABORT}}$	30	—	30	—		—
TsBG(C), ThC(BG)	BUSGNT	20	20	20	20		
Symbol	**Other Propagation Delays**	**Min**	**Max**	**Min**	**Max**	**Min**	**Max**
TIBA(SA+)	ADDR VALID to $\overline{\text{STRBA}}$ HIGH	CPRD-50		CPRD-50			
TIBD(SD+)	DATA VALID to $\overline{\text{STRBD}}$ HIGH	2 CPRD-63		2 CPRD-63			

Notes

1. Memory or I/O system time is the maximum allowable time from valid address to resulting valid data without requiring added wait states. If additional time is required for either memory or I/O to respond, it can be obtained merely by adding wait states in either the address or data time.

Instruction Execution Times

Instruction	Clock Cycles	Execution Times				Notes
		16 MHz	18 MHz	20 MHz	24 MHz	
COM	m	0.375	0.333	0.3	0.25	
NEG	m	0.375	0.333	0.3	0.25	
MOV	m	0.375	0.333	0.3	0.25	
INC	m	0.375	0.333	0.3	0.25	
ADC	m	0.375	0.333	0.3	0.25	
SUB	m	0.375	0.333	0.3	0.25	
ADD	m	0.375	0.333	0.3	0.25	
AND	m	0.375	0.333	0.3	0.25	
OR	m	0.375	0.333	0.3	0.25	
MUL	m+18s	4.875	4.332	3.900	2.917	
MULS	m+18s	4.875	4.332	3.900	2.917	
DIV (Normal)	m+19s	5.125	4.551	4.100	3.404	
DIV (Overflow)	m+2s	0.875	0.777	0.7	0.583	
DIVS (Normal)	m+20s	5.375	4.773	4.3	3.57	
DIVS (Overflow)	m+2s/21s	0.875/5.625	0.777/4.995	0.7/4.5	0.582/3.736	
NORM	m + s + ns	0.625 + 0.25n	0.555 + 0.222n	0.5 + 0.2n	0.417 + 0.167n	n = number of steps needed for normalization
SLLD	m + s + ns	0.625 + 0.25n	0.555 + 0.222n	0.5 + 0.2n	0.417 + 0.167n	n = number of shifts.
SALD	m + s + ns	0.625 + 0.25n	0.555 + 0.222n	0.5 + 0.2n	0.417 + 0.167n	
SARD	m + s + ns	0.625 + 0.25n	0.555 + 0.222n	0.5 + 0.2n	0.417 + 0.167n	
SLRD	m + s + ns	0.625 + 0.25n	0.555 + 0.222n	0.5 + 0.2n	0.417 + 0.167n	
SKNV	m + 2s	0.875	0.777	0.70	0.583	
JMP	m	0.375	0.333	0.3	0.25	Times for page-zero addressing. Add m for indirect. Add m for auto-increment/decrement.
JSR	m	0.375	0.333	0.3	0.25	
ISZ	3m	1.125	0.999	0.9	0.75	
DSZ	3m	1.125	0.999	0.9	0.75	
LDA	2m	0.750	0.666	0.6	0.5	
STA	2m	0.750	0.666	0.6	0.5	
LDB	2m	0.75	0.666	0.6	0.5	
STB	3m	1.125	0.999	0.9	0.75	
PSHA	2m+s	1.0	0.888	0.8	0.667	
POPA	2m	0.75	0.666	0.6	0.5	
PSHF	2m+s	1.0	0.888	0.8	0.667	
POPF	2m	0.75	0.666	0.6	0.5	
POPJ	2m	0.75	0.666	0.6	0.5	
PSHR	2m+2s	1.125	1.11	1.00	.8334	
TOPR	2m+s	1.0	0.888	0.8	0.667	
TOPW	2m+s	1.0	0.888	0.8	0.667	
MTSP	m+s	0.625	0.555	0.5	0.4167	
MTFP	m+s	0.625	0.555	0.5	0.4167	
MFSP	m	0.375	0.333	0.3	0.25	
MFFP	m	0.375	0.333	0.3	0.25	
SAV	6m+8s	4.25	3.774	3.4	2.834	
RET	6m+2s	2.75	2.442	2.2	1.8334	
DSP	m+s	0.625	0.555	0.5	0.4167	
NIO	m + i	0.75	0.666	0.6	0.5	
SKP I/O DEVICE	m + i + s	1.0	0.888	0.8	0.667	
DIA/B/C	m + i	0.75	0.666	0.6	0.5	
DOA/B/C	m + i	0.75	0.666	0.6	0.5	
ETRP	m + s	0.625	0.555	0.5	0.417	
DTRP	m + s	0.625	0.555	0.5	0.417	
E64K	m+2s	0.875	0.777	0.7	0.583	
D64K	m + s	0.625	0.555	0.5	0.417	Definitions:
HALT	s	0.25	0.222	0.2	0.167	m = 6 + Wm
WAIT	s	0.25	0.222	0.2	0.167	i = 6 + Wi
INTEN/INTDS	m+2s	0.875	0.777	0.7	0.583	s = 4 Wm = number of memory wait states
SKP CPU (Inton)	m + 2s	0.875	0.777	0.7	0.583	Wi = number of I/O wait states

Note: Times are given for specific clock frequencies. The clock may be operated from >0 to the specific maximum frequency within the specified temperature and voltage range. Execution times assume zero wait states.