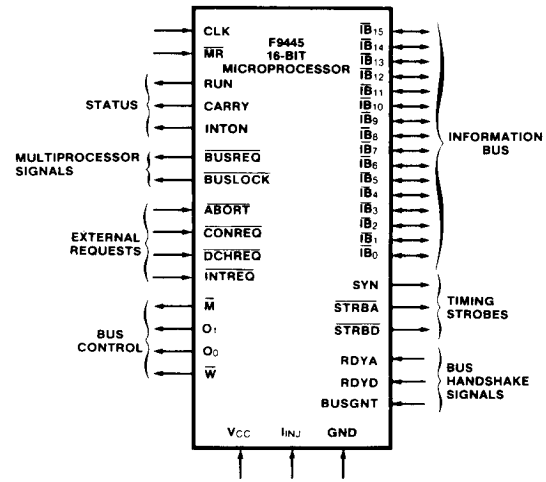
F9445
16-Bit Bipolar Microprocessor

Description

The F9445 is a 16-bit microprocessor implemented using Fairchild's Isoplanar Integrated Injection Logic ($I^3L^{®}$) technology. This bipolar technology and a sophisticated pipeline architecture combine to give the F9445 very fast execution times. The processor has eight program-accessible registers and the capability of directly addressing 128K bytes (64K words) of memory. Up to 4M bytes of physical memory may be accessed using the F9444 memory management unit. The F9445 can address 62 I/O devices, handle 16 levels of priority interrupt, and perform fast direct memory access. It has control lines to provide operator-console functions and has an on-chip self-test program. The F9445 CPU is supported with a comprehensive family of LSI support circuits to permit cost and performance effective usage in high-performance microcomputer systems. The support circuits include the F9446 Dynamic Memory Controller, F9447 I/O Controller, F9448 Programmable Multiport Interface, F9449 Multiple Data Channel Controller, F9444 Memory Management Unit and F9470 Console Controller. It is also supported with a library of software packages, including editors, debuggers, macro-assembler, relocating loader, real-time executive, interactive multi-user disk operating system and utilities, as well as high-level languages: FORTRAN, BASIC and PASCAL.

- **Advanced Parallel Architecture Leading to Very Fast Execution Times—250 ns Register to Register, 2.9 μs 16 x 16 Bit Multiply**
- **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 Capabilities**
- **Flexible Operator-Control Functions and Self-Test**
- **Static Operation with Single Clock up to 24 MHz**
- **LS TTL Input/Output Structure with I^3L Internal Circuits**
- **40-Pin DIP Needing a Single +5 V Power Supply**
- **Full Military Temperature and Voltage Ranges**
- **Radiation-Tolerant Technology**
- **Comprehensive Family of Support Circuits**

Pin Functions



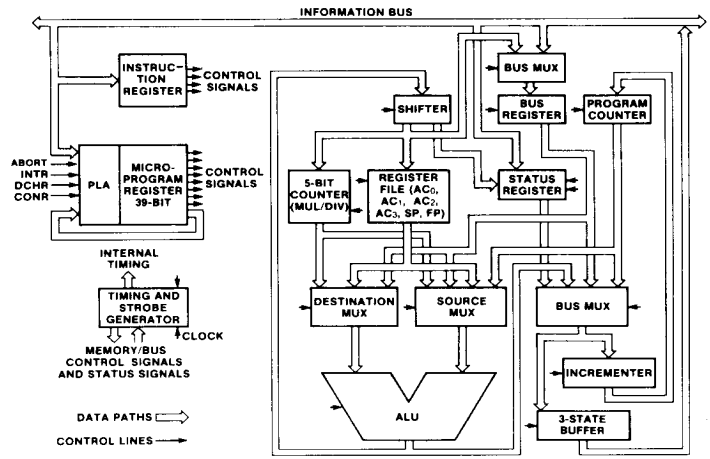
Absolute Maximum Ratings

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
Output Current (dc) (Output LOW)	+20 mA
Injector Current (I_{INJ})	+450 mA
Injector Voltage (V_{INJ})	−0.5 to +1.5 V

6

Fig. 1 F9445 Functional Diagram



Architecture

The F9445 microprocessor comprises three main blocks: the data path, the control unit, and the timing generator.

Data Path

The data path is 16 bits wide and is responsible for all the processing of data and address in the system. In many cases, data and address may be processed simultaneously.

The data path includes the following blocks (see *Figure 1*):

Register File (AC0, AC1, AC2, AC3, SP, FP)
Program Counter (PC)
Program Status Word or Status Register (PSW containing: Carry, Overflow, 32KW, ETRP flags)
Interrupt-On Flip-Flop (INTON)
Destination Mux
Source Mux
16-Bit ALU
17-Bit Shifter
5-Bit Counter (for multicycle instructions)
Bus Register Mux
Bus Register
Bus Mux and Buffer
Incrementer

Control Unit

The operations of the data path components are governed by the pipelined, microprogrammed control unit. This unit comprises three main elements (see *Figure 1*): the PLA (control store) to contain the microprogram, the pipeline register (microprogram register) to latch the microinstruction executed in the current cycle, and the instruction register to supply additional control bits during certain instructions. In addition, the control unit has a machine instruction pre-fetch mechanism which overlaps the fetching of the next instruction from memory during execution of short-cycle instructions, such as arithmetic-and-logic (ALU) instructions. This pre-fetch capability and the microprogram pipeline give the F9445 very fast and efficient instruction execution.

Timing Generator

The timing generator produces the system timings for the F9445 internal registers, memory, I/O, and console.

The clock is divided on-chip using a 3-bit twisted ring counter. The divide ratio is 6:1 or 4:1, depending on whether a short or long cycle is required. The long cycle can be extended indefinitely by lowering the inputs BUSGNT, RDYA, or RDYD. These signals hold the processor in state S1 (using BUSGNT or RDYA) or S3 (using RDYD) until the inputs are raised.

The twisted ring counter is also used to generate all the strobes by a combinational decode of its outputs and certain bits of the microprogram register.

Signal Descriptions

All F9445 inputs and outputs are TTL.

Information Bus

$\overline{IB}_0$ through $\overline{IB}_{15}$, Pins 11 through 26 — 16-bit Bus — Active LOW bidirectional; $\overline{IB}_0$ 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 — 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.

$\overline{\text{STRBA}}$, 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.

$\overline{\text{M}}$, 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.

	$\overline{M}$	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

If a skip is taken on an arithmetic-and-logic (ALU) instruction, the next instruction is fetched but not executed. In such fetches, the $\overline{\text{M}}$ and O lines will indicate the following states.

$\overline{M}$	O_1	O_0	State Indicated
0	0	0	S0 through S4
0	0	1	S5

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 ($\overline{\text{BUSREQ}}$ is HIGH).

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.

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

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.

$\overline{\text{BUSLOCK}}$, 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.

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

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; if that cycle is a write, it is inhibited (changed to read); sets the F9445 to 32K mode with trap enabled.

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

$\overline{\text{CONREQ}}$, 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, after the current instruction.

$\overline{\text{ABORT}}$, 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.

I_{INJ}, Pin 10 — Injection Current Input — Operates in 200-400 mA range at approximately 1 V; requires > 350 mA for maximum speed.

Register Set

The F9445 has eight user-accessible registers (see Figure 2), including seven 16-bit registers and a program status word (PSW) containing 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 comprise a 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, or 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 which 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

Fig. 2 F9445 Register Model

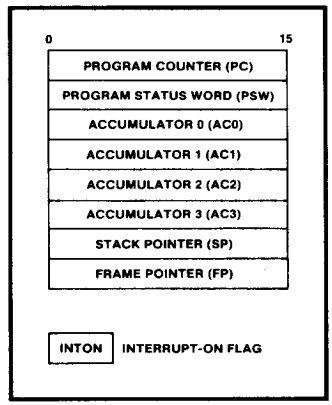
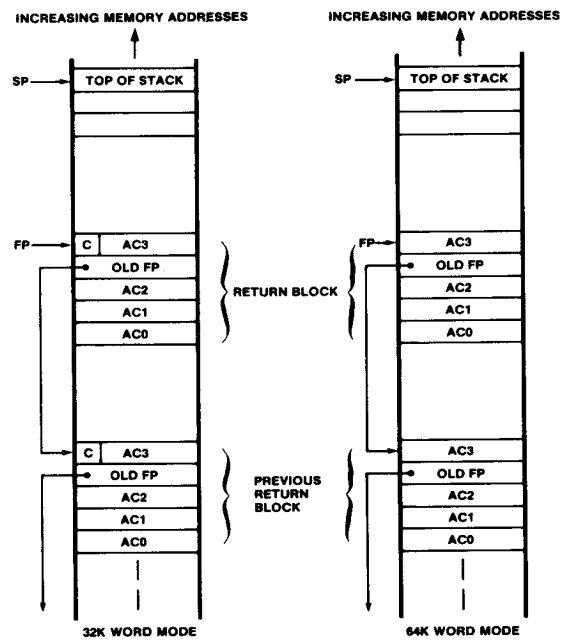


Fig. 3 Data Organization in a Stack (LIFO)



the result of the ALU operation as the base value of carry. Accumulators AC2 and AC3 also serve as index registers during memory addressing operations. 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 and 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 (see *Figure 3*).

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 PC+1 (and the value of the carry bit in 32K-word mode only) is stored in AC3. 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. 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.

Addressing Ranges and Modes

The F9445 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 F9445 can operate are 128K-byte (64K-word) or 64K-byte (32K-word) logical address space. The F9445 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) Address Range

After the master reset is activated or the D64K instruction is executed, the F9445 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 addressing instructions.

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 (11) 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 which 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 which 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". Such multiple levels of indirect addressing are only allowed in the 32K-word address range operations.

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 F9445 starts operating with the 128K-byte (64K-word) addressing range. In this range, the F9445 uses 16-bit addresses to fetch up to 64K words from the memory and supports all the 11 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 to be loaded or stored.

Instruction Set

The F9445 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.

Input/Output Operations

Input/output devices can transfer data to the F9445-based microcomputer via:

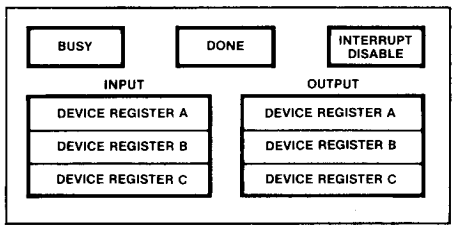
Programmed I/O using the I/O instructions of the F9445,

Memory-mapped I/O using the load/store instructions of the F9445, or

Direct memory access or 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 4*). The 2-bit status word comprised 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 F9445 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. 4 I/O Device Model



The F9445 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 F9445 can execute a Skip-On-Status instruction.The I/O cycle has the same timing as the memory cycle (see *Figures 13* and *14*). 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:

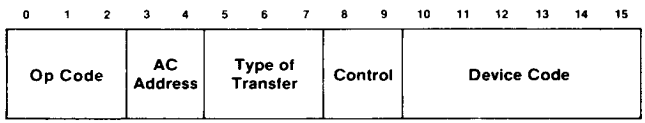
	$\overline{M}$	O_1	O_0	$\overline{W}$
I/O Input Execute	1	0	0	1
Instruction Fetch	0	0	0	1
I/O Output Execute	1	0	0	0
Interrupt Save	0	1	1	0

- 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 in the F9445 system comprises several fields as shown in *Figure 5*. 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 00 through 76 (octal). 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 convention, device code 0 is not used.

Fig. 5 Input/Output Instruction Fields



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 description of the introduction to this 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 F9445 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 IB_0 and IB_1 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 F9445 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 F9445 are timed with $\overline{STRBD}$ in the same way as the memory cycle.

At the address time, the F9445 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 interrupt-disable flag is normally set to disable interrupts, and 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, it enters a wait loop between each I/O transaction with the device. During execution of the loop, the CPU repeatedly monitors 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 CPU 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. One instruction reads data from the appropriate device register and places the device in the busy state. The CPU 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, $\overline{\text{INTREQ}}$, 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 $\overline{\text{INTREQ}}$ 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 executes a "jump-indirect-to-location-1" instruction to jump to the interrupt service routine. Location 1 should contain the address of the interrupt routine or an indirect address to the routine. The F9445, when interrupted, can check for the source of the interrupt in two ways:

It can test the state of the done flags in the 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 device that had sent an interrupt request to respond by placing its device code on bits 10 through 15 of the information bus.

As 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 F9445 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 bit in the mask is assigned to a specific device. 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 pre-interrupt states of the accumulators and carry, turn on the interrupt, and jump to the interrupted program. The instruction that enables the interrupt sets interrupt on (INTON), but the flag has no effect until the next instructior begins. Thus, after 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. If the service routine allows interrupts by higher priority devices, the routine should turn off the interrupt, before dismissing as indicated above, to prevent further interrupts during dismissal. In dismissing, the routine should re-enable lower priority devices.

The interrupt request input $\overline{\text{INTREQ}}$ is negative-level sensitive and is synchronized in the processor. Externally, interrupt requests may be latched with the leading edge of SYN. The interrupt request may be reset by the external I/O controller from a decode of the I/O instruction INTA.

The F9445 recognizes two other types of interrupts:

Abort Interrupt — This is activated by the active LOW of the $\overline{\text{ABORT}}$ input. The processor responds by:

Aborting the instruction being executed,
Storing the address of the aborted instruction in location 46 (octal), and
Jumping indirect to 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 page boundary (mod 256). This interrupt is of higher priority than the external interrupt ($\overline{\text{INTREQ}}$); 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 to 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 to non-mapped mode.

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 $\overline{\text{CONREQ}}$.

Data Channel
The data channel has three methods of operation with the F9445:

Data-channel cycle with F9445 controlling the memory, Data-channel cycle with external memory control, 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. F9445 responds by setting $\overline{M}$, O_1, and O_0 to "101" and $\overline{\text{BUSREQ}}$ to "1". 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.
3. F9445 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. F9445 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 F9445 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 (see *Figure 6*). 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 $\overline{\text{CONREQ}}$.
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.

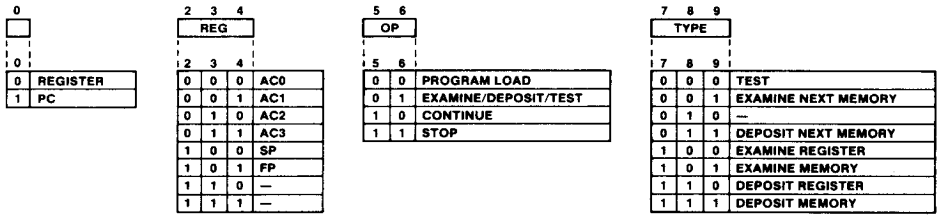
Console logic can be implemented in three levels of simplicity:

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.

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 *Figure 6*) 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.

Fig. 6 Console Codes



0
0 REGISTER
1 PC

2 3 4 (REG)	
0 0 0	AC0
0 0 1	AC1
0 1 0	AC2
0 1 1	AC3
1 0 0	SP
1 0 1	FP
1 1 0	—
1 1 1	—

5 6 (OP)	
0 0	PROGRAM LOAD
0 1	EXAMINE/DEPOSIT/TEST
1 0	CONTINUE
1 1	STOP

7 8 9 (TYPE)	
0 0 0	TEST
0 0 1	EXAMINE NEXT MEMORY
0 1 0	—
0 1 1	DEPOSIT NEXT MEMORY
1 0 0	EXAMINE REGISTER
1 0 1	EXAMINE MEMORY
1 1 0	DEPOSIT REGISTER
1 1 1	DEPOSIT MEMORY

Bus Arbitration

The F9445 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 F9445 requires use of the bus.

BUSGNT—When LOW, it is used to halt the 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.

Applications

Static Memory Interface

The F9445 bus structure allows easy connection of static memory. Both address and data are multiplexed onto the 16-bit information bus $\overline{\text{IB}}_{(0\text{-}15)}$. The mutually exclusive signals $\overline{\text{STRBA}}$ and $\overline{\text{STRBD}}$ indicate that the information bus is carrying address or data, respectively. The $\overline{\text{M}}$ signal ($\overline{\text{M}}$ = LOW) indicates that a memory cycle is taking place on the bus, while the $\overline{\text{W}}$ signal indicates whether the operation is a read or write. The timing of the $\overline{\text{STRBD}}$ is shorter for a write operation, to allow positive hold time for the memories. The signal RDYD may be held LOW to stretch the memory cycles for slow memories.

A typical scheme is shown in *Figure 7.* This diagram shows a 4K × 16 static RAM configuration (2114-type 1K × 4). The bus is buffered by a 74240 inverting 3-state buffer. Buffering is optional and depends on fan-out requirements of the memories. The buffer is normally connected for output, but is connected for input when necessary by a simple decode of the $\overline{\text{M}}$, $\overline{\text{W}}$ and $\overline{\text{STRBD}}$ lines.

An address latch (74533) is clocked with $\overline{\text{STRBA}}$. The memory address is decoded from these outputs and forms the chip-select ($\overline{\text{CS}}$) inputs to the RAM.

A shift register (74164) provides a time delay for RDYD for slow memories. An alternative for this would be a one-shot (9602). Fast memories do not require RDYD delayed.

6

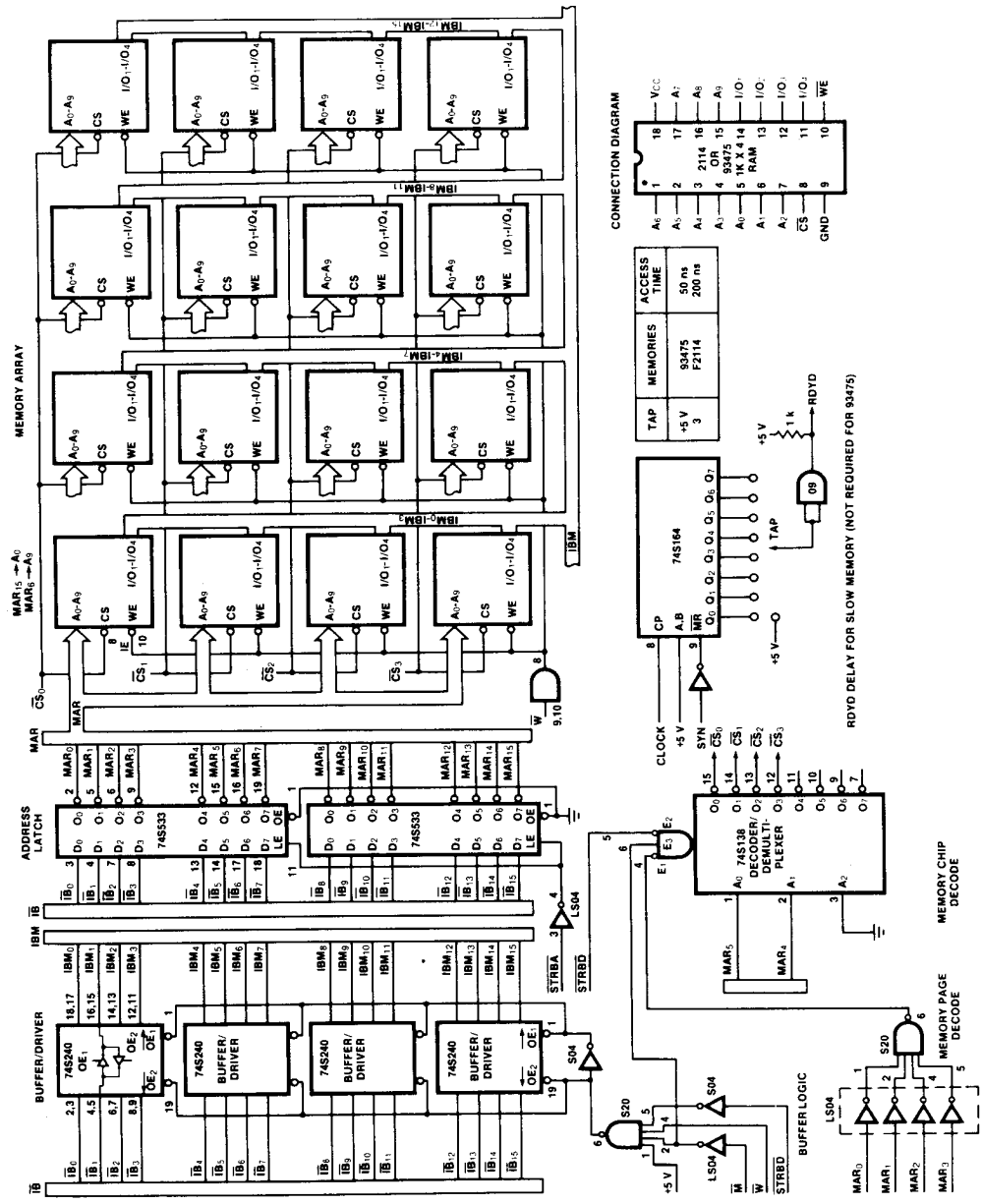


Fig. 7 Static Memory Connection Scheme

Input/Output

The F9445 I/O can utilize a simple scheme similar to the memory connection. To take full advantage of standard F9445 I/O instructions, however, I/O instructions must be externally decoded. An F9445 support circuit (F9448, F9447, F9470) can be used for this purpose.

To implement standard F9445 I/O without the support circuits mentioned above requires external logic. This can be implemented with an FPLA (93459). *Table 1* illustrates the PLA for I/O.

Table 1 I/O PLA Listing

```
                              *A LLLLLLLL
*P 00 *I --L-LLLHLHL--HLL *F -----AAA
*P 01 *I --L-HLLHLHL--LHL *F ------AA
*P 02 *I --L-LLLLHHL--HLL *F -----A-A
*P 03 *I -HH------L------ *F -----A-A
*P 04 *I --L-LLLLHHL--LHL *F -------A
*P 05 *I -LH------L------ *F -------A
*P 06 *I --L-LLLLHHL--LLH *F -----AA-
*P 07 *I --L-LLLLHHL--LHH *F ------A-
*P 08 *I --L-LLLLLHL--LHL *F -----A--
*P 09 *I ---------H---HHH *F A-------
*P 10 *I --L-LLLHLHL--HHH *F --AAA---
*P 11 *I --L-HLLHLHL--HHH *F ---AA---
*P 12 *I --L-LLLLHHL--HHH *F --A-A---
*P 13 *I --L-LLLLHHL--HHL *F --A-A---
*P 14 *I --LHLLLHLHLLH--- *F -AAAA---
*P 15 *I --LHHLLHLHLLH--- *F -A-AA---
*P 16 *I --LHLLLLHHLLH--- *F -AA-A---
*P 17 *I --LHLLLLHHLHH--- *F -A--A---
*P 18 *I --LHLLLHLHLHL--- *F -AAA----
*P 19 *I --LHHLLHLHLHL--- *F -A-A----
*P 20 *I --LHLLLLHHLHL--- *F -AA-----
*P 21 *I --L-HHHHHHH--HLH *F --AA----
*P 22 *I --L-HHHHHHH--HHL *F ---A----
*P 23 *I --L-HHHHHHH--LLH *F --A-----
*P 24 *I --L-HHHHHHH----- *F A-------
```

Key for Table 1:

- *A = Active level of outputs
- *P = Product term number
- *I = Inputs
- *F = Outputs
- – = Don't care
- H = High level
- L = Low level
- A = Active

The schematic (see *Figure 8*) shows a UART connection. The FPLA decodes the instructions and produces outputs from three multiplexers (74138). Spare outputs on these multiplexers can be used to drive other I/O devices.

Busy, done, mask and interrupt latches for both input and output are implemented. The baud rate generator (4702) is programmable for baud rates from 110 to 9600 baud.

In this scheme, the I/O bus is buffered (74240); this is optional. A one-shot (9602) provides a processor cycle delay by holding RDYD low. This allows the use of a slow UART (TR1263B). Converters (1488, 1489) are used for RS232-level connection, and current loop drivers are switch selected as shown. A one-shot (9602) provides a pulse for a TTY reader delay.

Dynamic Memory Control

Since dynamic memory is more difficult than static memory to connect to any processor, the F9445 requires some additional circuitry to drive dynamic memories (see *Figure 9*). There are several approaches to dynamic memory control:

6

Using an LSI special-purpose dynamic memory controller (e.g. F9446), which is by far the simplest solution;

Using standard SSI or MSI for the controller, requiring considerable board area;

Using software-assisted techniques, which reduces hardware requirements but can result in poorer overall performance; or

Using a standard MSI dynamic memory controller (e.g. 9642) with additional timing and control logic.

The last alternative has the advantage of using standard parts with a low part count and no software overhead. This is the scheme shown in *Figure 9*. The memory address register, data buffer and address decoder are required for any memory, static or dynamic. The 9642 multiplexes the 14-bit address for the dynamic memories, seven bits at a time. The memories require two strobes: a row address strobe ($\overline{RAS}$) and a column address strobe ($\overline{CAS}$). In the scheme shown, all memory chips receive the same $\overline{CAS}$ strobe, but the $\overline{RAS}$ strobe depends on the address. The strobes are sequenced using a combination of F9445 timing signals ($\overline{STRBA}$, SYN, $\overline{STRBD}$) and other signals generated by a 74164 shift register.

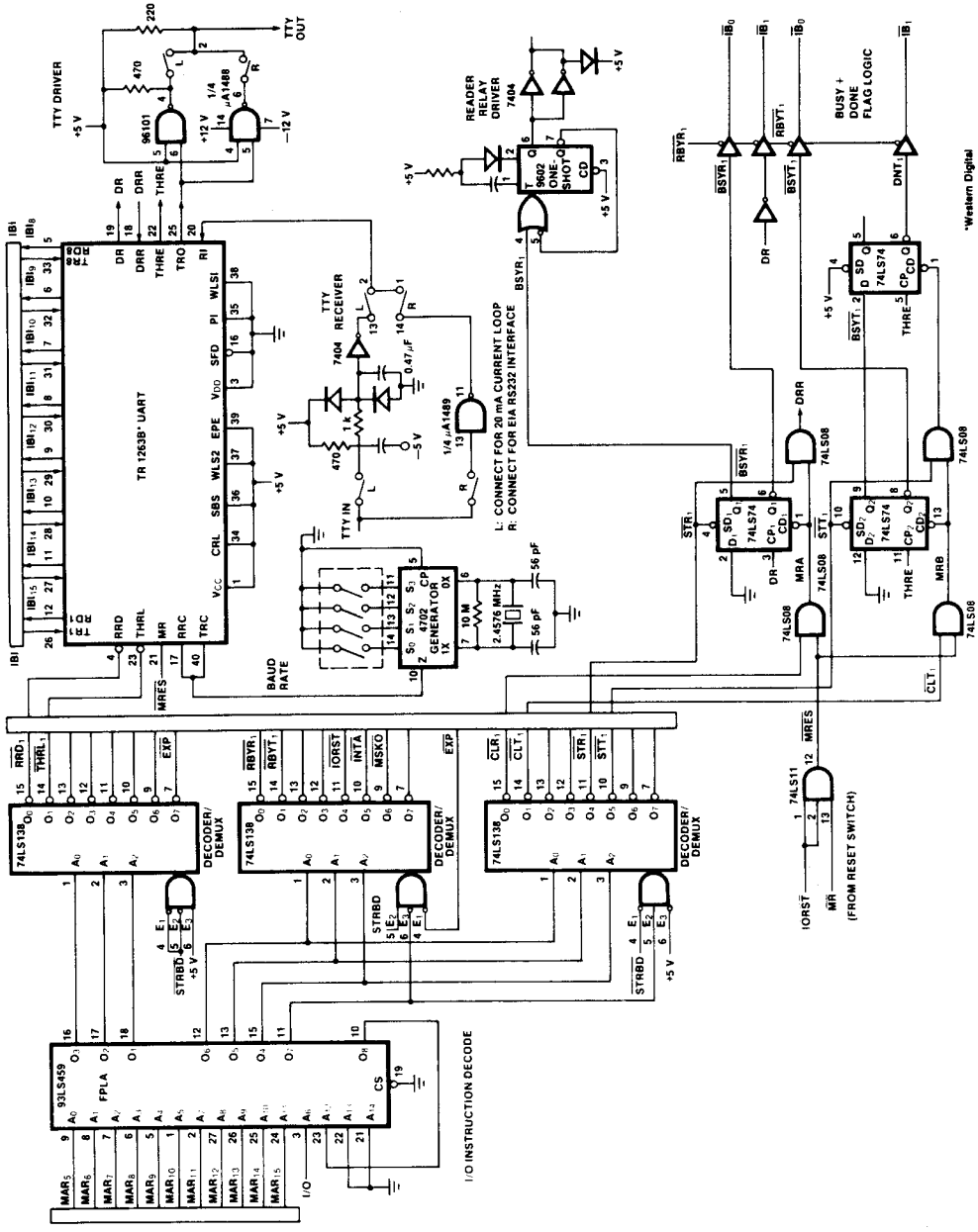


Fig. 8 F9445 Input/Output Connection Scheme (1 of 2)

Fig. 8 F9445 Input/Output Connection Scheme (2 of 2)

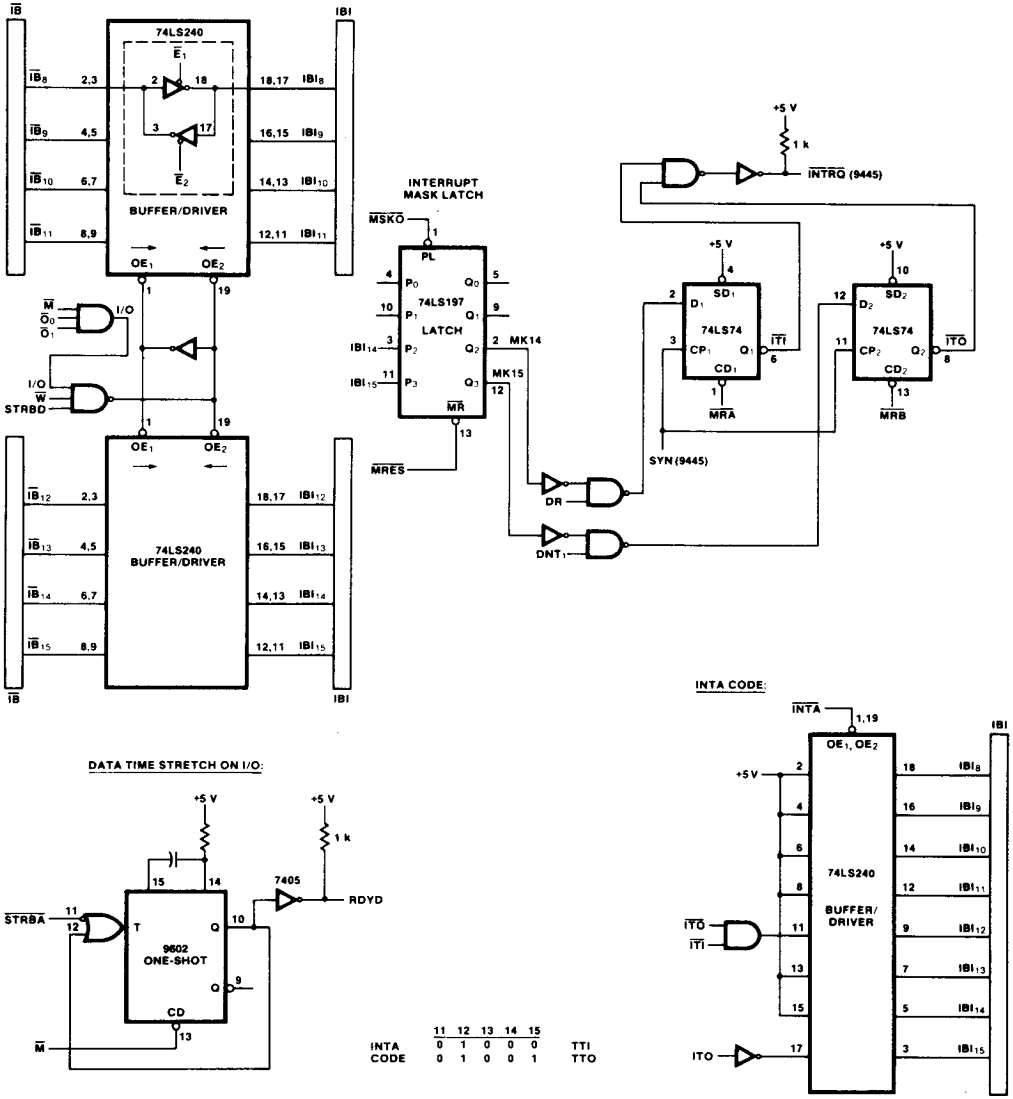
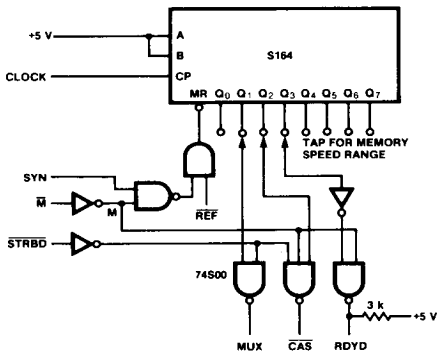
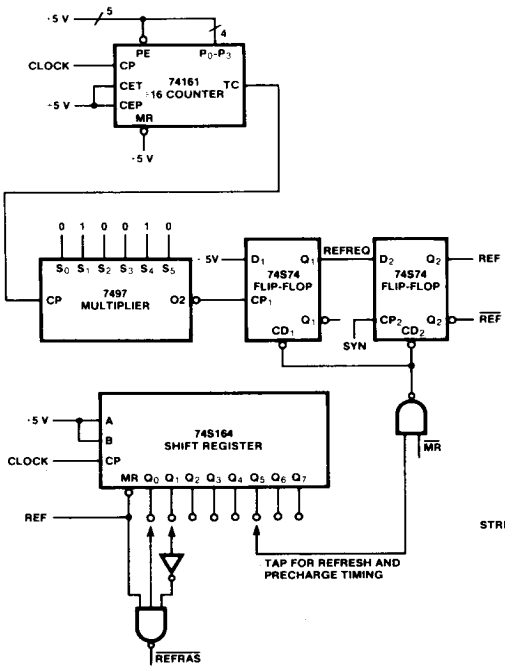


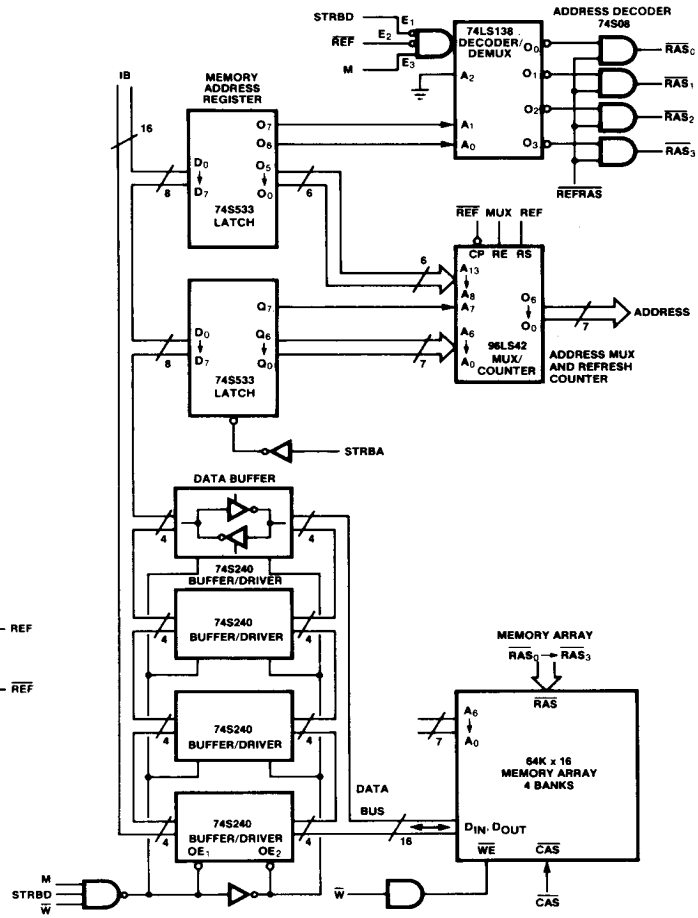
Fig. 9 F9445 Dynamic Memory Connection Scheme

Dynamic Memory Timing



Refresh Timing





Notes

1. D_{IN} connected to D_{OUT} connected to Data Bus.
2. All F16K devices have the same address lines and $\overline{CAS}$, $\overline{WE}$ line.
3. Each bank of 16 F16K has a separate $\overline{RAS}$ line (4 banks).
4. Each slice of 4 F16K is connected to a separate data bus line (16 slices).

The memory requires "refreshing" every 2 ms. The 9642 contains a 7-bit refresh counter. Every 15.63 ms (2/128), the memory controller enters "refresh" mode. This is synchronized with SYN to avoid any conflict. Another 74164 shift register controls the refresh timing which requires only an $\overline{RAS}$ strobe. After the refresh cycle, the refresh counter is incremented and the normal memory timing is resumed.

The refresh cycle takes place when needed and may take place during non-memory processor cycles. In these cases, the processor is not halted, and the refresh cycle is overlapped.

Different memory types have different speed requirements. These requirements can be met by changing the "taps" on the 74164 shift registers.

Console Control

On an application board, a minimal console is usually required. The APL (automatic program load) function can be easily implemented by pulsing the Console Request line LOW. There are no critical timing requirements since this signal is latched internally. The F9445 will continue to execute APL commands until the Console Request is raised. Since the bus must be HIGH for the APL to execute correctly, bits 5 and 6 of the bus may be tied to +5 V through 3 kΩ resistors as pullups.

For debugging and evaluation purposes, a console is a very useful tool. It gives complete control of the processor independent of software and memory operation.

Since the console commands are microprogrammed into the F9445, a full console design is fairly simple, the simplest full console uses the F9470 console-controller circuit, which drives an RS232 terminal and contains two serial I/O ports and a timer. The F9447 I/O controller can also be used to provide some console functions.

Interfacing to standard switches and lamps requires switch debouncing and encoding operations. The circuit shown in *Figure 10* uses R-S latches for switch debouncing and an FPLA (93409) for switch encoding.

An address latch is strobed on every $\overline{STRBA}$, and a data latch is strobed on every $\overline{STRBD}$ except "console code read." This results in the correct display on the lamps. The data switches are enabled with "console data read."

A Single-Step function is included. This function requires two additional latches, plus some decode logic, and implements a Continue followed by a Halt.

The Console Request is set whenever any operation switch is pressed and is reset when the console code is read from the FPLA. The circuit provides for control of two processors sharing the same bus.

All the switches are momentary-action type except the data switches and the select-processor switch.

A full listing of the FPLA is shown in *Table 2*.

The console provides all F9445 console functions, including Self-Test, plus the additional function of Single-Step, and is compact enough to be implemented with all switches, lamps, logic and connectors on a double-sided 17½-by-5½-inch printed-circuit board.

Table 2 Console PLA Listing

```
                              *A LLLLLLLL
*P 00 *I -H-HHHHHHHHHL--- *F -----A--
*P 01 *I -H-HHHHHHHHL---- *F --A--A--
*P 02 *I -H-HHHHHHHL----- *F --AA-A--
*P 03 *I -H-HHHHHHL---H-- *F ----AA--
*P 04 *I -H-HHHHHHL---L-- *F ----AA-A
*P 05 *I -H-HHHHHL------- *F --A-AA--
*P 06 *I -H-HHHHL-----H-- *F ---AAA--
*P 07 *I HL-H---L-----H-- *F ---AAA--
*P 08 *I HL-H---L-----L-- *F ---AAA-A
*P 09 *I HL-H-----L---L-- *F ----AA-A
*P 10 *I HL-H-----L---H-- *F ----AA--
*P 11 *I -H-HHHHL-----L-- *F ---AAA-A
*P 12 *I -H-HHHL--------- *F --AAAA--
*P 13 *I -H-HHL---------- *F --------
*P 14 *I -H-HL----------- *F ------A-
*P 15 *I ---L------------ *F -----AA-
*P 16 *I HL-H------------ *F ------A-
*P 17 *I LL-H-----H------ *F -----AA-
*P 18 *I -H-HHHHHHHHHH--- *F -----AA-
```

Key for Table 2:

- *A = Active level of outputs
- *P = Product term number
- *I = Inputs
- *F = Outputs
- – = Don't care
- H = High level
- L = Low level
- A = Active

6

Fig. 10 F9445 Console Connection Scheme (1 of 3)

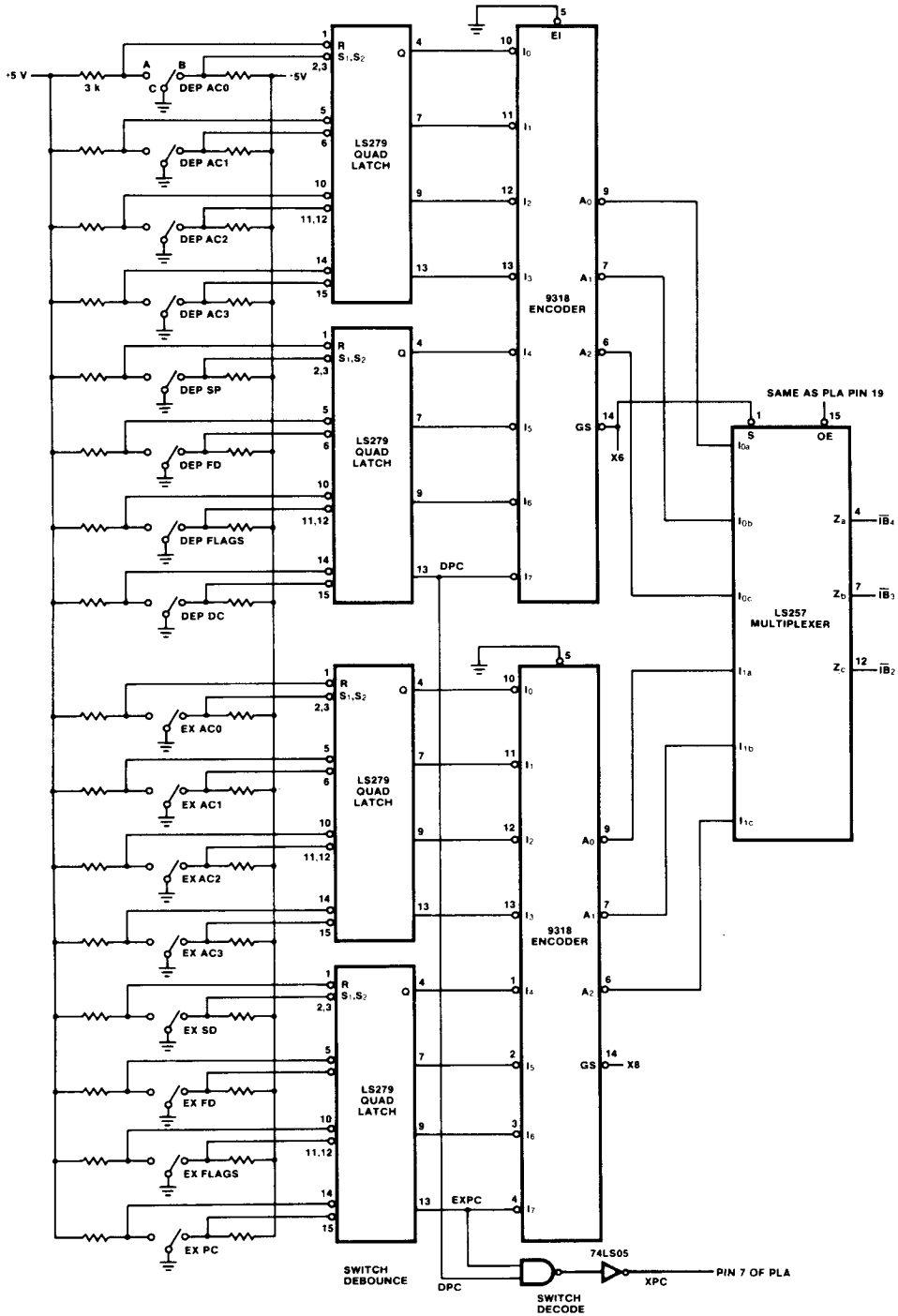


Fig. 10 F9445 Console Connection Scheme (2 of 3)

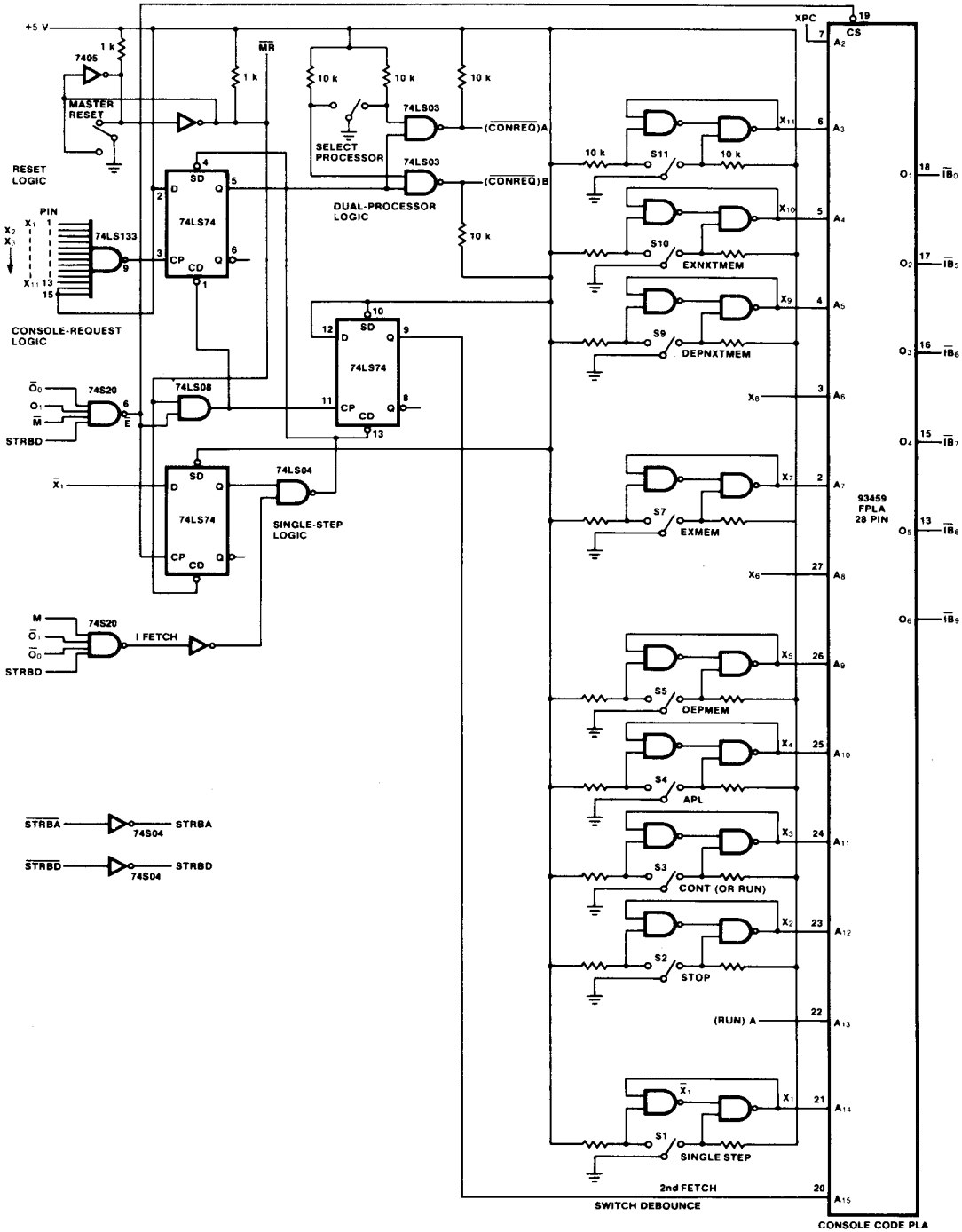
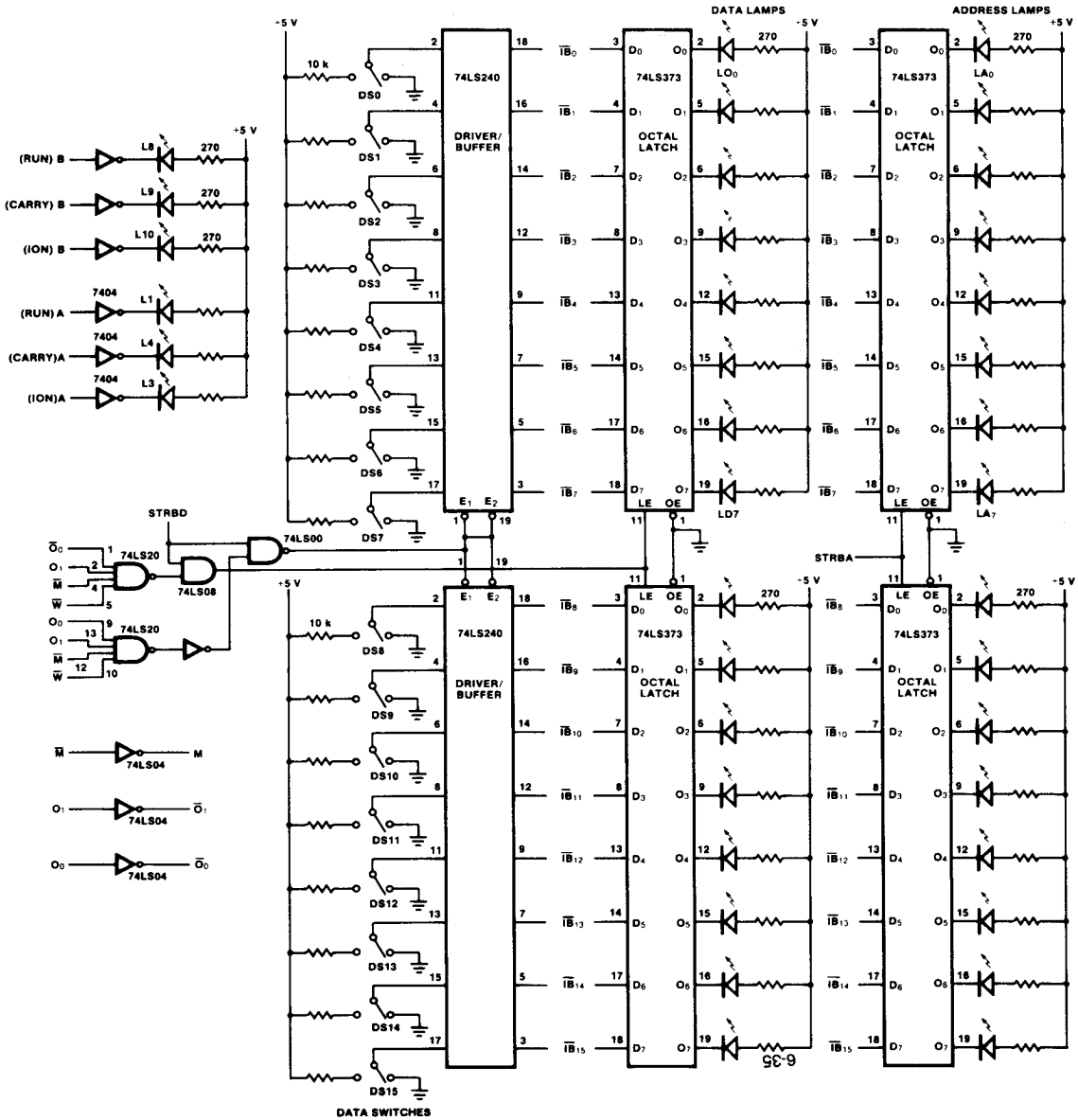


Fig. 10 F9445 Console Connection Scheme (3 of 3)



A Multiprocessor Scheme
There are many ways to envision two or more processors working concurrently. The method of interconnecting the processors depends on the application and the performance objectives. Listed here are a few of the options.

Independent Operation
For those processors which can be made to run independent tasks, this provides the most efficient scheme. Each processor has independent memory and resources.

Shared I/O
Each processor has its own memory but shares an I/O bus. This allows high-speed operation while minimizing system resource requirements.

Local and Common Memory
This gives a good compromise between performance and resource requirements. Each processor normally runs from its own memory at high speed. Accesses to a common memory are rarer and, because of the arbitration problems, slower.

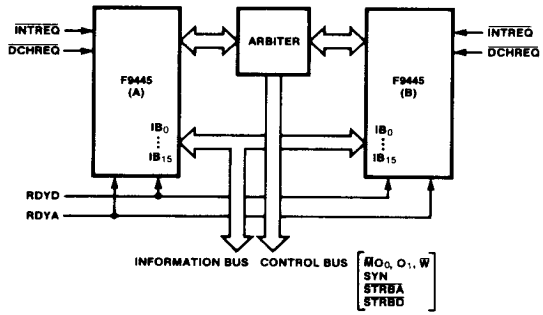
Tightly Coupled
Two or more processors share the same memory and I/O. This scheme is easiest to implement but, because of the completely shared resources, does not give as high performance as the Local and Common Memory scheme. However, for certain applications and for two processors only, this scheme can give a considerable performance increase over a single-processor system with very little hardware overhead. This scheme is described in the following paragraphs.

A general scheme for two tightly coupled F9445 processors is shown in Figure 11. The processors share a common bus and an arbiter selects which processor uses the bus and multiplexes the control lines accordingly. The I/O arbitration scheme is very simple: each processor assigns the bus to the other processor when it commences any cycle that does not use the bus, as long as $\overline{\text{BUSLOCK}}$ is not set.

The scheme is most efficient when the instruction mix includes many "long" instructions, such as Multiply, Divide, Parametric Shift and Normalize. Since only one processor is using the bus at any time, the synchronization signals RDYA and RDYD can be the same for both processors. However, the interrupt request ($\overline{\text{INTREQ}}$) and data-channel request ($\overline{\text{DCHREQ}}$) lines should be separate to avoid any conflicts in I/O handling.

6

Fig. 11 A Possible General Multiprocessor Scheme



F9445 Instruction Execution Times

Instruction	Clock Cycles	Execution Times			Notes
		16 MHz	20 MHz	24 MHz	
COM	6	0.375	0.3	0.25	
NEG	6	0.375	0.3	0.25	
MOV	6	0.375	0.3	0.25	
INC	6	0.375	0.3	0.25	Times for no-skip or unfulfilled skip;
ADC	6	0.375	0.3	0.25	for fulfilled skip; add 0.3 (0.25 at
SUB	6	0.375	0.3	0.25	24 MHz)
ADD	6	0.375	0.3	0.25	
AND	6	0.375	0.3	0.25	
OR	6	0.375	0.3	0.25	
MUL	70	4.375	3.5	2.9	
MULS	70	4.375	3.5	2.9	
DIV (Normal)	86	5.375	4.3	3.6	
DIV (Overflow)	14	0.875	0.7	0.58	
DIVS (Normal)	114	7.125	5.7	4.7	
DIVS (Overflow)	26	1.625	1.3	1.1	
NORM	10 + 4n	0.625 + 0.25n	0.5 + 0.2n	0.42 + 0.17n	n = number of steps needed for normalization. Time is 0.7 (0.59) if n=0.
SLLD	10 + 4n	0.625 + 0.25n	0.5 + 0.2n	0.42 + 0.17n	n = number of shifts;
SALD	10 + 4n	0.625 + 0.25n	0.5 + 0.2n	0.42 + 0.17n	time is 0.7 (0.59) if n=0.
SARD	10 + 4n	0.625 + 0.25n	0.5 + 0.2n	0.42 + 0.17n	
SLRD	10 + 4n	0.625 + 0.25n	0.5 + 0.2n	0.42 + 0.17n	
SKNV	14	0.875	0.7	0.58	
JMP	6	0.375	0.3	0.25	Times for page-zero addressing; add
JSR	6	0.375	0.3	0.25	0.3 (0.25) for indirect; add 0.3 (0.25)
ISZ	22	1.375	1.1	0.92	for auto-increment/decrement; add
DSZ	22	1.375	1.1	0.92	0.2 (0.18) for indexed.
LDA	12	0.75	0.6	0.5	
STA	12	0.75	0.6	0.5	
LDB	24	1.5	1.2	1.0	
STB	26	1.625	1.3	1.1	
PSHA	16	1.0	0.8	0.67	
POPA	16	1.0	0.8	0.67	
PSHF	16	1.0	0.8	0.67	
POPF	16	1.0	0.8	0.67	
POPJ	16	1.0	0.8	0.67	
PSHR	16	1.0	0.8	0.67	
TOPR	16	1.0	0.8	0.67	
TOPW	16	1.0	0.8	0.67	
MTSP	6	0.375	0.3	0.25	
MTFP	6	0.375	0.3	0.25	
MFSP	6	0.375	0.3	0.25	
MFFP	6	0.375	0.3	0.25	
SAV	60	3.75	3.0	2.5	
RET	80	5.0	4.0	3.3	
DSP	6	0.375	0.3	0.25	
NIO	12	0.625	0.6	0.5	
SKP	16	1.0	0.8	0.67	
DIA/B/C	12	1.0	0.6	0.5	
DOA/B/C	12	1.0	0.6	0.5	
ETRP	10	0.625	0.5	0.42	
DTRP	10	0.625	0.5	0.42	
E64K	10	0.625	0.5	0.42	
D64K	10	0.625	0.5	0.42	

Note: Execution times are given for 20 MHz and 24 MHz clock. The clock may be operated from > 0 to 24 MHz within the specified temperature and voltage range.

Fig. 12 ALU Cycle Timing*

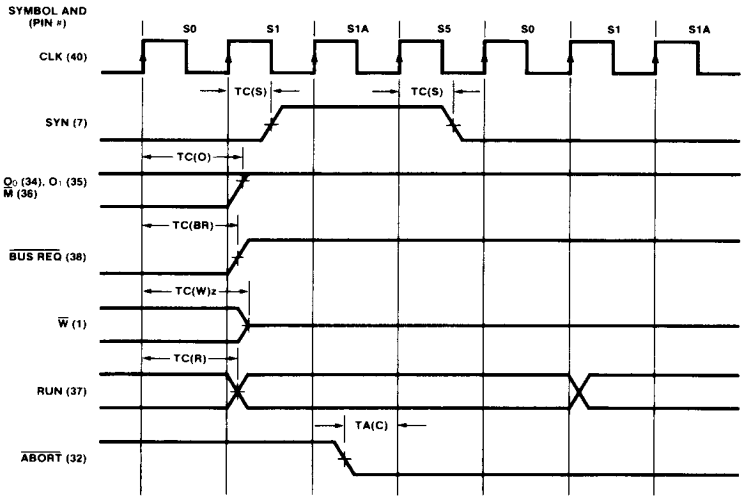
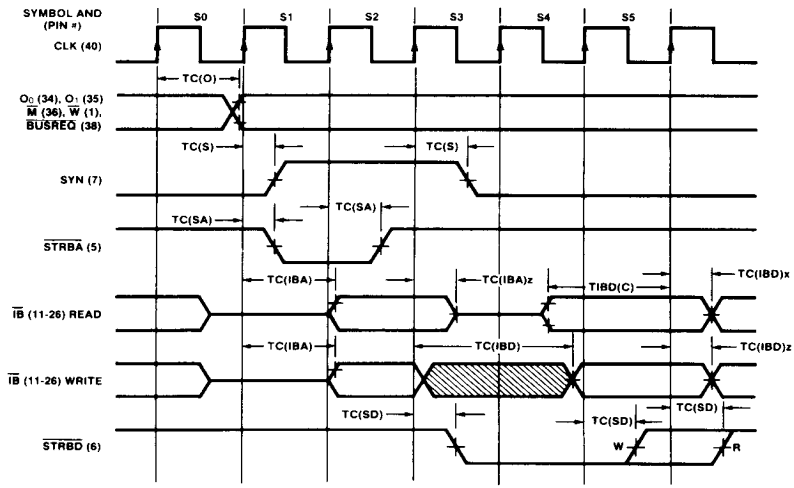
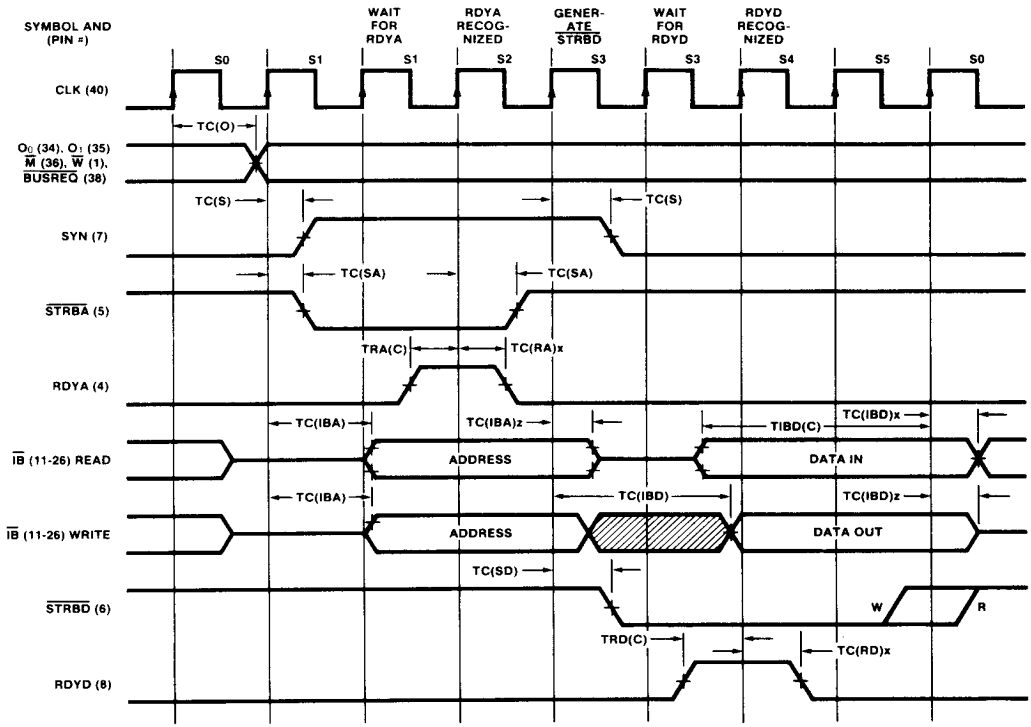


Fig. 13 Minimum Memory Cycle Timing*



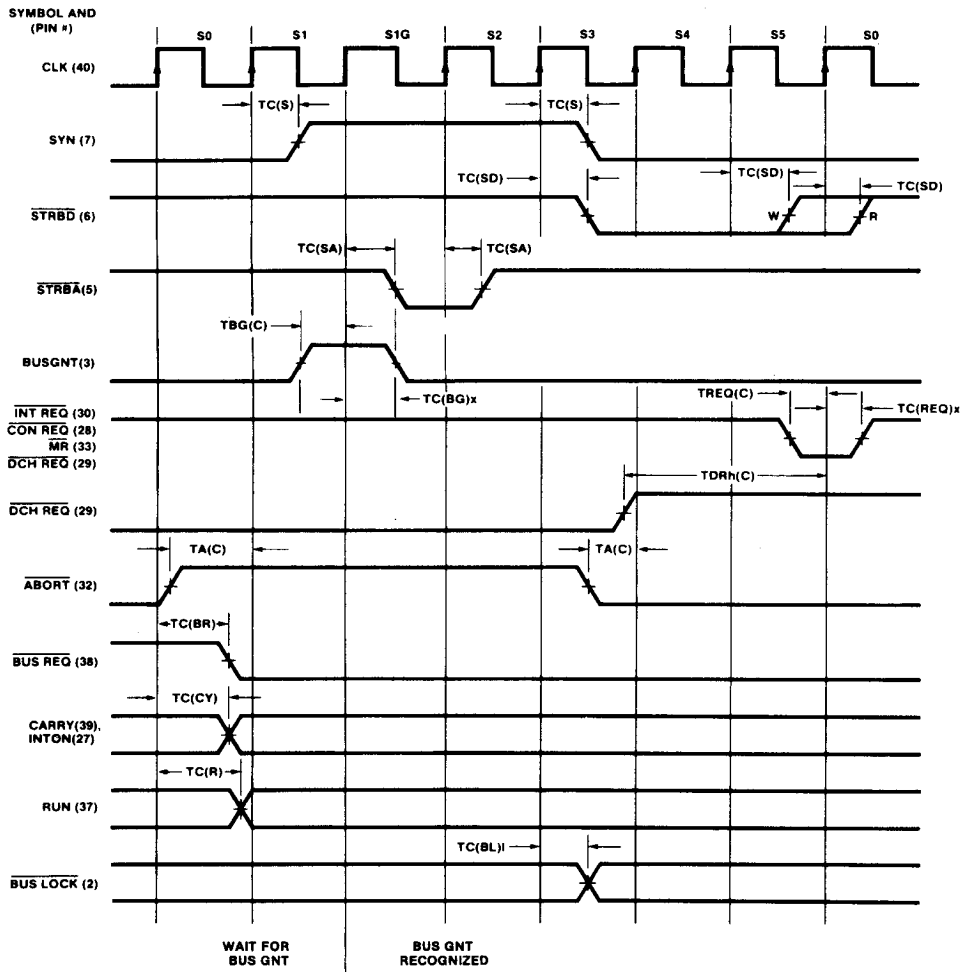
*See Timing Parameter Symbol Conventions at end of data sheet.

Fig. 14 Extended Memory Cycle Timing*



*See *Timing Parameter Symbol Conventions* at end of data sheet.

Figure 15. Bus and Status Control Timing*



*See *Timing Parameter Symbol Conventions* at end of data sheet.

**If this DCH REQ set-up time is missed, it is not recognized for another complete cycle.

Guaranteed Operating Ranges

Part Number	Supply Voltage (V_{CC})			Case Temperature
	Min	Typ	Max	
F9445DC	4.75 V	5.0 V	5.25 V	0 to +75°C
F9445DM	4.5 V	5.0 V	5.5 V	−55 to +125°C

DC Characteristics

(Over guaranteed operating ranges unless other wise noted.)
I_{INJ}(min) = 375 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; $\overline{IB}_{(0\text{-}15)}$, $\overline{W}$	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	40	μA	V_{CC} = Max, V_{IN} = 2.7 V, I_{INJ} = MIN
I_{IH}	Input HIGH Current; $\overline{IB}_{(0\text{-}15)}$ (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 $\overline{IB}_{0\text{-}15}$, W			100	μA	V_{CC} = Max, V_{OUT} = 2.4 V, I_{INJ} = MIN
I_{OZL}	Output OFF State (High Impedance) Current $\overline{IB}_{0\text{-}15}$		−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
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} = 300 MIN
V_{INJ}	Injector Voltage		1.3		V	I_{INJ} = 400 mA

*Not more than one output to be shorted at a time.

AC Characteristics

T_C = 0 to 75 °C; V_{CC} = 4.75 to 5.25 V; I_{INJ} = 375 mA; C_L = 15 pF.
Input conditioning: Rise Time = 6 ns; Fall time = 6 ns; Amplitude = 0 to 3 V

Refer to *Symbol Conventions* at the end of this data sheet for explanation of the timing parameter symbols.

Symbol	Characteristic	Min	Typ	Max	Unit
TC(O)	Propagation delay, CLK to O_0, O_1, $\overline{M}$		60		ns
TC(S)	Propagation delay, CLK to SYN		30		ns
TC(W)	Propagation delay, CLK to $\overline{W}$		70		ns
TC(W)z	Propagation delay, CLK to $\overline{W}$ going 3-state		70		ns
TC(IBA)	Propagation delay, CLK to $\overline{IB}_{(0\text{-}15)}$, address		60		ns
TC(IBA)z	Propagation delay, CLK to $\overline{IB}_{(0\text{-}15)}$, address, going 3-state (read cycle)		35		ns
TC(SA)	Propagation delay, CLK to $\overline{STRBA}$		30		ns
TC(IBD)	Propagation delay, CLK to $\overline{IB}_{(0\text{-}15)}$, data out		75		ns
TC(IBD)z	Propagation delay, CLK to $\overline{IB}_{(0\text{-}15)}$, data out, going three-state		35		ns
TC(SD)	Propagation delay, CLK to $\overline{STRBD}$		25		ns
TRA(C) TC(RA)x	Setup time, RDYA to CLK Hold time, CLK to RDYA		3 10		ns ns
TRD(C) TC(RD)x	Setup time, RDYD to CLK Hold time, CLK to RDYD		2 10		ns ns
TIBD(C)	Setup time, $\overline{IB}_{(0\text{-}15)}$, data in, to CLK (read or fetch cycle)		75		ns
TC(IBD)x	Hold time, $\overline{IB}_{(0\text{-}15)}$, data in, after CLK (read or fetch cycle)		25		ns
TREQ(C)	Setup time, $\overline{INTREQ}$, $\overline{DCHREQ}$, $\overline{CONREQ}$, $\overline{MR}$ to CLK, all are the same timing relative to S5		15		ns
TC(REQ)x	Hold time, $\overline{INTREQ}$, $\overline{DCHREQ}$, $\overline{CONREQ}$, $\overline{MR}$ after CLK, all are the same timing		20		ns
TDRh(C)	Data channel ($\overline{DCHREQ}$) off setup time from CLK (to finish data-channel cycle)		100		ns
TA(C)	Setup time, $\overline{ABORT}$ to CLK		30		ns
TC(BL)1	Propagation delay, CLK to $\overline{BUSLOCK}$ going LOW		35		ns
TBG(C) TC(BG)x	Setup time, BUSGNT to CLK Hold time, CLK after BUSGNT		10 10		ns ns
TC(R)	Propagation delay, CLK to RUN		80		ns
TC(CY)	Propagation delay, CLK to CARRY		50		ns
TC(INT)	Propagation delay, CLK to INTON		50		ns
TC(BR)	Propagation delay, CLK to $\overline{BUSREQ}$		40		ns

Timing Parameter Symbol Conventions

The abbreviated symbols used for ac characteristic timing parameters in this data sheet are defined as follows:

The timing symbol convention is: TAb(C)d

The timing symbols all begin with the letter "T".

The second position, represented by "A", indicates the signal node beginning the interval.

The position "b" defines the direction of signal transition at the beginning node "A", if such definition is necessary; the new state of the signal may be: l = Low; h = High; z = 3-state; x = Don't care; v = Valid

The position "C", which always appears within parentheses, indicates the signal node ending the interval.

The position "d" is the same as "b" but refers to the state of the signal at the node indicated by the mnemonic in position "C".

Ordering Information

ORDER CODE	SPEED	TEMPERATURE
F9445-24 DC	24 MHz	0°C to +75°C
F9445-24 DM	24 MHz	-55°C to +125°C
F9445-24 DMQB	24 MHz	-55°C to +125°C
F9445-20 DC	20 MHz	0°C to +75°C
F9445-20 DM	20 MHz	-55°C to +125°C
F9445-20 DMQB	20 MHz	-55°C to +125°C
F9445-16 DC	16 MHz	0°C to +75°C
F9445-16 DM	16 MHz	-55°C to +125°C
F9445-16 DMQB	16 MHz	-55°C to +125°C

For other temperature ranges, contact Fairchild Sales Office.

All packages are 40-pin ceramic DIPs

Connection Diagram
40-Pin DIP (Top View)