Marinchip Systems M9900 CPU Theory of Operation

by Fred Camerer and John Walker

Marinchip Systems 16 St. Jude Road
Mill Valley, CA 94941 ■ (415) 383-1545

M9900 CPU Theory of Operation

Table of contents

1. Introduction

Marinchip Systems has developed a CPU board which allows the Texas Instruments TMS9900 processor to be used on the enormously popular S-100 bus. This board, the M9900 CPU, replaces the 8080 or Z-80 CPU board in an S-100 mainframe, and uses the existing memory and peripherals, or almost any S-100 compatible board. The TMS9900 is an extremely powerful processor, which features hardware multiply and divide instructions, multiple sets of 16 general purpose registers, addressing modes including direct, indirect, indexed, and auto-increment, and a context switch mechanism that allows both rapid interrupt response and user extension of the hardware instruction set. The TMS9900 is a single chip processor that is comparable to the PDP-11 in most respects, and exceeds it in several important ways, notably the provision of 16 registers instead of 8 and the fact that the user can have as many independent register sets as he wants, and need not be limited to hardware registers within the CPU.

This manual explores the problems of interfacing a fully parallel 16 bit processor to the S-100 bus, concentrating on the peculiarities of the S-100 bus structure and the solutions that were developed to surmount them. The performance tradeoffs and compromises made in the design will be discussed, and finally some recommendations regarding standards for certain S-100 signals will be presented.

Since this manual discusses signals generated by both the 8080 and the TMS9900 processors, as well as S-100 bus signals, some conventions have been adopted to lessen confusion. Signals generated by either CPU chip are referred to by the name given them by the chip manufacturer. Which chip is being referred to should be clear from context. S-100 signals will always be suffixed by their pin number in parentheses. Active low signals will be identified by a minus sign following the signal name.

2. TMS-9900 Bus Structure

The task of interfacing one processor to a bus designed for another involves designing logic that will transform the signals generated by one processor into the corresponding signals of the other processor. The fact that the processors have different word lengths increases the complexity of the solution, but introduces few conceptual problems. First, let us look at the signals used by the TMS9900 processor. The TMS9900 is one of the simplest microprocessors to interface, largely because there is no multiplexing of information on package pins. Each pin has one clearly defined function, so the external decoders and latches required in many other systems may be eliminated. The price paid for this simplicity is the 64 pin package required to accomodate all the independent signals. The signals which are relevant to memory accesses are the address bus, data bus, and control bus. The address bus consists of 15 lines representing bits 1 through 15 of the address being referenced. Since the TMS9900 uses 16 bit wide memory, the 15 address bits permit addressing of 64K bytes of memory. (Byte addressing is handled by masking data within the processor, and is of no concern in interfacing it.) The data bus is a 16 bit bidirectional bus which behaves exactly like the data bus of an 8080, except that it is twice as wide. The control bus consists of the signals that control the external memory. These signals all appear on their own dedicated pins, so no status latching is required as for the 8080. The signal MEMEN- is a low-true signal which indicates a memory access is in progress. When it appears, a valid memory address is present on the address bus. The signal DBIN indicates the direction of the memory data transfer. When DBIN is high, the output drivers on the 16 data bus lines have been disabled to allow the memory to place data on the lines in response to a memory read. MEMEN- and DBIN simultaneously active indicate a memory read, while MEMEN- active and DBIN inactive indicate a memory write. The TMS9900 generates a write enable signal, WE-, which goes active during a write cycle after the address and data have been present for sufficient time to satisfy the setup time requirements of most static memories. This signal allows easy interfacing of memories to the processor, but is not used in the M9900 CPU because the memory cycle must be segmented into two independent 8 bit accesses. The READY and WAIT lines permit the processor to delay to accomodate slow memories. After the address has been presented to the memory bus and MEMEN- goes active, READY is sampled. If high, the memory cycle will complete on the next clock cycle. If READY is low, the WAIT signal is brought active and the processor will delay one cycle, at which time it will test READY again, and either proceed or delay again depending on its state. In the M9900 CPU the READY line is used to cause the processor to wait for the two cycles on the S-100 bus to be completed. The WAIT line is not used, since the M9900 must generate the S-100 wait signal in response to a wait state encountered on either (or both) of the 8 bit accesses. The above signals comprise the complete memory interface of the TMS9900, and seem to be the minimum orthogonal set of signals allowing control of variable speed memories with no external decoding. The TMS9900 has no complex timing relationships between signals. Every signal generated by the processor is related to one of the four clock phases, and there is a uniform propagation delay, typically 20 nanoseconds, from the clock edge to the signal.

3. 8080 Bus Structure

The memory bus of the 8080 is similar in concept to that of the 9900, but more complicated in practice because the control signals relevant to memory are divided into command and control signals which appear directly on processor pins and status signals which are output on the data bus during the first state of each machine cycle. A memory cycle in the 8080 consists of three or more machine states. During the first state, the 8 bit data bus outputs the status bits which indicate what type of cycle is being performed. The processor generates a signal, SYNC, which identifies the first state of the cycle and may be used to latch the status bits for use later in the cycle. The status bits, properly decoded, identify the type of cycle. The ones relevant to memory and I/O cycles are MEMR, WO-, INP, and OUT. The MEMR bit identifies a memory read cycle (except when it comes on during a halt cycle, which will be ignored henceforth). The INP and OUT bits identify I/O read and write cycles, respectively. The WO- signal is active when a write is being done to either memory or an I/O device: external logic must generate the write strobes for I/O and memory based upon the OUT bit. The rest of the status bits are used to implement the 8080 interrupt scheme, to indicate when the machine is halted, to flag accesses to the stack, and to indicate when an instruction byte is being fetched. The command and control signals are almost identical to those of the TMS9900 in name and function. The DBIN signal indicates that the data bus is in input mode to receive data, WR- is the delayed output data strobe, READY is the line that may be used to stretch the length of a memory or I/O cycle, and WAIT is the signal that confirms that the processor is waiting. The address bus on the 8080 is 16 bits wide, since the 8080 uses byte-addressable memory, and the data bus is an 8 bit bidirectional bus. I/O on the 8080 provides 256 ports. An IN or OUT instruction referencing one of these ports causes a bus cycle identical to a memory cycle, except that it is identified as I/O by the presence of the INP or OUT bits on the status bus. In an I/O cycle, the 8 bit port address is presented on BOTH the high and low 8 bits of the address bus. (This "address mirror" feature is intended to reduce bus loading by dividing devices between the high and low bytes of the address lines in system without address buffering.) The timing relationships of the 8080 signals are quite complicated. The 8080 is driven with a two phase clock. The two phases must not overlap, and have different widths. The command and control signals may be delayed as much as 120 nanoseconds from their controlling clock edge, and the address and data may not be stable for as much as 220 nanoseconds after the clock edge.

4. S-100 Bus Structure

The S-100 bus is largely the result of simply buffering the signals generated by the 8080 and directly presenting them to the bus. It will help to break up the bus signals into groups and discuss them independently.

4.1. Clocks

There are three clock signals on the S-100 bus. The first two are the Phase 1 and Phase 2 processor clocks, presented as TTL levels, PH1(24) and PH2(25). The third signal is named CLOCK-(49), and is an inverted Phase 2 clock, delayed about 60 nanoseconds (an artifact of the original Altair design). Both PH2(25) and CLOCK-(49) are commonly used as timing references by peripheral boards.

4.2. Address Lines

The 16 address lines appear on the bus, buffered directly from the processor address lines. The line ADDR DSBL-(22) will cause the address lines to go to high impedance when pulled low.

4.3. Data Lines

The 8 bit bidirectional processor data bus is split by logic on the CPU board into an 8 bit data in bus and an 8 bit data out bus on the S-100 bus. Whether the data in bus is applied to the processor data bus is controlled by the DBIN signal from the processor. Since the data out lines are dedicated to output, their drivers are always on, except if disabled by pulling the line DO DSBL-(23) low.

4.4. Command and Control Lines

The direct control lines from the processor appear on the command and control S-100 lines. The signal PSYNC(76) is the SYNC signal from the processor that identifies the first state of each machine cycle. PWR-(77) is the processor delayed write strobe. PDBIN(78) is the DBIN signal from the processor. Most (but not all) S-100 boards only drive the data in bus when PDBIN(78) is high. PINTE(28) is the signal from the processor (INTE) which indicates that the processor has interrupts enabled. PWAIT(27) is the WAIT signal that indicates the processor is waiting for the READY signal from a peripheral. PHLDA(26) is the HLDA processor signal which acknowledges a DMA request. (Refer to the section below on DMA control signals for more information on the PHLDA(26) signal.) The six command and control lines may be forced to high impedance by pulling the line CC DSBL-(19) low.

4.5. Status Lines

When the status signals appear on the processor data bus during the first state of a cycle, they are latched and presented on a separate set of status lines on the S-100 bus. The status bus remains stable for the duration of the machine cycle. The SM1(44) line indicates the current cycle is fetching the first byte of an instruction. The SOUT(45) bit identifies an output cycle (initiated by the OUT instruction). The SINP(46) pin identifies an input (IN instruction) cycle. SMEMR(47) indicates a memory read cycle, and SHLTA(48) indicates that the processor is halted. The SWO-(97) signal indicates a write-type operation: if SOUT is low, it is a memory write. If high, it is an I/O write. The SINTA(96) signal identifies the special 8080 cycle that responds to an interrupt by requesting an instruction to be executed by the processor, and the SSTACK(98) signal flags the current memory cycle as referring to the stack. The status bus signals may be forced to high impedance by pulling the STAT DSBL-(18) line low.

4.6. Memory Control Lines

These lines are a collection of signals relating to memory and I/O cycles. The PRDY(72) line is applied through a buffer to the READY pin on the 8080, and is used by slow memory and peripherals to stretch a cycle that references them. The MWRITE(68) signal is a copy of the PWR-(77) write strobe that only goes low on memory writes (e.g., when SOUT(45) is low). This is the most commonly used write control signal in memory boards. It is also quite confusing in that many S-100 machines do not generate it on the CPU board. The Altair and IMSAI, for example, generate this signal on the front panel board, with the result that memories which require this signal will not run if the front panel is removed. To compensate for this, many of the "reset and go" PROM/RAM boards contain logic to generate this signal, since they are replacing the front panel in a system. This signal is logically a CPU signal, and really has no business being generated anywhere else, but the weight of history is great, so for the time being confusion will reign. The UNPROT(20), PROT(70), and PS-(69) signals respectively clear, set, and contain the status of the memory protect flip-flop on the currently addressed memory board. Since memory protect is little used, they will not be discussed further. The PHANTOM-(67) signal is used by many systems and boards to allow an automatic power-on start from a nonzero address. The memory board at address zero is strapped to be deselected regardless of the address on the bus when PHANTOM-(67) is low. A PROM board then can, when triggered by a reset signal, disable the memory at address zero, enable itself, and cause the CPU to execute instructions from the PROM even though the CPU "thinks" it is executing from address zero. The PHANTOM-(67) line is normally turned off once the processor has jumped into the PROM itself. The memory at address zero may then be referenced normally.

4.7. Interrupt Signals

The lines VI0-(4) through VI7-(11) are the vectored interrupt requests. Pulling one of these lines low will cause an interrupt at the corresponding level, assuming that interrupts are enabled and the processor is not servicing a higher priority interrupt. VI0-(4) causes an interrupt which is not distinguishable from a reset of the processor, and hence should be used with discretion. Since the early S-100 processors did not provide vectored interrupts on the CPU board, the signal PINT-(73) was defined to permit a separate vectored interrupt controller board to request an interrupt from the processor. In such a system, the vectored interrupt controller board received the 8 vectored interrupt lines, chose the highest priority, then requested an interrupt using the PINT-(73) line and supplied the processor with an interrupt instruction when the SINTA(96) line went high. Because of the way the 8080 CPU interrupt structure worked, systems without a vectored interrupt board could provide a single level interrupt simply by pulling the PINT-(73) line low.

4.8. Front Panel Signals

The following signals provide the interface between the front panel and the boards on the bus. Since some systems have complex front panels and others have only a reset switch, most of these signals may be absent in a particular system. The only signals in the front panel group present in all S-100 mainframes are PRESET-(75) and POC-(99). PRESET-(75) is pulled low when the reset switch on the front panel is activated. The CPU board normally responds to PRESET-(75) by resetting the processor and pulling POC-(99) low. POC-(99) is also driven by logic that keeps it low on a power-up operation until the supply voltages have stabilized. Thus, if peripheral boards use POC-(99) as their clear signal, they will be properly reset both by initial application of power to the system and by the reset switch. The following signals are present only in those systems with elaborate front panels. The RUN(71) signal is used to indicate that the front panel RUN/STOP switch is in the RUN position. The SSW DSBL-(53) signal is used by some front panels to implement data input to the processor from the front panel switches. The EXT CLR-(54) signal resets I/O devices independent of the processor. What it does and when it is activated depends upon the peripheral and mainframe being examined. The SS(21) signal is used by certain front panels to implement a single step function. It allows the front panel to place data directly on the processor data bus (this requires a connection from the front panel to the CPU card). The XRDY(3) signal allows the front panel to stop the processor without interfering with the PRDY(72) signal used by memory and peripherals. The typical CPU ANDs XRDY and PRDY to develop the READY signal for the 8080.

4.9. DMA Control Signals

Direct Memory Access (DMA) can be implemented on the S-100 bus through use of the PHOLD-(74) signal. When this signal is pulled low, the processor suspends execution at the end of the current instruction, and raises the HLDA line, which appears on the bus as PHLDA(26). Once the requesting peripheral sees this signal, it can pull the lines STAT DSBL-(18), CC DSBL-(19), ADDR DSBL-(22), and DO DSBL-(23) low and assume control of the bus itself. It should be noted that pulling the CC DSBL-(19) line low forces the PHLDA(26) signal itself to high impedance. If this signal is being used as a direct enable by the DMA board, its floating may lead to havoc, especially because line 25 right next door is the Phase 1 2MHZ clock, which puts a lot of noise on the floating line. Pulling up PHLDA(26) to +5 with a resistor neatly solves this problem, but few CPU boards do this.

4.10. Power and Ground Lines

Power to the S-100 bus is unregulated, since each board regulates the raw supply to its desired logic levels. Pins 1 and 51 supply a nominal +8 volts. Pin 2 supplies +16 volts, and pin 52 supplies -16 volts. There are no specifications and few conventions about the voltages actually found on these pins. Specifically, some very high voltages occur in certain mainframes. If the +8 line goes to 12 volts, the dissipation in the on board regulators may exceed the capacity of their heat sinks, forcing thermal shutdown of the regulator and malfunction of the board. One might point out as an aside that since these supplies are unregulated they contain ripple, and at least one CPU board counts on this! The ripple is amplified with an op-amp and used to provide a line frequency real time clock. Pins 50 and 100 are the common ground return for all supplies, and the logic ground reference. Whether or not this is tied to safety (green wire) ground depends upon the mainframe. Pin 55 is defined as chassis ground, but very few mainframes actually connect this pin.

The reader might have noticed that it took a lot more words, explanations, and hedging to explain the S-100 bus than to explain the 8080 processor signals themselves. This is the case because one of the many different CPU boards or "compatible" I/O boards gives lie to just about any definitive statement regarding the bus one might choose to make (except possibly that it has 100 pins).

5. Interfacing to the S-100 Bus

The job of interfacing the TMS9900 to the S-100 bus fell largely into three phases: mapping the signals from the 9900 bus to the S-100 bus, defining the subsystems that would make up the CPU card, and logic design and debug. First of all, the memory cycles of the TMS9900 and the 8080 were drawn out side by side, and a mapping was defined between the two. It was found that every signal corresponded in both directions closely enough that once 16 bit TMS9900 bus was multiplexed onto the 8 bit S-100 bus, all the other signals could be translated. In this phase of the design, it was decided to implement S-100 I/O by providing memory mapped I/O in a page of the TMS9900 addressing space. This was done because the unique TMS9900 I/O scheme did not lend itself to adaptation to existing S-100 peripherals, and because the memory bus interface that had to be designed for the TMS9900 already contained all the logic required to implement the I/O cycle. Once it became clear that the problem was solvable, the following subsystems that made up the TMS9900 CPU board were defined.

5.1. Clock Generator

This subsystem contains the generator for the four phase clock required by the TMS9900 and the logic that transforms the 9900 clock pulses into a synchronized replica of an 8080 clock. A properly delayed inverted signal is generated for the CLOCK-(49) signal.

5.2. Address Bus Driver

This subsystem takes the address bus of the 9900 and drives the address lines on the S-100 bus. Depending upon whether the address on the address bus is within the memory mapped I/O region, this logic either drives the address directly onto the bus (for normal memory accesses) or places the low 8 bits of the 9900 address bus on both the high and low 8 bits of the S-100 address lines (for I/O accesses). For normal memory references, the upper 15 address bits on the bus are supplied by the 9900 chip, and the low order bit is generated by the memory bus controller subsystem (see below).

5.3. Data Bus Driver

This subsystem contains the drivers that route data to and from the data in and data out lines. The data out lines are driven by two sets of drivers that can route either the high or the low byte of the 9900 16 bit data bus to the data out lines. The data in lines are received by two 8 bit latches that save the data appearing on the data in lines for parallel presentation to the 9900 as a 16 bit data word. In addition, extra drivers provide the data paths necessary to use the data in and data out lines as a 16 bit parallel bidirectional data bus for use with special Marinchip high-performance memories (see the description of "Sixteen bit mode" below).

5.4. Memory Bus Controller

This subsystem is the heart of the M9900 CPU. When a memory access is initiated by the TMS9900, the memory bus controller is started. Depending upon whether the address presented by the 9900 is within the memory mapped I/O region or not, the controller generates an I/O or two 8 bit memory cycles. While the controller is operating, the READY line of the 9900 is held low to force it to wait for the data to be transferred on the S-100 bus. The memory bus controller is a random logic implementation of the 8080 memory timing logic, which generates PSYNC(76), PWR-(77), MWRITE(68), PDBIN(78), and PWAIT(27) as if an 8080 were directly connected to the bus. PRDY(72) is honored with the same timing as the 8080 expects. The memory bus controller also generates the signals to the data bus drive subsystem that route data to and from the S-100 bus and the 9900 data bus, and the strobes that latch data from the S-100 bus at the same time an 8080 would have sampled it. When the memory bus controller gets the PRDY(72) for the final byte of a transfer, it raises READY to the 9900, allowing it to proceed with its execution. By anticipating ready it is possible to complete the 16 bit memory cycle in the same time an 8080 would have taken to fetch two bytes (six machine states). I/O cycles take three machine states. The PINTE(28) signal, which indicates that interrupts are enabled in the 8080, is always high, since the 9900 does not provide this status on an external pin. The PHLDA(26) signal, which acknowledges a DMA request is simply the HOLDA pin of the 9900 buffered to the bus. A pull-up resistor is provided on this line to prevent the "floating PHLDA(26)" problem mentioned above. The DMA request line, PHOLD(74), is latched before being passed to the HOLD- line on the 9900. While this signal is internally latched in the 9900, the internal latching is done at a different time in the 9900 and the 8080, so the external latch accomodates any device that counts on the 8080's timing. This may be paranoid, but better to be over compatible than to discover the CPU doesn't work with somebody's "sloppy disc" controller.

5.5. Status Bus Generator

Logic decodes signals from the 9900 and the memory bus controller to generate the status bus signals. SWO-(97), SHLTA(48), SOUT(45), SINP(46), and SMEMR(47) are generated compatibly with the 8080. Since SSTACK(98) represents a concept not applicable to the 9900, it is always low. The SINTA(96) signal, also not applicable to the 9900, is used to identify a cycle as I/O as opposed to memory. This signal is used to simplify the design of special 16 bit wide memories, which may optionally be used by the M9900 CPU. On an 8080 system, the status bus signals may also be read from the data bus when PSYNC(76) is high (an artifact of the way the 8080 supplies these signals). The M9900 CPU does not include logic to place these signals on the data bus, since it was felt that no board designer would count on this quirk of the 8080. Few do.

5.6. Reset and Load Generator

The reset and load generation subsystem generates the two nonmaskable interrupts for the 9900. The subsystem is designed so that the reset switch on the S-100 mainframe may generate either signal. The 9900 reset traps to address zero, while the load causes a trap to address FFFC, so by strapping the CPU board properly, the user may run with ROM at either end of memory. The reset and load subsystem also decodes the 9900 instructions that force a reset or load operation, and generate signals equivalent to performing the actions via the external switches. A power-up reset circuit causes the CPU to start automatically at the selected address after power is stable. The power on clear signal, POC-(99), is generated.

5.7. Interrupt Controller

The 9900 features 16 level priority interrupts within the chip, so only a latch to stabilize the eight interrupt requests on the S-100 bus and a priority encoder are required to provide 8 level interrupts. The other 8 9900 interrupts were not used, and the 8080 trick of requesting an interrupt via the PINT-(73) line was not implemented. The user can easily enough strap PINT-(73) to one of the vectored interrupt lines if this is required.

6. S-100 Bus Ambiguities

In the process of designing the M9900 CPU and getting it to work with a wide variety of S-100 devices, several problems cropped up repeatedly. These problems resulted from ambiguities in the definition of the S-100 bus being interpreted differently by various peripheral designers. The most serious of these ambiguities are discussed below.

6.1. Wait State Timing Reference

In peripherals that stretch the bus cycle with the PRDY(72) signal, there is no convention regarding which pin defines the start of a memory or I/O cycle. Some boards use PSYNC(76), others use the status bus signals or MWRITE(68). The M9900 CPU originally generated the status bus signals immediately at the start of the memory cycle rather than waiting 200 nanoseconds like the 8080, and this caused some boards to miss their wait states because they were starting one-shots to time the wait state when the status bus signals changed. The early status signals from the M9900 caused the one-shot to expire before the processor sampled PRDY(72). The final M9900 design delays the status bus signals for compatibility. Since there are many ways to decide what kind of cycle is being performed from the signals on the bus, there are correspondingly many ways to time the cycle. It seems unreasonable to expect every processor to reproduce all the 8080 signal timing relationships exactly, so a simple definition of what is to be used would be a great boon to CPU and peripheral board designers both. It would be nice if we decided to make MWRITE(68) universal. Then we could count on SINP(46), SOUT(45), SMEMR(47), and MWRITE(68) as four simple signals that identify the current cycle.

6.2. DMA Protocol

The whole area of DMA protocol on the S-100 bus is very loosely defined. The statement that the DMA device must keep PHOLD-(74) low until the processor responds with PHLDA(26) would ease one area of confusion. The existence of separate disable lines for each of the buses is too ingrained in ingenious bootstrap schemes to do away with at this late date, however, some statement such as "The disable lines will never be activated unless the PHLDA(26) line is active" would be a step in the right direction.

6.3. Utility Clocks

An 8080 on the S-100 bus provides perfectly good 2 Mhz clocks on three separate pins. Boards which need a clock for general timing such as generating communications baud rates or implementing a real time clock have no particular reason to choose one over the other. This is a problem when trying to use a processor that runs at a speed other than 2 Mhz. We should define PH1(24) and PH2(25) as CPU clocks which run at the processor clock rate and to which memory timing is relative. CLOCK-(49) should be defined as a 2 Mhz utility clock signal which bears no specific timing relationship to the CPU clock.

7. Unique Features of the Design

The M9900 CPU has succeeded in interfacing the TMS9900 to the S-100 bus. The CPU is a single board system that will run with most existing S-100 peripherals and memories. It is, in fact, more compatible with 8080 peripherals than many Z-80 boards, particularly in interfacing to dynamic memories. The major innovation in S-100 bus architecture introduced by the M9900 CPU is its ability to use the S-100 bus for 16 bit parallel data transfers.

7.1. Sixteen Bit Mode

Obviously, forcing a 16 bit parallel processor to do all of its memory accesses in byte-sized chunks reduces its performance compared to a fully parallel system. In most applications, the increased power of the TMS9900 instruction set results in such a performance gain that the overhead introduced by the extra memory accesses is overcome. For those applications which require the greatest CPU power, the M9900 CPU allows the use of 16 bit parallel memory in the S-100 chassis. The key to this option is the signal that Marinchip Systems has designated SXTN-(60). When a memory cycle starts, if this signal is pulled low by the addressed memory board within 125 nanoseconds, the SYNC cycle will be aborted, and the memory transfer will be performed 16 bits parallel, using the S-100 data out bus for the high byte and the S-100 data in bus for the low byte. A memory cycle performed in 16 bit mode will complete in one microsecond (assuming no wait states), compared to three microseconds if conventional S-100 memories are used. This can result in performance gains approaching three-to-one. Since the memory itself informs the processor if it is capable of 16 bit operation, it is possible to mix conventional 8 bit and 16 bit memory in the same chassis. Obviously, a 16 bit memory is a rather special beast, especially so if it is capable of operating in 8 bit mode for DMA transfers. None the less, the design of such a memory is a straightforward task beside the design of the M9900 CPU itself.

7.2. CRU I/O Signals

Little has been said about the TMS9900's unique bit-addressable I/O scheme, the Communications Register Unit (CRU). This mechanism performs bit-serial transfers from the processor to external devices within a 4096 bit address space. This approach permits peripheral chips to be built with few costly pins and no external logic. For example, the UART for the 9900 family is an 18 pin device, compared with the 28 to 40 pins required with other microprocessors. Since the CRU I/O was not used in interfacing to the S-100 bus, it was simply brought to the bus on three unused pins. Future 9900 peripherals may be built to take advantage of this unique I/O system.

8. Board Operation Details

This section of the manual discusses the specific design and operation of the M9900 CPU board. The reader should refer to the Texas Instruments TMS9900 data manual for details on the internal operation of the microprocessor, and to the M9900 CPU schematic drawings (furnished with the kit or assembled unit) for specifics of the circuits described herein.

M9900 CPU Theory of Operation

Throughout this section and the M9900 CPU schematics, a signal name ending in the digit "1" is an active-high signal, while a name which ends in "0" denotes a signal which is asserted when low.

8. 1. Power

Power is supplied via self-contained regulator circuits in the S-100 bus tradition. Voltages generated on the board are +5, -5, and +12.

8. 2. Clocks

Timing for the logic on the board, the 9900 processor circuit, and the driven signals of the S-100 bus are generated by means of a 74LS362 clock generator circuit. This includes a 32 MHz crystal controlled oscillator and logic which provides a four phase non-overlapping clock repeating at 2MHz. The four high voltage outputs control the 9900 microprocessor, while four TTL compatible outputs and their inversions are used for timing the remainder of the logic on the board. Notice that phase two of the 9900 internal cycle is synchronized with the S-100 bus phase one clock signal; this skewing of the four CLK signals provides a stable address during the entire PSYNC and the remainder of the S-100 cycles. Regarding nomenclature here, CLK10 is a TTL clock signal which is low during phase one clock time and high otherwise; this is the 9900's phase two time. CLK11 is the inverted, high-active clock one. Notice that it directly produces the S-100 bus clock one signal PH1. CLK20 and CLK21 are similarly the low- and high-active clocks of the second quarter cycle and correspond to 9900 phase three; CLK30 and CLK31, the third, during phase four; and CLK40 and CLK41, the fourth, during 9900 phase one.

8. 3. Memory Read Cycles

Memory read cycles are initiated by the logic in response to the active (low) MEMEN output of the 9900 at CLK2 time (9900 phase three) with DBIN also active (high). Two S-100 memory cycles are generated for each 9900 memory access. The address drivers copy fifteen bits from the 9900 to the S-100 address lines. The least significant address bit is derived from the flipflop FSTHLF, which remains set only during the first of two S-100 eight bit memory accesses and thus addresses the even (and later the odd) byte of the 9900 memory access. The leading edge of CLK2 sets flipflop RUN. Bus signal PSYNC goes active (and remains active until the next CLK2). As CLK3 concludes, flipflop XSYNC is set, and immediately signal XRUN1 goes active which enables the bus signal SMEMR. At the leading edge of CLK2 flipflop ENDSYNC is set and this turns off PSYNC. This also activates bus signal PDBIN.

The next CLK1 sets flipflop TWT3, which remains active during those S-100 bus cycles known as TW and T3. The circuitry of flipflops WAITING and PWAIT (which directly generate S-100 bus signal PWAIT) now stall the logic until bus signal PRDY is high at a CLK3 trailing edge, at which time flipflop WAITING clears. At the next CLK1 the data is latched by signal CLKIN8HIGH1 for later presentation to the 9900 high byte data inputs, and PWAIT is cleared. Flipflop XSYNC is cleared with the next CLK4 and signal XRUN1 goes low which disables SMEMR. CLK1 clears TWT3 indicating the conclusion of the first half of the cycle.

With XSYNC off at CLK2 time, ENDSYNC goes off for a full clock cycle, generating a second PSYNC on the S-100 bus. During this second cycle FSTHLF is off and the address is correct for the second byte of the memory access. The logic is then again interlocked by WAITING and PWAIT, but this time the READY input to the 9900 is at last activated. The remaining eight bits are latched for the 9900 low byte data inputs with signal CLKIN1. CLK3 turns off the signal RUN, and ENDSYNC, XSYNC and XRUN1 go inactive in succession. The 9900 consumes the sixteen bits of data at CLK4 time, 9900 phase one. This is the final clock phase of a complete cycle; the 9900 will turn off MEMEN with the next clock unless it requires another memory cycle in immediate succession. CLK1 turns off TWT3 and the logic is completely idle until MEMEN is again active at CLK2.

Other S-100 bus signals are simulated in a straightforward manner.

8.4. Memory Write Cycles

Memory write cycles function in a similar manner, except that data is gated from the 9900 to the eight S-100 data out buses, half at a time, and two write cycles are simulated. Bus signal PWR is active (low) in response to flipflop TWT3 as is required.

8.5. Input / Output

Input/output with S-100 devices is accomplished by using two pages of memory mapped I/O implemented by logic on the M9900 CPU board. The 256 sixteen bit memory locations between F000 and F1FE hexadecimal are mapped into the 256 input port addresses on the 8080, and the 256 word addresses from F200 to F3FE are mapped into the 8080 output ports. A move from the input area (F000 to F1FE) will generate an S-100 bus cycle identical to that generated by an 8080 IN instruction. A move to the output area (F200 to F3FE) will generate a bus cycle identical to an 8080 OUT instruction. Note that the 8080 I/O port number is doubled and then added to the base address of the proper memory mapped area to compute the I/O address. For example, port 16 hex on the 8080 would be read by moving from address F02C and written by moving to address F22C. The byte sent to the port on a write is taken from the low-order 8 bits of the word sent to the I/O address. On input, the upper 8 bits of the word transferred are pulled to zero by diodes, and the input byte appears in the low byte. The status bus generator will correctly generate SINP or SOUT for I/O cycles. The reason for providing two separate areas for input and output lies in the fact that the MOV instruction reads the destination before storing the data there. If a simple memory-mapped approach were taken, it would be impossible to support full-duplex operation on the I/O ports because sending data to a port would cause data to be read from that port and discarded.

8.6. Reset and Load Generation

RESET and LOAD features of the 9900 are incorporated in this design. Two flipflops form a timer which drives the RESET input of the 9900 and also the S-100 bus POC. When power is turned on, capacitor C2 charges and eventually sets the 74LS362 flipflop releasing RESET. If S-100 bus PRESET is activated or a RSET instruction is decoded, a flipflop is set which discharges C2 and initiates a RESET cycle just as at power on time. When a LOAD instruction is decoded a flipflop is set to activate the LOAD input of the 9900. (This causes a software context switch through the vector at memory location X'FFFC'.) With an wired option on the board, the PRESET line and power on sequence can also set the 9900's LOAD input, so that an initial context switch is defined by the memory contents at X'FFFC' instead of by the RESET context pointer at X'0000'.

8.7. Interrupts

Interrupts can be presented to the processor from the S-100 bus. The eight interrupt request lines are latched and the most significant of those which are active is encoded and presented to the 9900 on lines IC0 through IC3. The 9900 interrupt request line, INTREQ, is activated. When the 9900 sees the request, and if that priority interrupt is allowed, a context switch is performed, and that priority of interrupt is then locked out, as are interrupts of lower priority. A higher priority interrupt will be latched, however, and will cause another context switch. Bus signal SINTA is used by the M9900 CPU for a signal relevant to sixteen bit mode memory accesses. Hence, it cannot be used to reset an interrupt request.

8.8. Halt

The idle state as indicated by bus SHLTA is derived from the 9900 IDLE instruction. This is decoded in a manner similar to the RSET and LOAD instructions and sets a flipflop. The flipflop remains set until a memory cycle occurs, whereupon signal XSYNCO resets it.

8.9. Status Bus

The op-code fetch bus SM1 is a direct copy of the microprocessor's IAQ signal and remains active through the entire sixteen bit fetch cycle.

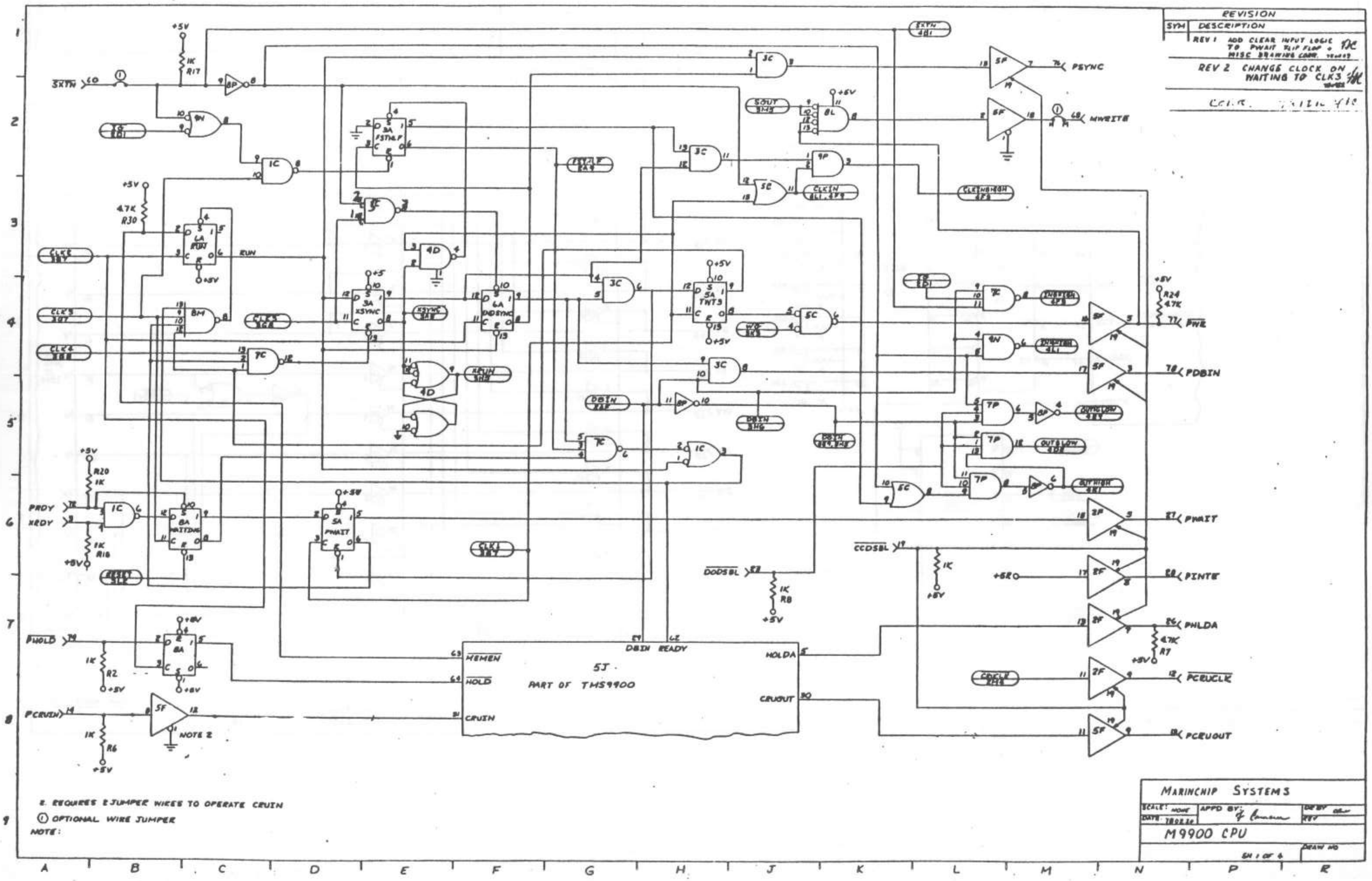
Signal SINTA has been assigned a new meaning (since its old meaning does not apply). SINTA is high during I/O operations and low during memory operations. Its value is not defined at other times.

8.10. Sixteen Bit Mode

Sixteen bit mode is an extension of the S-100 bus definition which enables sixteen bit transfers to be performed in two machine cycles with special memory and I/O devices, even intermixed with conventional eight bit memory and I/O boards. This is accomplished with a new signal on the bus, SXTNO, which if pulled low by the memory or I/O board enables the sixteen bit data path. The feature is more fully described in documentation for Marinchip Systems sixteen bit equipment. A timing diagram is available for the truly eager.

9. Summary

The M9900 CPU is a happy marriage of the advanced 16 bit architecture of the TMS9900 and the widely accepted S-100 bus. The interfacing of the 9900 revealed several characteristics and problems of the S-100 bus that deserve to be standardized. The design of the M9900 CPU took a very conservative approach to the S-100 bus, and tried to generate virtually all the signals provided by an 8080, with the same timing relationships. Careful work by a standards group would have permitted a much more straightforward design, and will greatly ease the task of others who wish to interface foreign processors to the bus. The extensions to the S-100 bus made by Marinchip Systems were to provide the unique performance and features of the 9900 within the existing bus architecture.



REVISION	
SYM	DESCRIPTION
	REV 1 ADD CLEAR INPUT LOGIC TO PWAIT FLIP FLOP + MISC DRAWING CORR.
	REV 2 CHANGE CLOCK ON WAITING TO CLK3
	CCLR. [illegible]

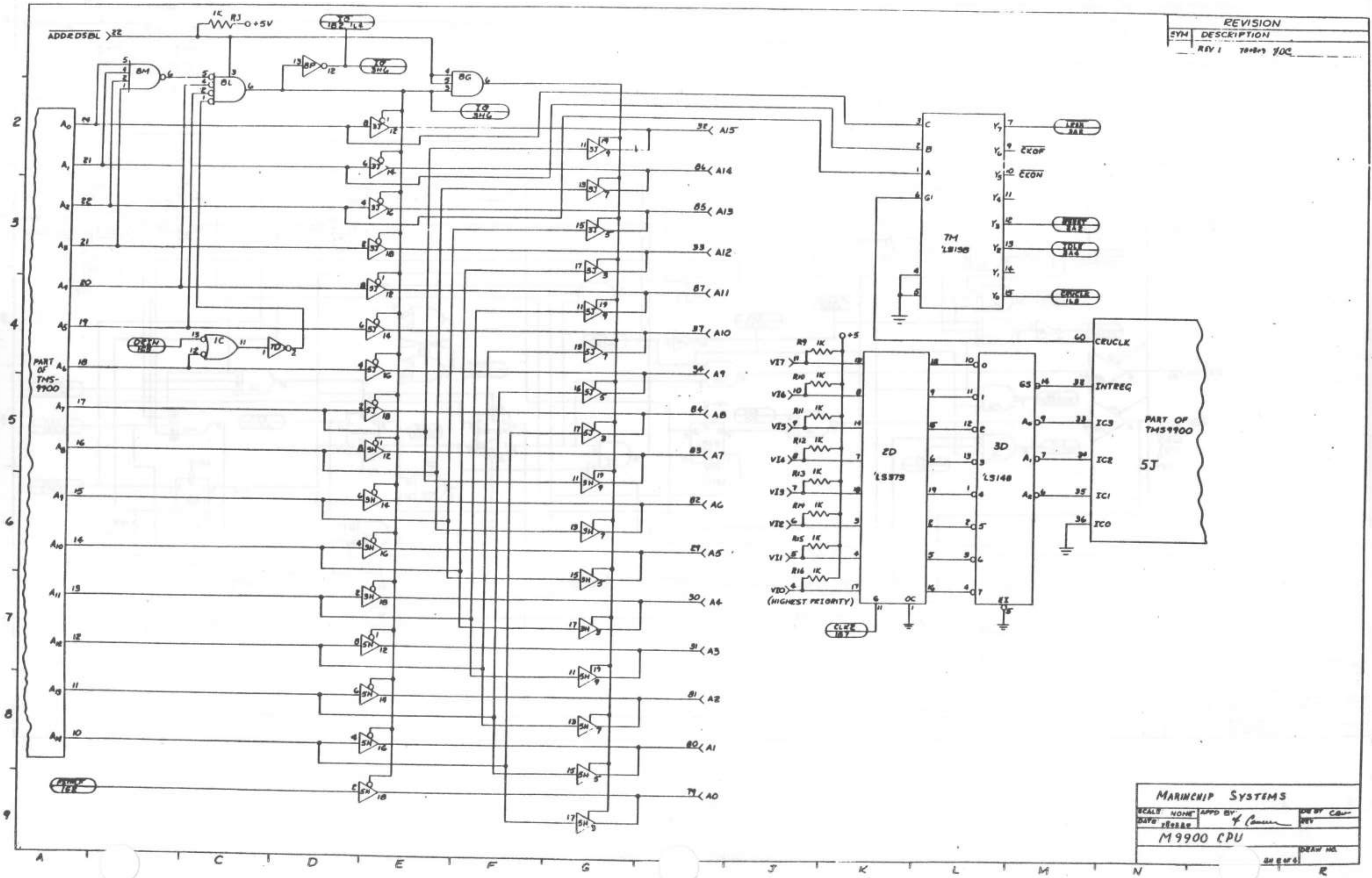
NOTE:
① OPTIONAL WIRE JUMPER
2. REQUIRES 2 JUMPER WIRES TO OPERATE CRUIN

MARINCHIP SYSTEMS

SCALE: NONE
DATE: 780830
APPD BY:
DR BY
REV

M9900 CPU

SH 1 OF 4
DRAW NO

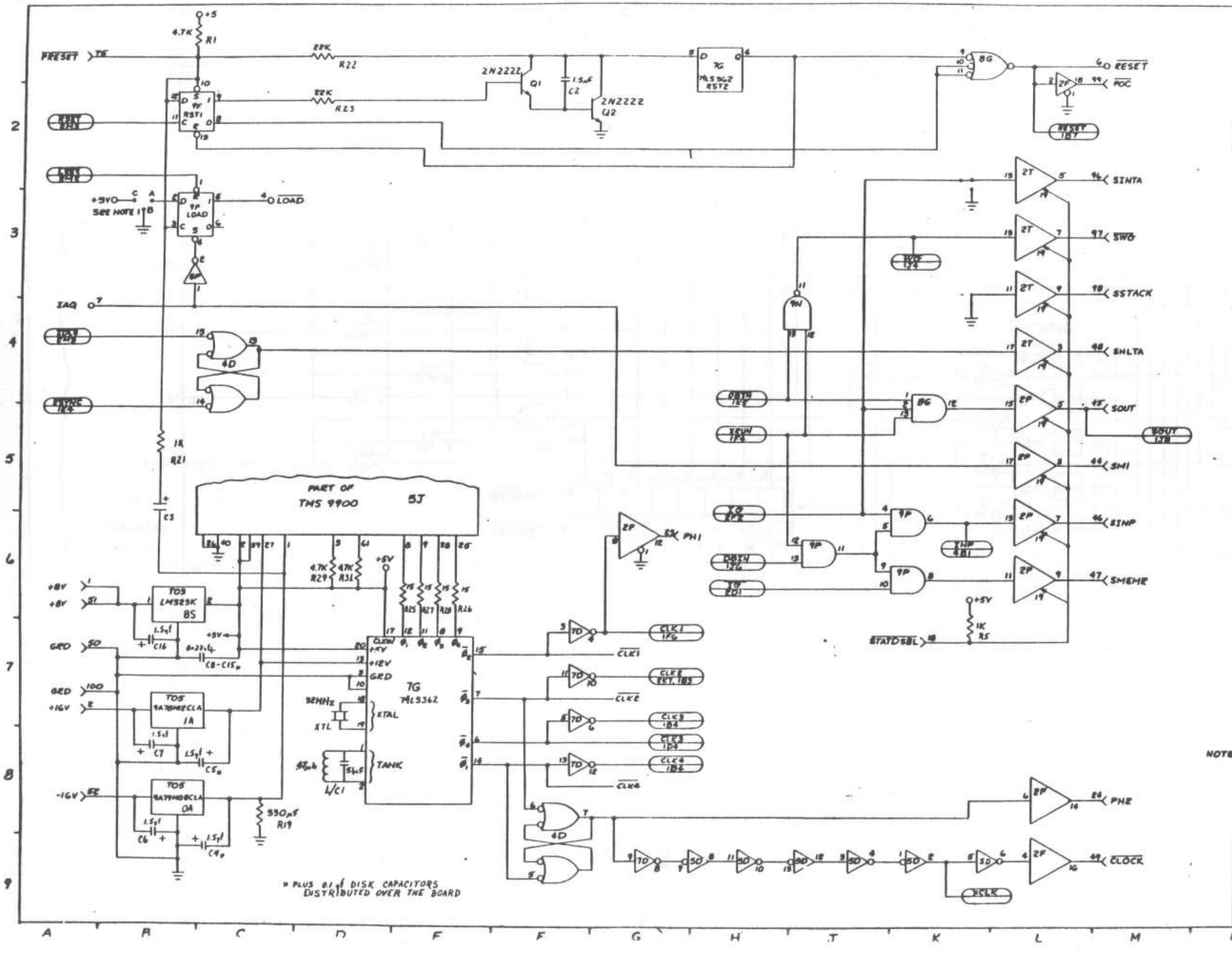


REVISION	
SYM	DESCRIPTION
	REV 1 780809 JOC

MARINCHIP SYSTEMS

SCALE: NONE
DATE: 780820
APPD BY: J Cameron
DR BY: CGW
REV
M9900 CPU
SH 2 OF 4
DRAW NO.

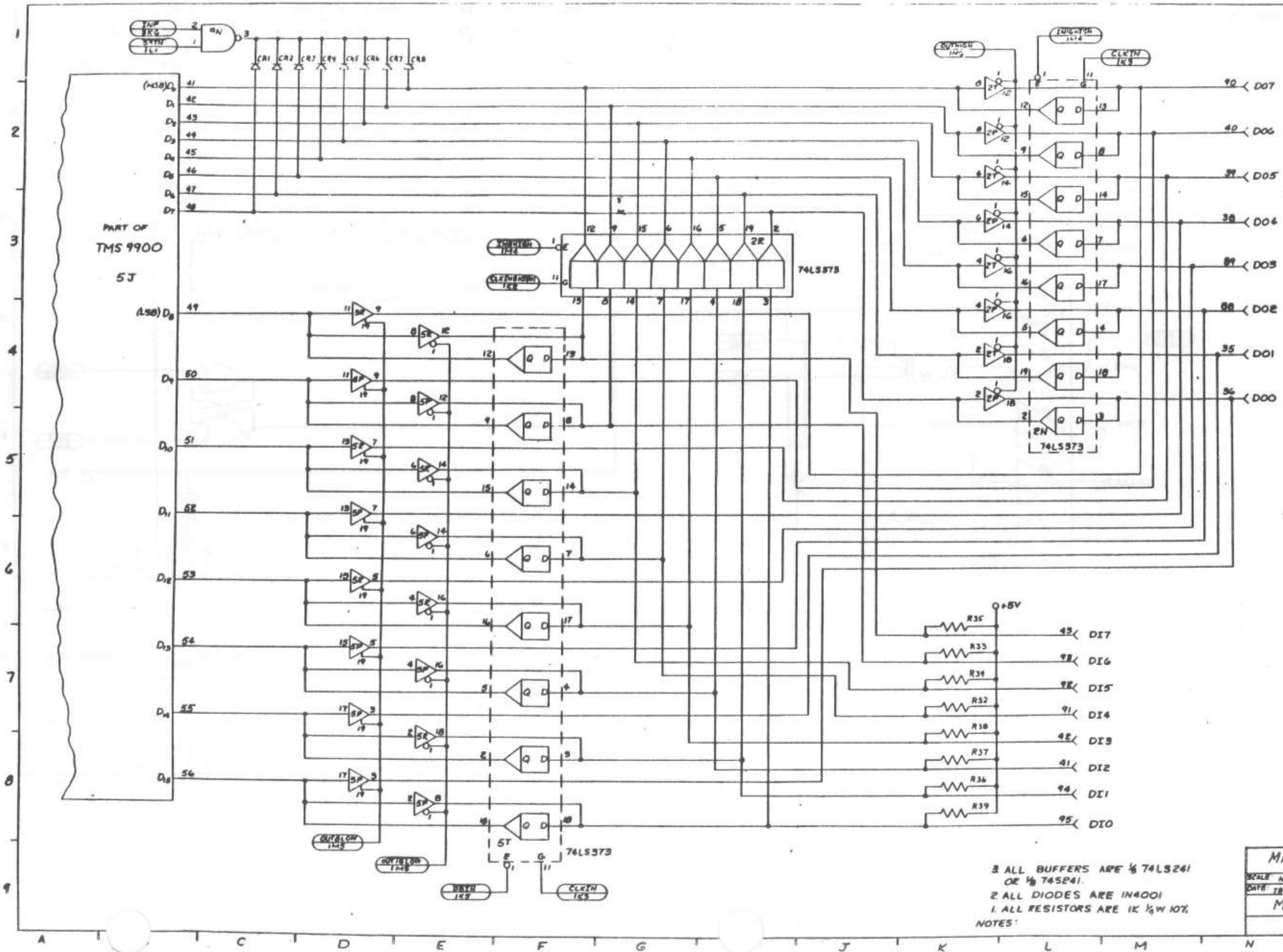
REVISION	
SYM	DESCRIPTION
	REV 1 780809



NOTE 1 STRAP 'A' TO 'B' FOR LOAD OPERATION ON PRESET. STRAP TO 'C' FOR RESET ON PRESET.

MARINCHIP SYSTEMS		
SCALE: NONE DATE: 780230	APPD BY:	DR BY CBW REV
M9900 CPU		
SH 8 OF 4		DRAW NO.

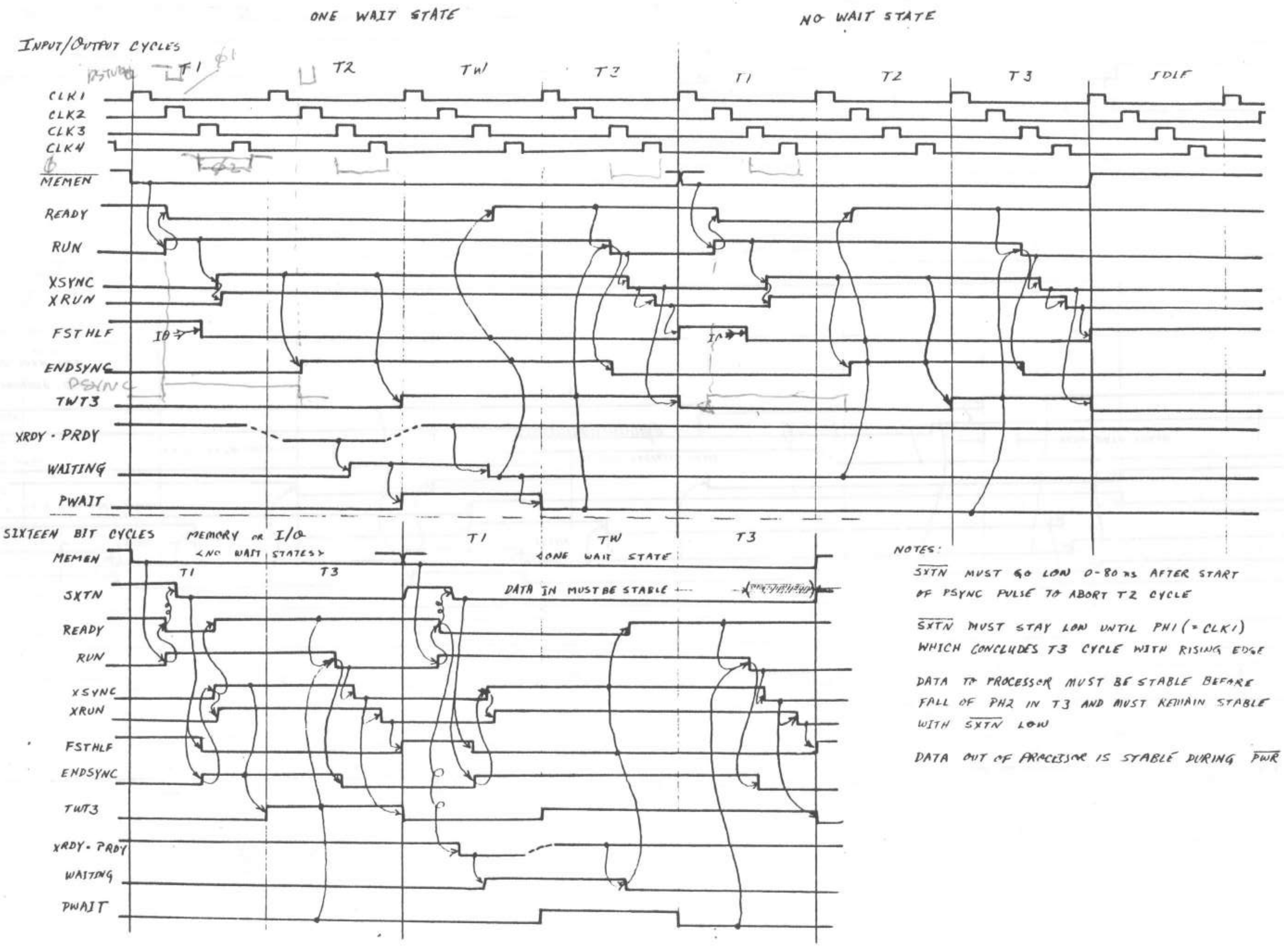
REVISION	
SYM	DESCRIPTION
	REV 1 790809 YdC



NOTES:
1. ALL RESISTORS ARE 1K ¼W 10%
2. ALL DIODES ARE IN4001
3. ALL BUFFERS ARE ⅛ 74LS241 OR ⅛ 74S241.

MARINCHIP SYSTEMS

SCALE NONE	APPD BY:		DR BY CAM
DATE 780520			REV
M9900 CPU			
		SH 4 OF 4	DRAW NO.



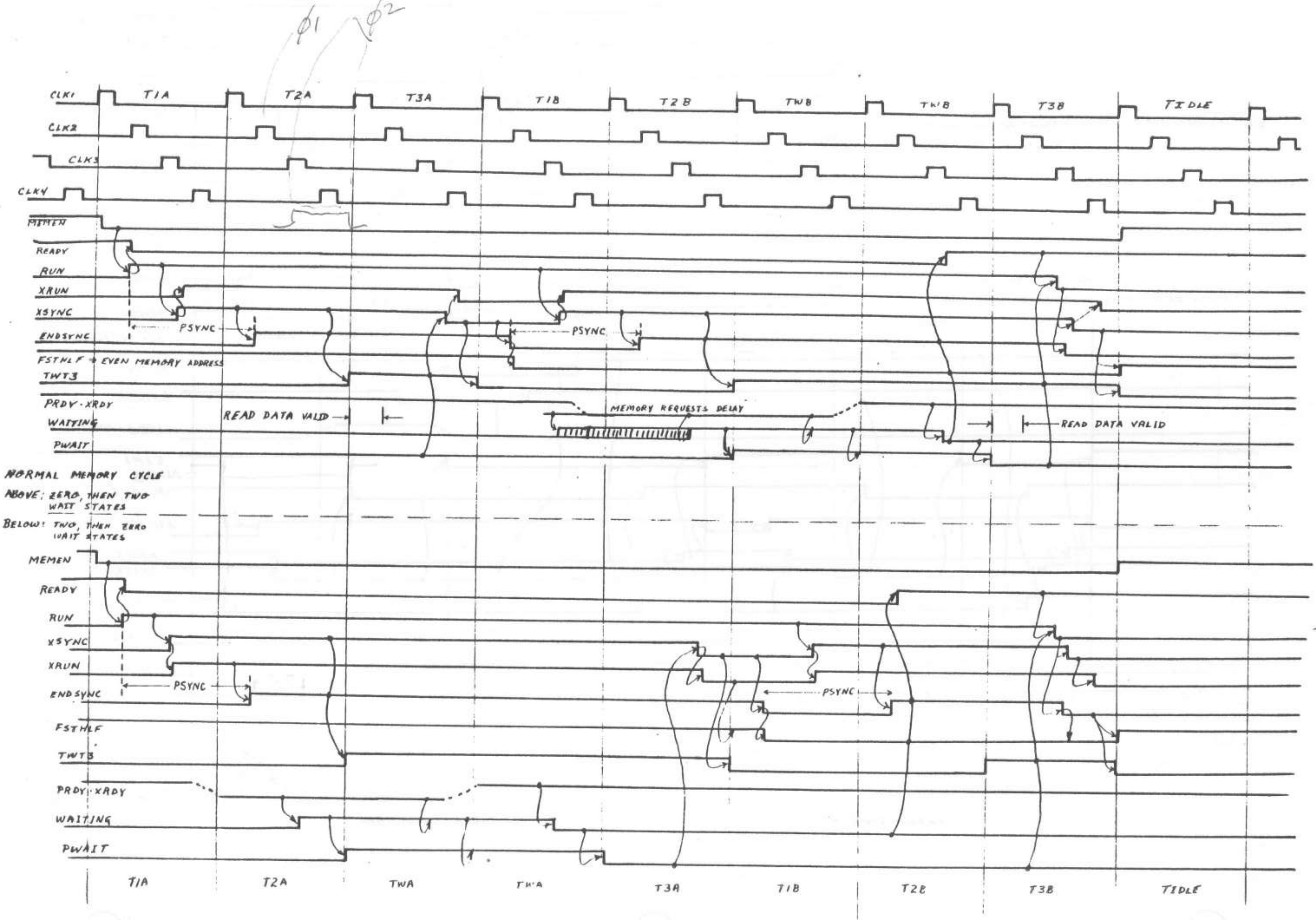
NOTES:

$\overline{SXTN}$ MUST GO LOW 0-80 ns AFTER START OF PSYNC PULSE TO ABORT T2 CYCLE

$\overline{SXTN}$ MUST STAY LOW UNTIL PHI (= CLK1) WHICH CONCLUDES T3 CYCLE WITH RISING EDGE

DATA TO PROCESSOR MUST BE STABLE BEFORE FALL OF PH2 IN T3 AND MUST REMAIN STABLE WITH $\overline{SXTN}$ LOW

DATA OUT OF PROCESSOR IS STABLE DURING $\overline{PWR}$



NORMAL MEMORY CYCLE
ABOVE: ZERO, THEN TWO WAIT STATES
BELOW: TWO, THEN ZERO WAIT STATES

Information contained in this publication is believed to be accurate and reliable. However, responsibility is assumed neither for its use nor for any infringement of patents or rights of others that may result from its use. No license is granted by implication or otherwise under any patent or patent right of Texas Instruments or others.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

LIST OF ILLUSTRATIONS

LIST OF TABLES

1. INTRODUCTION

1.1 DESCRIPTION

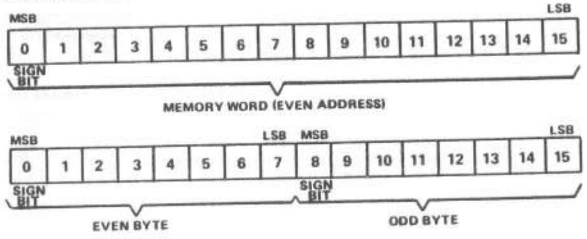
The TMS 9900 microprocessor is a single-chip 16-bit central processing unit (CPU) produced using N-channel silicon-gate MOS technology (see Figure 1). The instruction set of the TMS 9900 includes the capabilities offered by full minicomputers. The unique memory-to-memory architecture features multiple register files, resident in memory, which allow faster response to interrupts and increased programming flexibility. The separate bus structure simplifies the system design effort. Texas Instruments provides a compatible set of MOS and TTL memory and logic function circuits to be used with a TMS 9900 system. The system is fully supported by software and a complete prototyping system.

1.2 KEY FEATURES

- 16-Bit Instruction Word
- Full Minicomputer Instruction Set Capability Including Multiply and Divide
- Up to 65,536 Bytes of Memory
- 3.3 – MHz Speed
- Advanced Memory-to-Memory Architecture
- Separate Memory, I/O, and Interrupt-Bus Structures
- 16 General Registers
- 16 Prioritized Interrupts
- Programmed and DMA I/O Capability
- N-Channel Silicon-Gate Technology

2. ARCHITECTURE

The memory word of the TMS 9900 is 16 bits long. Each word is also defined as 2 bytes of 8 bits. The instruction set of the TMS 9900 allows both word and byte operands. Thus, all memory locations are on even address boundaries and byte instructions can address either the even or odd byte. The memory space is 65,536 bytes or 32,768 words. The word and byte formats are shown below.



2.1 REGISTERS AND MEMORY

The TMS 9900 employs an advanced memory-to-memory architecture. Blocks of memory designated as workspace replace internal-hardware registers with program-data registers. The TMS 9900 memory map is shown in Figure 2. The first 32 words are used for interrupt trap vectors. The next contiguous block of 32 memory words is used by the extended operation (XOP) instruction for trap vectors. The last two memory words, FFFC$_{16}$ and FFFE$_{16}$, are used for the trap vector of the LOAD signal. The remaining memory is then available for programs, data, and workspace registers. If desired, any of the special areas may also be used as general memory.

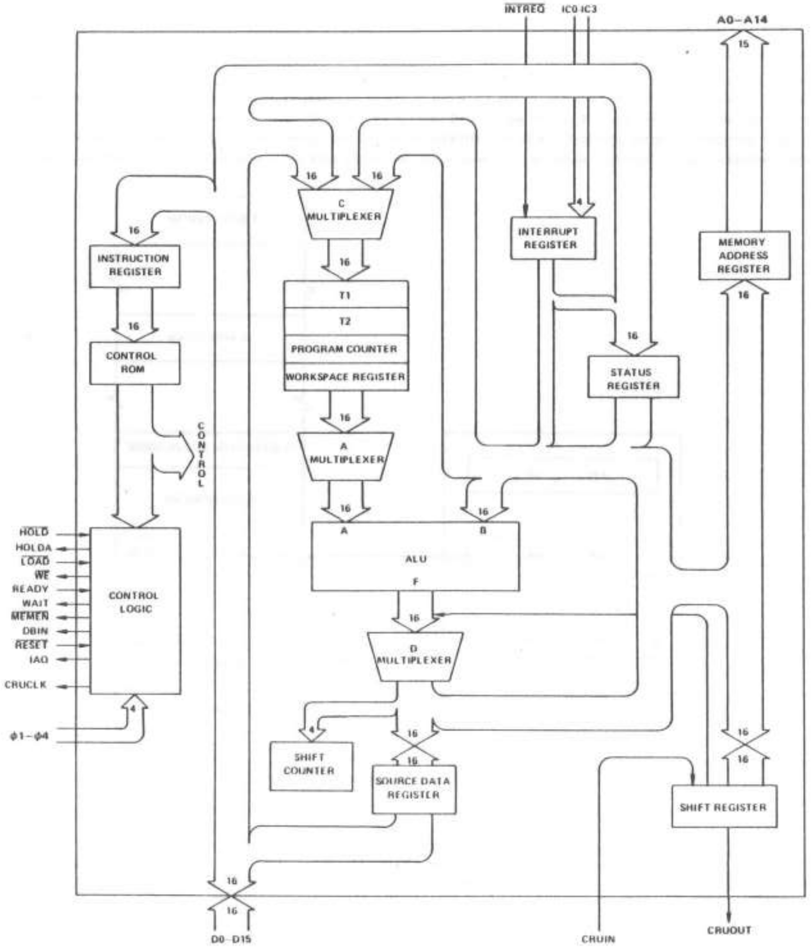
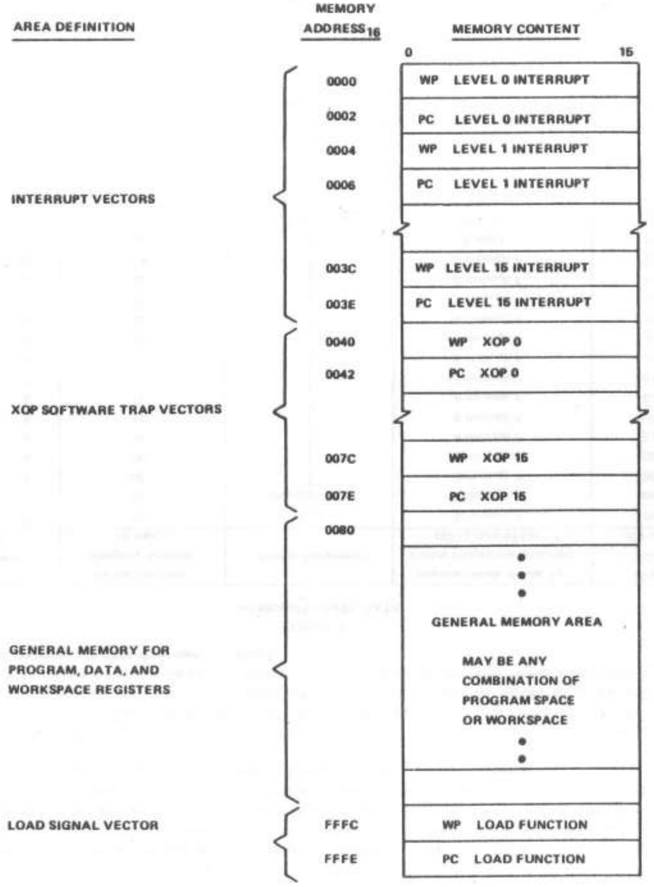


FIGURE 1 – ARCHITECTURE

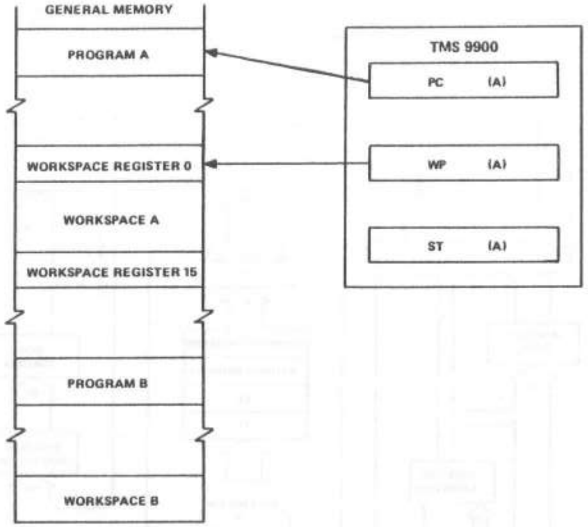


AREA DEFINITION	MEMORY ADDRESS$_{16}$	MEMORY CONTENT (0 … 15)
INTERRUPT VECTORS	0000	WP LEVEL 0 INTERRUPT
	0002	PC LEVEL 0 INTERRUPT
	0004	WP LEVEL 1 INTERRUPT
	0006	PC LEVEL 1 INTERRUPT
	003C	WP LEVEL 15 INTERRUPT
	003E	PC LEVEL 15 INTERRUPT
XOP SOFTWARE TRAP VECTORS	0040	WP XOP 0
	0042	PC XOP 0
	007C	WP XOP 15
	007E	PC XOP 15
GENERAL MEMORY FOR PROGRAM, DATA, AND WORKSPACE REGISTERS	0080	• • • GENERAL MEMORY AREA MAY BE ANY COMBINATION OF PROGRAM SPACE OR WORKSPACE • •
LOAD SIGNAL VECTOR	FFFC	WP LOAD FUNCTION
	FFFE	PC LOAD FUNCTION

FIGURE 2 – MEMORY MAP

Three internal registers are accessible to the user. The program counter (PC) contains the address of the instruction following the current instruction being executed. This address is referenced by the processor to fetch the next instruction from memory and is then automatically incremented. The status register (ST) contains the present state of the processor and will be further defined in Section 3.4. The workspace pointer (WP) contains the address of the first word in the currently active set of workspace registers.

A workspace-register file occupies 16 contiguous memory words in the general memory area (see Figure 2). Each workspace register may hold data or addresses and function as operand registers, accumulators, address registers, or index registers. During instruction execution, the processor addresses any register in the workspace by adding the register number to the contents of the workspace pointer and initiating a memory request for the word. The relationship between the workspace pointer and its corresponding workspace is shown below.



The workspace concept is particularly valuable during operations that require a context switch, which is a change from one program environment to another (as in the case of an interrupt) or to a subroutine. Such an operation, using a conventional multi-register arrangement, requires that at least part of the contents of the register file be stored and reloaded. A memory cycle is required to store or fetch each word. By exchanging the program counter, status register, and workspace pointer, the TMS 9900 accomplishes a complete context switch with only three store cycles and three fetch cycles. After the switch the workspace pointer contains the starting address of a new 16-word workspace in memory for use in the new routine. A corresponding time saving occurs when the original context is restored. Instructions in the TMS 9900 that result in a context switch include:

1. Branch and Load Workspace Pointer (BLWP)
2. Return from Subroutine (RTWP)
3. Extended Operation (XOP).

Device interrupts, RESET, and LOAD also cause a context switch by forcing the processor to trap to a service subroutine.

2.2 INTERRUPTS

The TMS 9900 employs 16 interrupt levels with the highest priority level 0 and lowest level 15. Level 0 is reserved for the RESET function and all other levels may be used for external devices. The external levels may also be shared by several device interrupts, depending upon system requirements.

The TMS 9900 continuously compares the interrupt code (IC0 through IC3) with the interrupt mask contained in status-register bits 12 through 15. When the level of the pending interrupt is less than or equal to the enabling mask level (higher or equal priority interrupt), the processor recognizes the interrupt and initiates a context switch following completion of the currently executing instruction. The processor fetches the new context WP and PC from the interrupt vector locations. Then, the previous context WP, PC, and ST are stored in workspace registers 13, 14, and 15, respectively, of the new workspace. The TMS 9900 then forces the interrupt mask to a value that is one less than the level of the interrupt being serviced, except for level-zero interrupt, which loads zero into the mask. This allows only interrupts of higher priority to interrupt a service routine. The processor also inhibits interrupts until the first instruction of the service routine has been executed to preserve program linkage should a higher priority interrupt occur. All interrupt requests should remain active until recognized by the processor in the device-service routine. The individual service routines must reset the interrupt requests before the routine is complete.

If a higher priority interrupt occurs, a second context switch occurs to service the higher priority interrupt. When that routine is complete, a return instruction (RTWP) restores the first service routine parameters to the processor to complete processing of the lower-priority interrupt. All interrupt subroutines should terminate with the return instruction to restore original program parameters. The interrupt-vector locations, device assignment, enabling-mask value, and the interrupt code are shown in Table 1.

TABLE 1
INTERRUPT LEVEL DATA

Interrupt Level	Vector Location (Memory Address In Hex)	Device Assignment	Interrupt Mask Values To Enable Respective Interrupts (ST12 thru ST15)	Interrupt Codes IC0 thru IC3
(Highest priority) 0	00	Reset	0 through F*	0000
1	04	External device	1 through F	0001
2	08		2 through F	0010
3	0C		3 through F	0011
4	10		4 through F	0100
5	14		5 through F	0101
6	18		6 through F	0110
7	1C		7 through F	0111
8	20		8 through F	1000
9	24		9 through F	1001
10	28		A through F	1010
11	2C		B through F	1011
12	30		C through F	1100
13	34		D through F	1101
14	38		E and F	1110
(Lowest priority) 15	3C	External device	F only	1111

*Level 0 can not be disabled.

The TMS 9900 interrupt interface utilizes standard TTL components as shown in Figure 3. Note that for eight or less external interrupts a single SN74148 is required and for one external interrupt INTREQ is used as the interrupt signal with a hard-wired code IC0 through IC3.

2.3 INPUT/OUTPUT

The TMS 9900 utilizes a versatile direct command-driven I/O interface designated as the communications-register unit (CRU). The CRU provides up to 4096 directly addressable input bits and 4096 directly addressable output bits. Both input and output bits can be addressed individually or in fields of from 1 to 16 bits. The TMS 9900 employs three dedicated I/O pins (CRUIN, CRUOUT, and CRUCLK) and 12 bits (A3 through A14) of the address bus to interface with the CRU system. The processor instructions that drive the CRU interface can set, reset, or test any bit in the CRU array or move between memory and CRU data fields.

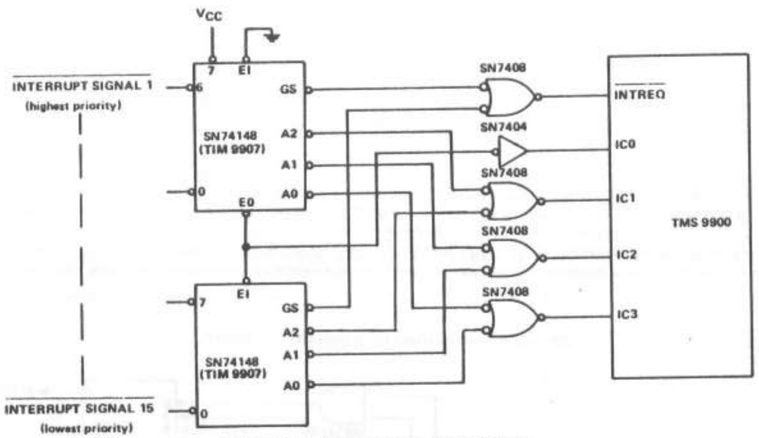


FIGURE 3 – TMS 9900 INTERRUPT INTERFACE

2.4 SINGLE-BIT CRU OPERATIONS

The TMS 9900 performs three single-bit CRU functions: test bit (TB), set bit to one (SBO), and set bit to zero (SBZ). To identify the bit to be operated upon, the TMS 9900 develops a CRU-bit address and places it on the address bus, A3 to A14.

For the two output operations (SBO and SBZ), the processor also generates a CRUCLK pulse, indicating an output operation to the CRU device, and places bit 7 of the instruction word on the CRUOUT line to accomplish the specified operation (bit 7 is a one for SBO and a zero for SBZ). A test-bit instruction transfers the addressed CRU bit from the CRUIN input line to bit 2 of the status register (EQUAL).

The TMS 9900 develops a CRU-bit address for the single-bit operations from the CRU-base address contained in workspace register 12 and the signed displacement count contained in bits 8 through 15 of the instruction. The displacement allows two's complement addressing from base minus 128 bits through base plus 127 bits. The base address from W12 is added to the signed displacement specified in the instruction and the result is loaded onto the address bus. Figure 4 illustrates the development of a single-bit CRU address.

2.5 MULTIPLE-BIT CRU OPERATIONS

The TMS 9900 performs two multiple-bit CRU operations: store communications register (STCR) and load communications register (LDCR). Both operations perform a data transfer from the CRU-to-memory or from memory-to-CRU as illustrated in Figure 5. Although the figure illustrates a full 16-bit transfer operation, any number of bits from 1 through 16 may be involved. The LDCR instruction fetches a word from memory and right-shifts it to serially transfer it to CRU output bits. If the LDCR involves eight or fewer bits, those bits come from the right-justified field within the addressed byte of the memory word. If the LDCR involves nine or more bits, those bits come from the right-justified field within the whole memory word. When transferred to the CRU interface, each successive bit receives an address that is sequentially greater than the address for the previous bit. This addressing mechanism results in an order reversal of the bits; that is, bit 15 of the memory word (or bit 7) becomes the lowest addressed bit in the CRU and bit 0 becomes the highest addressed bit in the CRU field.

An STCR instruction transfers data from the CRU to memory. If the operation involves a byte or less transfer, the transferred data will be stored right-justified in the memory byte with leading bits set to zero. If the operation involves from nine to 16 bits, the transferred data is stored right-justified in the memory word with leading bits set to zero.

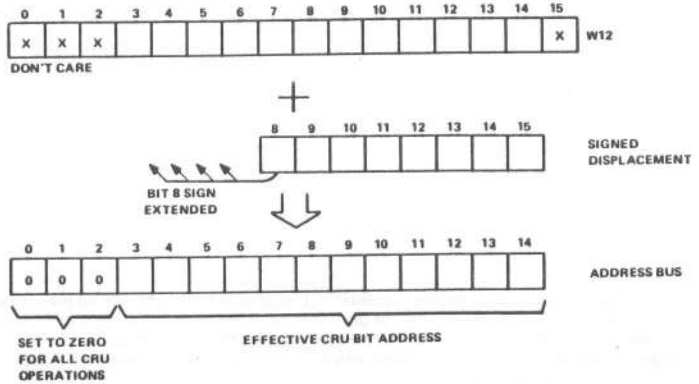
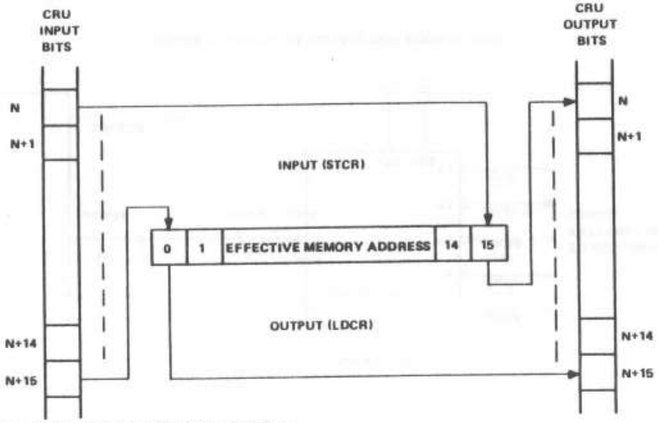


FIGURE 4 – TMS 9900 SINGLE-BIT CRU ADDRESS DEVELOPMENT



N = BIT SPECIFIED BY CRU BASE REGISTER

FIGURE 5 – TMS 9900 LDCR/STCR DATA TRANSFERS

When the input from the CRU device is complete, the first bit from the CRU is the least-significant-bit position in the memory word or byte.

Figure 6 illustrates how to implement a 16-bit input and a 16-bit output register in the CRU interface. CRU addresses are decoded as needed to implement up to 256 such 16-bit interface registers. In system application, however, only the exact number of interface bits needed to interface specific peripheral devices are implemented. It is not necessary to have a 16-bit interface register to interface an 8-bit device.

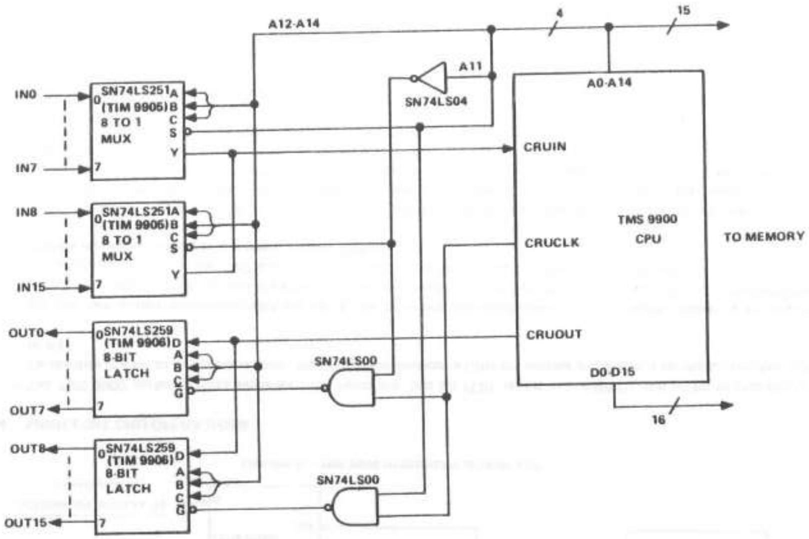


FIGURE 6 – TMS 9900 16-BIT INPUT/OUTPUT INTERFACE

2.6 EXTERNAL INSTRUCTIONS

The TMS 9900 has five external instructions that allow user-defined external functions to be initiated under program control. These instructions are CKON, CKOF, RSET, IDLE, and LREX. These mnemonics, except for IDLE, relate to functions implemented in the 990 minicomputer and do not restrict use of the instructions to initiate various user-defined functions. IDLE also causes the TMS 9900 to enter the idle state and remain until an interrupt, $\overline{\text{RESET}}$, or $\overline{\text{LOAD}}$ occurs. When any of these five instructions are executed by the TMS 9900, a unique 3-bit code appears on the most-significant 3 bits of the address bus (A0 through A2) along with a CRUCLK pulse. When the TMS 9900 is in an idle state, the 3-bit code and CRUCLK pulses occur repeatedly until the idle state is terminated. The codes are:

EXTERNAL INSTRUCTION	A0	A1	A2
LREX	H	H	H
CKOF	H	H	L
CKON	H	L	H
RSET	L	H	H
IDLE	L	H	L

Figure 7 illustrates typical external decode logic to implement these instructions. Note that a signal is generated to inhibit CRU decodes during external instructions.

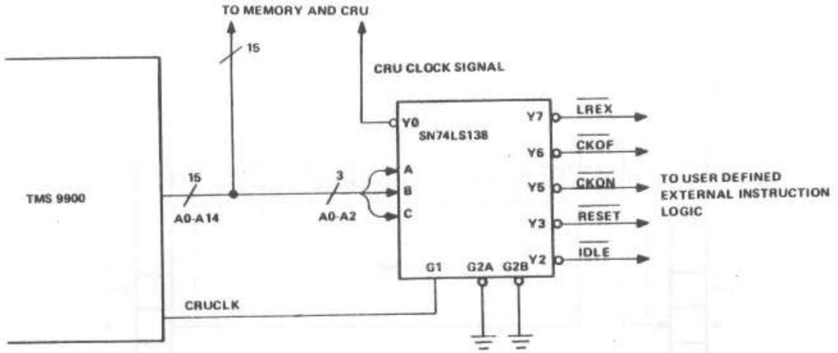


FIGURE 7 – EXTERNAL INSTRUCTION DECODE LOGIC

2.7 LOAD FUNCTION

The $\overline{\text{LOAD}}$ signal allows cold-start ROM loaders and front panels to be implemented for the TMS 9900. When active, $\overline{\text{LOAD}}$ causes the TMS 9900 to initiate an interrupt sequence immediately following the instruction being executed. Memory location FFFC is used to obtain the vector (WP and PC). The old PC, WP and ST are loaded into the new workspace and the interrupt mask is set to 0000. Then, program execution resumes using the new PC and WP.

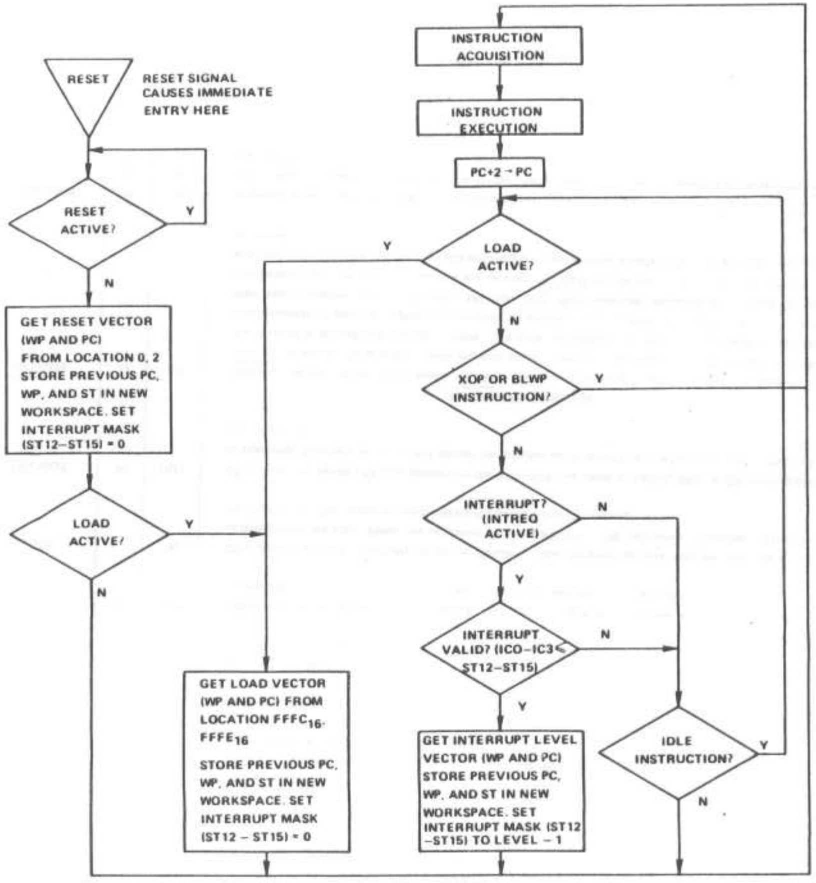

RESET
RESET SIGNAL CAUSES IMMEDIATE ENTRY HERE
RESET ACTIVE?
Y
N
GET RESET VECTOR (WP AND PC) FROM LOCATION 0, 2 STORE PREVIOUS PC, WP, AND ST IN NEW WORKSPACE. SET INTERRUPT MASK (ST12–ST15) = 0
LOAD ACTIVE?
Y
N
GET LOAD VECTOR (WP AND PC) FROM LOCATION FFFC16-FFFE16
STORE PREVIOUS PC, WP, AND ST IN NEW WORKSPACE. SET INTERRUPT MASK (ST12 – ST15) = 0
INSTRUCTION ACQUISITION
INSTRUCTION EXECUTION
PC+2 → PC
LOAD ACTIVE?
Y
N
XOP OR BLWP INSTRUCTION?
Y
N
INTERRUPT? (INTREQ ACTIVE)
N
Y
INTERRUPT VALID? (IC0–IC3≤ ST12–ST15)
N
Y
GET INTERRUPT LEVEL VECTOR (WP AND PC) STORE PREVIOUS PC, WP, AND ST IN NEW WORKSPACE. SET INTERRUPT MASK (ST12 –ST15) TO LEVEL – 1
IDLE INSTRUCTION?
Y
N

FIGURE 8 - TMS 9900 CPU FLOW CHART

2.8 TMS 9900 PIN DESCRIPTION

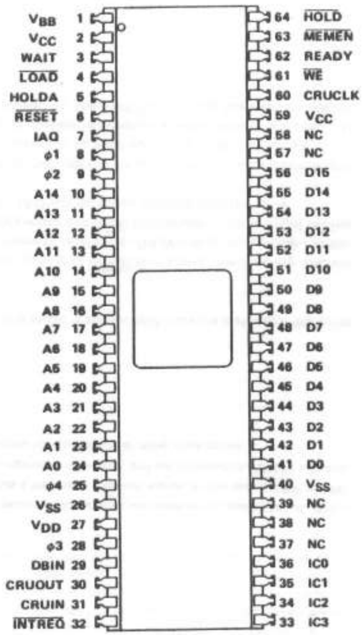
Table 2 defines the TMS 9900 pin assignments and describes the function of each pin.

TABLE 2

TMS 9900 PIN ASSIGNMENTS AND FUNCTIONS

SIGNATURE	PIN	I/O	DESCRIPTION
			ADDRESS BUS
A0 (MSB)	24	OUT	A0 through A14 comprise the address bus. This 3-state bus provides the memory-address vector to the external-memory system when $\overline{\text{MEMEN}}$ is active and I/O-bit addresses and external-instruction addresses to the I/O system when $\overline{\text{MEMEN}}$ is inactive. The address bus assumes the high-impedance state when HOLDA is active.
A1	23	OUT	
A2	22	OUT	
A3	21	OUT	
A4	20	OUT	
A5	19	OUT	
A6	18	OUT	
A7	17	OUT	
A8	16	OUT	
A9	15	OUT	
A10	14	OUT	
A11	13	OUT	
A12	12	OUT	
A13	11	OUT	
A14 (LSB)	10	OUT	
			DATA BUS
D0 (MSB)	41	I/O	D0 through D15 comprise the bidirectional 3-state data bus. This bus transfers memory data to (when writing) and from (when reading) the external-memory system when $\overline{\text{MEMEN}}$ is active. The data bus assumes the high-impedance state when HOLDA is active.
D1	42	I/O	
D2	43	I/O	
D3	44	I/O	
D4	45	I/O	
D5	46	I/O	
D6	47	I/O	
D7	48	I/O	
D8	49	I/O	
D9	50	I/O	
D10	51	I/O	
D11	52	I/O	
D12	53	I/O	
D13	54	I/O	
D14	55	I/O	
D15 (LSB)	56	I/O	
			POWER SUPPLIES
V_{BB}	1		Supply voltage (–5 V NOM)
V_{CC}	2,59		Supply voltage (5 V NOM). Pins 2 and 59 must be connected in parallel.
V_{DD}	27		Supply voltage (12 V NOM)
V_{SS}	26,40		Ground reference. Pins 26 and 40 must be connected in parallel.
			CLOCKS
$\phi1$	8	IN	Phase-1 clock
$\phi2$	9	IN	Phase-2 clock
$\phi3$	28	IN	Phase-3 clock
$\phi4$	25	IN	Phase-4 clock

TMS 9900 PIN ASSIGNMENTS



NC – No Internal connection

TABLE 2 (CONTINUED)

SIGNATURE	PIN	I/O	DESCRIPTION
			BUS CONTROL
DBIN	29	OUT	Data bus in. When active (high), DBIN indicates that the TMS 9900 has disabled its output buffers to allow the memory to place memory-read data on the data bus during $\overline{\text{MEMEN}}$. DBIN remains low in all other cases except when HOLDA is active.
$\overline{\text{MEMEN}}$	63	OUT	Memory enable. When active (low), $\overline{\text{MEMEN}}$ indicates that the address bus contains a memory address.
$\overline{\text{WE}}$	61	OUT	Write enable. When active (low), $\overline{\text{WE}}$ indicates that memory-write data is available from the TMS 9900 to be written into memory.
CRUCLK	60	OUT	CRU clock. When active (high), CRUCLK indicates that external interface logic should sample the output data on CRUOUT or should decode external instructions on A0 through A2.
CRUIN	31	IN	CRU data in. CRUIN, normally driven by 3-state or open-collector devices, receives input data from external interface logic. When the processor executes a STCR or TB instruction,it samples CRUIN for the level of the CRU input bit specified by the address bus (A3 through A14).
CRUOUT	30	OUT	CRU data out. Serial I/O data appears on the CRUOUT line when an LDCR, SBZ, or SBO instruction is executed. The data on CRUOUT should be sampled by external I/O interface logic when CRUCLK goes active (high).
			INTERRUPT CONTROL
$\overline{\text{INTREQ}}$	32	IN	Interrupt request. When active (low), $\overline{\text{INTREQ}}$ indicates that an external interrupt is requested. If $\overline{\text{INTREQ}}$ is active, the processor loads the data on the interrupt-code-input lines IC0 through IC3 into the internal interrupt-code-storage register. The code is compared to the interrupt mask bits of the status register. If equal or higher priority than the enabled interrupt level (interrupt code equal or less than status register bits 12 through 15) the TMS 9900 interrupt sequence is initiated. If the comparison fails, the processor ignores the request. $\overline{\text{INTREQ}}$ should remain active and the processor will continue to sample IC0 through IC3 until the program enables a sufficiently low priority to accept the request interrupt.
IC0 (MSB) IC1 IC2 IC3 (LSB)	36 35 34 33	IN IN IN IN	Interrupt codes. IC0 is the MSB of the interrupt code, which is sampled when $\overline{\text{INTREQ}}$ is active. When IC0 through IC3 are LLLH, the highest external-priority interrupt is being requested and when HHHH, the lowest-priority interrupt is being requested.
			MEMORY CONTROL
$\overline{\text{HOLD}}$	64	IN	Hold. When active (low), $\overline{\text{HOLD}}$ indicates to the processor that an external controller (e.g., DMA device) desires to utilize the address and data buses to transfer data to or from memory. The TMS 9900 enters the hold state following a hold signal when it has completed its present memory cycle.* The processor then places the address and data buses in the high-impedance state (along with $\overline{\text{WE}}$, $\overline{\text{MEMEN}}$, and DBIN) and responds with a hold-acknowledge signal (HOLDA). When $\overline{\text{HOLD}}$ is removed, the processor returns to normal operation.
HOLDA	5	OUT	Hold acknowledge. When active (high), HOLDA indicates that the processor is in the hold state and the address and data buses and memory control outputs ($\overline{\text{WE}}$, $\overline{\text{MEMEN}}$, and DBIN) are in the high-impedance state.
READY	62	IN	Ready. When active (high), READY indicates that memory will be ready to read or write during the next clock cycle. When not-ready is indicated during a memory operation, the TMS 9900 enters a wait state and suspends internal operation until the memory systems indicate ready.
WAIT	3	OUT	Wait. When active (high), WAIT indicates that the TMS 9900 has entered a wait state because of a not-ready condition from memory.

*If the cycle following the present memory cycle is also a memory cycle, it, too, is completed before the TMS9900 enters the hold state. The maximum number of consecutive memory cycles is three.

TABLE 2 (CONCLUDED)

SIGNATURE	PIN	I/O	DESCRIPTION
			TIMING AND CONTROL
IAQ	7	OUT	Instruction acquisition. IAQ is active (high) during any memory cycle when the TMS 9900 is acquiring an instruction. IAQ can be used to detect illegal op codes.
$\overline{\text{LOAD}}$	4	IN	Load. When active (low), $\overline{\text{LOAD}}$ causes the TMS 9900 to execute a nonmaskable interrupt with memory address FFFC_{16} containing the trap vector (WP and PC). The load sequence begins after the instruction being executed is completed. $\overline{\text{LOAD}}$ will also terminate an idle state. If $\overline{\text{LOAD}}$ is active during the time $\overline{\text{RESET}}$ is released, then the $\overline{\text{LOAD}}$ trap will occur after the $\overline{\text{RESET}}$ function is completed. $\overline{\text{LOAD}}$ should remain active for one instruction period. IAQ can be used to determine instruction boundaries. This signal can be used to implement cold-start ROM loaders. Additionally, front-panel routines can be implemented using CRU bits as front-panel-interface signals and software-control routines to control the panel operations.
$\overline{\text{RESET}}$	6	IN	Reset. When active (low), $\overline{\text{RESET}}$ causes the processor to be reset and inhibits $\overline{\text{WE}}$ and CRUCLK. When $\overline{\text{RESET}}$ is released, the TMS 9900 then initiates a level-zero interrupt sequence that acquires WP and PC from locations 0000 and 0002, sets all status register bits to zero, and starts execution. $\overline{\text{RESET}}$ will also terminate an idle state. $\overline{\text{RESET}}$ must be held active for a minimum of three clock cycles.

2.9 TIMING

2.9.1 MEMORY

A basic memory read and write cycle is shown in Figure 9. The read cycle is shown with no wait states and the write cycle is shown with one wait state.

$\overline{\text{MEMEN}}$ goes active (low) during each memory cycle. At the same time that $\overline{\text{MEMEN}}$ is active, the memory address appears on the address bus bits A0 through A14. If the cycle is a memory-read cycle, DBIN will go active (high) at the same time $\overline{\text{MEMEN}}$ and A0 through A14 become valid. The memory-write signal $\overline{\text{WE}}$ will remain inactive (high) during a read cycle. If the read cycle is also an instruction acquisition cycle, IAQ will go active (high) during the cycle.

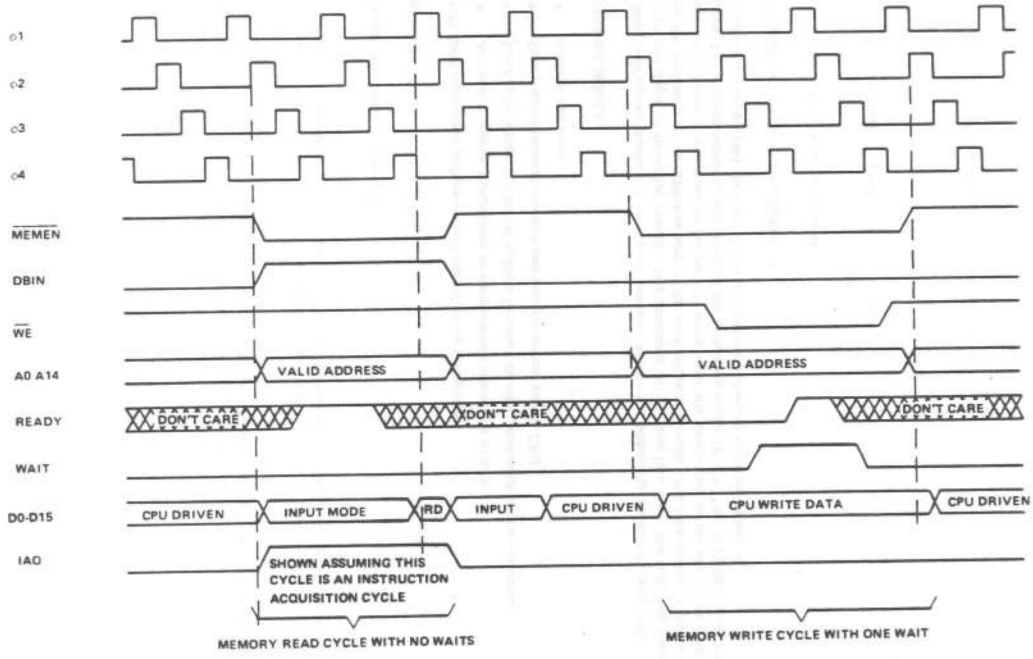
The READY signal, which allows extended memory cycles, is shown high during $\phi 1$ of the second clock cycle of the read operation. This indicates to the TMS 9900 that memory-read data will be valid during $\phi 1$ of the next clock cycle. If READY is low during $\phi 1$, then the TMS 9900 enters a wait state suspending internal operation until a READY is sensed during a subsequent $\phi 1$. The memory read data is then sampled by the TMS 9900 during the next $\phi 1$, which completes the memory-read cycle.

At the end of the read cycle, $\overline{\text{MEMEN}}$ and DBIN go inactive (high and low, respectively). The address bus may also change at this time, however, the data bus remains in the input mode for one clock cycle after the read cycle.

A write cycle is similar to the read cycle with the exception that $\overline{\text{WE}}$ goes active (low) as shown and valid write data appears on the data bus at the same time the address appears. The write cycle is shown as an example of a one-wait-state memory cycle. READY is low during $\phi 1$ resulting in the WAIT signal shown.

2.9.2 HOLD

Other interfaces may utilize the TMS 9900 memory bus by using the hold operation (illustrated in Figure 10) of the TMS 9900. When $\overline{\text{HOLD}}$ is active (low), the TMS 9900 enters the hold state at the next available non-memory cycle. Considering that there can be a maximum of three consecutive memory cycles, the maximum delay between $\overline{\text{HOLD}}$ going active to HOLDA going active (high) could be $t_{c(\phi)}$ (for setup) + (6 + 3W) $t_{c(\phi)}$ + $t_{c(\phi)}$ (delay for HOLDA), where W is the number of wait states per memory cycle and $t_{c(\phi)}$ is the clock cycle time. When the TMS 9900 has entered the hold state, HOLDA goes active (high) and A0 through A15, D0 through D15 DBIN, $\overline{\text{MEMEN}}$, and $\overline{\text{WE}}$ go into a high-impedance state to allow other devices to use the memory buses. When $\overline{\text{HOLD}}$ goes inactive (high), the TMS 9900 resumes processing as shown. If hold occurs during a CRU operation, the TMS 9900 uses an extra clock cycle (after the removal of the $\overline{\text{HOLD}}$ signal) to reassert the CRU address providing the normal setup times for the CRU bit transfer that was interrupted.



RD = READ DATA

FIGURE 9 – TMS 9900 MEMORY BUS TIMING

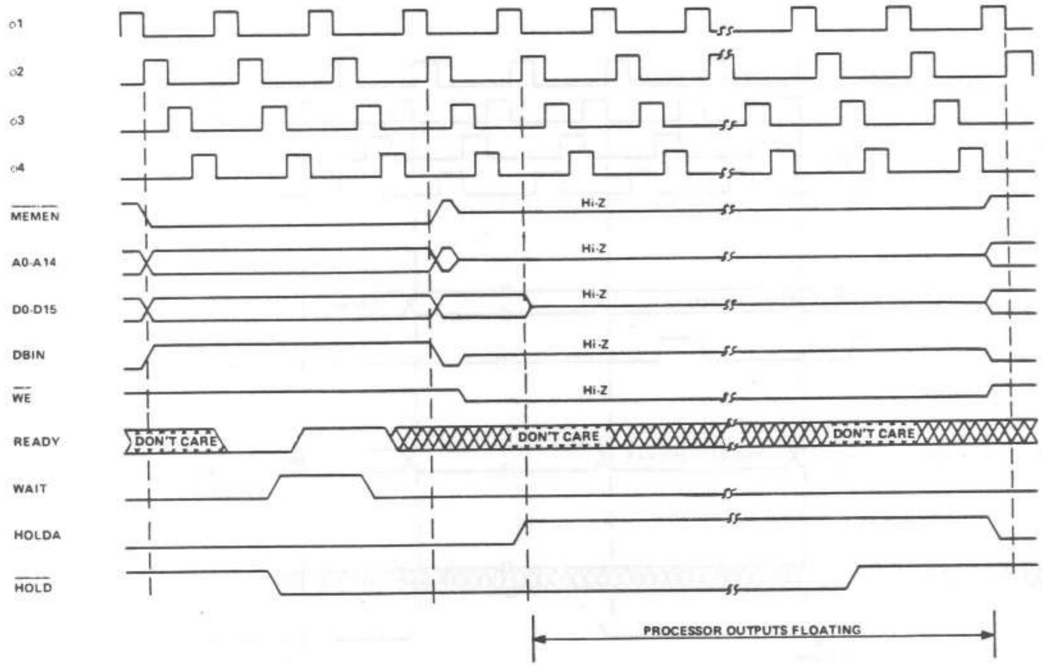


FIGURE 10 – TMS 9900 HOLD TIMING

2.9.3 CRU

CRU interface timing is shown in Figure 11. The timing for transferring two bits out and one bit in is shown. These transfers would occur during the execution of a CRU instruction. The other cycles of the instruction execution are not illustrated. To output a CRU bit, the CRU-bit address is placed on the address bus A0 through A14 and the actual bit data on CRUOUT. During the second clock cycle a CRU pulse is supplied by CRUCLK. This process is repeated until the number of bits specified by the instruction are completed.

The CRU input operation is similar in that the bit address appears on A0 through A14. During the subsequent cycle the TMS 9900 accepts the bit input data as shown. No CRUCLK pulses occur during a CRU input operation.

3. TMS 9900 INSTRUCTION SET

3.1 DEFINITION

Each TMS 9900 instruction performs one of the following operations:

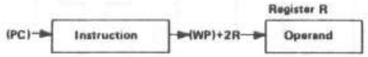
- Arithmetic, logical, comparison, or manipulation operations on data
- Loading or storage of internal registers (program counter, workspace pointer, or status)
- Data transfer between memory and external devices via the CRU
- Control functions.

3.2 ADDRESSING MODES

TMS 9900 instructions contain a variety of available modes for addressing random-memory data (e.g., program parameters and flags), or formatted memory data (character strings, data lists, etc.). The following figures graphically describe the derivation of the effective address for each addressing mode. The applicability of addressing modes to particular instructions is described in Section 3.5 along with the description of the operations performed by the instruction. The symbols following the names of the addressing modes [R, *R, *R+, @ LABEL, or @ TABLE (R)] are the general forms used by TMS 9900 assemblers to select the addressing mode for register R.

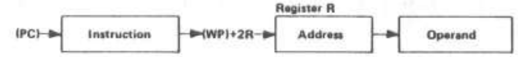
3.2.1 WORKSPACE REGISTER ADDRESSING R

Workspace Register R contains the operand.



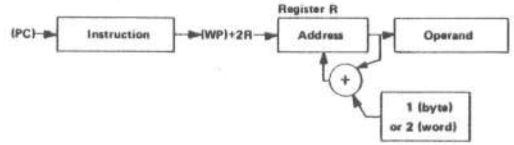
3.2.2 WORKSPACE REGISTER INDIRECT ADDRESSING *R

Workspace Register R contains the address of the operand.



3.2.3 WORKSPACE REGISTER INDIRECT AUTO INCREMENT ADDRESSING *R+

Workspace Register R contains the address of the operand. After acquiring the operand, the contents of workspace register R are incremented.



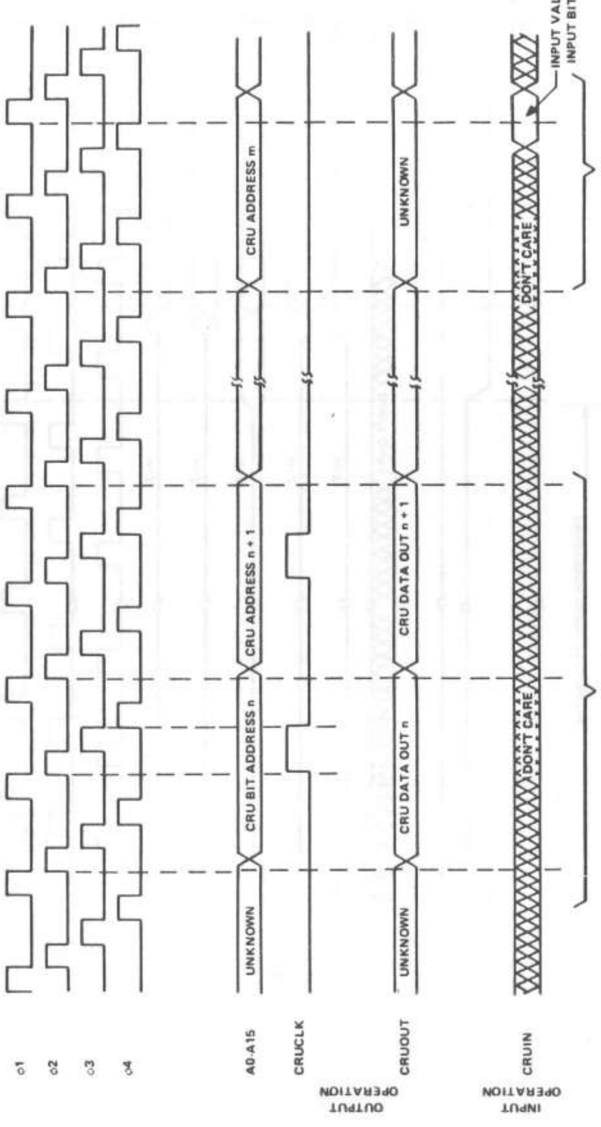
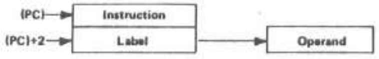


FIGURE 11 – TMS 9900 CRU INTERFACE TIMING

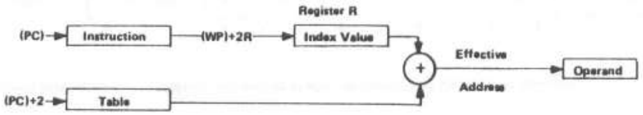
3.2.4 SYMBOLIC (DIRECT) ADDRESSING @ LABEL

The word following the instruction contains the address of the operand.



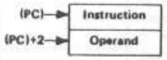
3.2.5 INDEXED ADDRESSING @ TABLE (R)

The word following the instruction contains the base address. Workspace register R contains the index value. The sum of the base address and the index value results in the effective address of the operand.



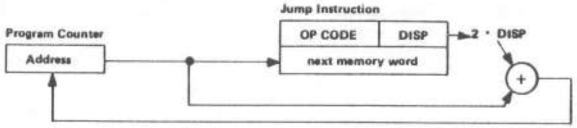
3.2.6 IMMEDIATE ADDRESSING

The word following the instruction contains the operand.



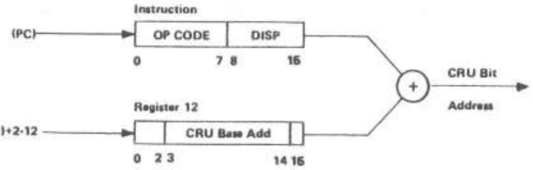
3.2.7 PROGRAM COUNTER RELATIVE ADDRESSING

The 8-bit signed displacement in the right byte (bits 8 through 15) of the instruction is multiplied by 2 and added to the updated contents of the program counter. The result is placed in the PC.



3.2.8 CRU RELATIVE ADDRESSING

The 8-bit signed displacement in the right byte of the instruction is added to the CRU base address (bits 3 through 14 of the workspace register 12). The result is the CRU address of the selected CRU bit.



3.3 TERMS AND DEFINITIONS

The following terms are used in describing the instructions of the TMS 9900:

TERM	DEFINITION
B	Byte Indicator (1=byte, 0 = word)
C	Bit count
D	Destination address register
DA	Destination address
IOP	Immediate operand
LSB(n)	Least significant (right most) bit of (n)
MSB(n)	Most significant (left most) bit of (n)
N	Don't care
PC	Program counter
Result	Result of operation performed by instruction
S	Source address register
SA	Source address
ST	Status register
STn	Bit n of status register
T_D	Destination address modifier
T_S	Source address modifier
W	Workspace register
WRn	Workspace register n
(n)	Contents of n
a → b	a is transferred to b
|n|	Absolute value of n
+	Arithmetic addition
–	Arithmetic subtraction
AND	Logical AND
OR	Logical OR
⊕	Logical exclusive OR
$\bar{n}$	Logical complement of n

3.4 STATUS REGISTER

The status register contains the interrupt mask level and information pertaining to the instruction operation.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
ST0	ST1	ST2	ST3	ST4	ST5	ST6	not used (=0)					ST12	ST13	ST14	ST15
L>	A>	=	C	O	P	X						Interrupt Mask			

BIT	NAME	INSTRUCTION	CONDITION TO SET BIT TO 1
ST0	LOGICAL GREATER THAN	C,CB	If MSB(SA) = 1 and MSB(DA) = 0, or if MSB(SA) = MSB(DA) and MSB of [(DA)-(SA)] = 1
		CI	If MSB(W) = 1 and MSB of IOP = 0, or if MSB(W) = MSB of IOP and MSB of [IOP-(W)] = 1
		ABS	If (SA) ≠ 0
		All Others	If result ≠ 0
ST1	ARITHMETIC GREATER THAN	C,CB	If MSB(SA) = 0 and MSB(DA) = 1, or if MSB(SA) = MSB(DA) and MSB of [(DA)-(SA)] = 1
		CI	If MSB(W) = 0 and MSB of IOP = 1, or if MSB(W) = MSB of IOP and MSB of [IOP-(W)] = 1
		ABS	If MSB(SA) = 0 and (SA) ≠ 0
		All Others	If MSB of result = 0 and result ≠ 0

– Continued

BIT	NAME	INSTRUCTION	CONDITION TO SET BIT TO 1
ST2	EQUAL	C, CB	If (SA) = (DA)
		C1	If (W) = IOP
		COC	If (SA) and $(\overline{DA})$ = 0
		CZC	If (SA) and (DA) = 0
		TB	If CRUIN = 1
		ABS	If (SA) = 0
		All others	If result = 0
ST3	CARRY	A, AB, ABS, AI, DEC, DECT, INC, INCT, NEG, S, SB	If CARRY OUT = 1
		SLA, SRA, SRC, SRL	If last bit shifted out = 1
ST4	OVERFLOW	A, AB	If MSB(SA) = MSB(DA) and MSB of result ≠ MSB(DA)
		AI	If MSB(W) = MSB of IOP and MSB of result ≠ MSB(W)
		S, SB	If MSB(SA) ≠ MSB(DA) and MSB of result ≠ MSB(DA)
		DEC, DECT	If MSB(SA) = 1 and MSB of result = 0
		INC, INCT	If MSB(SA) = 0 and MSB of result = 1
		SLA	If MSB changes during shift
		DIV	If MSB(SA) = 0 and MSB(DA) = 1, or if MSB(SA) = MSB(DA) and MSB of [(DA)-(SA)] = 0
		ABS, NEG	If (SA) = 8000_{16}
ST5	PARITY	CB, MOVB	If (SA) has odd number of 1's
		LDCR, STCR	If 1 < C < 8 and (SA) has odd number of 1's
		AB, SB, SOCB, SZCB	If result has odd number of 1's
ST6	XOP	XOP	If XOP instruction is executed
ST12–ST15	INTERRUPT MASK	LIMI	If corresponding bit of IOP is 1
		RTWP	If corresponding bit of WR15 is 1

3.5 INSTRUCTIONS

3.5.1 Dual Operand Instructions with Multiple Addressing Modes for Source and Destination Operand

General format:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
OP CODE			B	T_D		D				T_S		S			

If B = 1 the operands are bytes and the operand addresses are byte addresses. If B = 0 the operands are words and the operand addresses are word addresses.

The addressing mode for each operand is determined by the T field of that operand.

T_S OR T_D	S OR D	ADDRESSING MODE	NOTES
00	0, 1, . . . 15	Workspace register	1
01	0, 1, . . . 15	Workspace register indirect	
10	0	Symbolic	4
10	1, 2, . . . 15	Indexed	2,4
11	0, 1, . . . 15	Workspace register indirect auto-increment	3

NOTES: 1. When a workspace register is the operand of a byte instruction (bit 3 = 1), the left byte (bits 0 through 7) is the operand and the right byte (bits 8 through 15) is unchanged.
2. Workspace register 0 may not be used for indexing.
3. The workspace register is incremented by 1 for byte instructions (bit 3 = 1) and is incremented by 2 for word instructions (bit 3 = 0).
4. When T_S = T_D = 10, two words are required in addition to the instruction word. The first word is the source operand base address and the second word is the destination operand base address.

MNEMONIC	OP CODE			B	MEANING	RESULT COMPARED TO 0	STATUS BITS AFFECTED	DESCRIPTION
	0	1	2	3				
A	1	0	1	0	Add	Yes	0-4	(SA)+(DA) → (DA)
AB	1	0	1	1	Add bytes	Yes	0-5	(SA)+(DA) → (DA)
C	1	0	0	0	Compare	No	0-2	Compare (SA) to (DA) and set appropriate status bits
CB	1	0	0	1	Compare bytes	No	0-2,5	Compare (SA) to (DA) and set appropriate status bits
S	0	1	1	0	Subtract	Yes	0-4	(DA) – (SA) → (DA)
SB	0	1	1	1	Subtract bytes	Yes	0-5	(DA) – (SA) → (DA)
SOC	1	1	1	0	Set ones corresponding	Yes	0-2	(DA) OR (SA) → (DA)
SOCB	1	1	1	1	Set ones corresponding bytes	Yes	0-2,5	(DA) OR (SA) → (DA)
SZC	0	1	0	0	Set zeroes corresponding	Yes	0-2	(DA) AND $(\overline{SA})$ → (DA)
SZCB	0	1	0	1	Set zeroes corresponding bytes	Yes	0-2,5	(DA) AND $(\overline{SA})$ → (DA)
MOV	1	1	0	0	Move	Yes	0-2	(SA) → (DA)
MOVB	1	1	0	1	Move bytes	Yes	0-2,5	(SA) → (DA)

3.5.2 Dual Operand Instructions with Multiple Addressing Modes for the Source Operand and Workspace Register Addressing for the Destination

General format:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
OP CODE						D				T_S		S			

The addressing mode for the source operand is determined by the T_S field.

T_S	S	ADDRESSING MODE	NOTES
00	0, 1, . . . 15	Workspace register	
01	0, 1, . . . 15	Workspace register indirect	
10	0	Symbolic	
10	1, 2, . . . 15	Indexed	1
11	0, 1, . . . 15	Workspace register indirect auto increment	2

NOTES: 1. Workspace register 0 may not be used for indexing.
2. The workspace register is incremented by 2.

MNEMONIC	OP CODE 0 1 2 3 4 5	MEANING	RESULT COMPARED TO 0	STATUS BITS AFFECTED	DESCRIPTION
COC	0 0 1 0 0 0	Compare ones corresponding	No	2	Test (D) to determine if 1's are in each bit position where 1's are in (SA). If so, set ST2.
CZC	0 0 1 0 0 1	Compare zeros corresponding	No	2	Test (D) to determine if 0's are in each bit position where 1's are in (SA). If so, set ST2.
XOR	0 0 1 0 1 0	Exclusive OR	Yes	0-2	(D) ⊕ (SA) → (D)
MPY	0 0 1 1 1 0	Multiply	No		Multiply unsigned (D) by unsigned (SA) and place unsigned 32-bit product in D (most significant) and D+1 (least significant). If WR15 is D, the next word in memory after WR15 will be used for the least significant half of the product.
DIV	0 0 1 1 1 1	Divide	No	4	If unsigned (SA) is less than or equal to unsigned (D), perform no operation and set ST4. Otherwise, divide unsigned (D) and (D+1) by unsigned (SA). Quotient → (D), remainder → (D+1). If D = 15, the next word in memory after WR 15 will be used for the remainder.

3.5.3 Extended Operation (XOP) Instruction

General format:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	1	0	1	1	D				T_S		S			

The T_S and S fields provide multiple mode addressing capability for the source operand. When the XOP is executed, ST6 is set and the following transfers occur:

$(40_{16} + 4D) \rightarrow (WP)$
$(42_{16} + 4D) \rightarrow (PC)$
SA → (new WR11)
(old WP) → (new WR13)
(old PC) → (new WR14)
(old ST) → (new WR15)

The TMS 9900 does not test interrupt requests ($\overline{\text{INTREQ}}$) upon completion of the XOP instruction.

3.5.4 Single Operand Instructions

General format:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
OP CODE										T_S		S			

The T_S and S fields provide multiple mode addressing capability for the source operand.

MNEMONIC	OP CODE	MEANING	RESULT COMPARED TO 0	STATUS BITS AFFECTED	DESCRIPTION
	0 1 2 3 4 5 6 7 8 9				
B	0 0 0 0 0 1 0 0 0 1	Branch	No	–	SA → (PC)
BL	0 0 0 0 0 1 1 0 1 0	Branch and link	No	–	(PC) → (WR11); SA → (PC)
BLWP	0 0 0 0 0 1 0 0 0 0	Branch and load workspace pointer	No	–	(SA) → (WP); (SA+2) → (PC); (old WP) → (new WR13); (old PC) → (new WR14); (old ST) → (new WR15); the interrupt input ($\overline{\text{INTREQ}}$) is not tested upon completion of the BLWP instruction.
CLR	0 0 0 0 0 1 0 0 1 1	Clear operand	No	–	0 → (SA)
SETO	0 0 0 0 0 1 1 1 0 0	Set to ones	No	–	FFFF_{16} → (SA)
INV	0 0 0 0 0 1 0 1 0 1	Invert	Yes	0-2	$\overline{(SA)}$ → (SA)
NEG	0 0 0 0 0 1 0 1 0 0	Negate	Yes	0-4	–(SA) → (SA)
ABS	0 0 0 0 0 1 1 1 0 1	Absolute value*	No	0-4	|(SA)| → (SA)
SWPB	0 0 0 0 0 1 1 0 1 1	Swap bytes	No	–	(SA), bits 0 thru 7 → (SA), bits 8 thru 15; (SA), bits 8 thru 15 → (SA), bits 0 thru 7.
INC	0 0 0 0 0 1 0 1 1 0	Increment	Yes	0-4	(SA) + 1 → (SA)
INCT	0 0 0 0 0 1 0 1 1 1	Increment by two	Yes	0-4	(SA) + 2 → (SA)
DEC	0 0 0 0 0 1 1 0 0 0	Decrement	Yes	0-4	(SA) – 1 → (SA)
DECT	0 0 0 0 0 1 1 0 0 1	Decrement by two	Yes	0-4	(SA) – 2 → (SA)
X†	0 0 0 0 0 1 0 0 1 0	Execute	No	–	Execute the instruction at SA.

*Operand is compared to zero for status bit.

†If additional memory words for the execute instruction are required to define the operands of the instruction located at SA, these words will be accessed from PC and the PC will be updated accordingly. The instruction acquisition signal (IAQ) will not be true when the TMS 9900 accesses the instruction at SA. Status bits are affected in the normal manner for the instruction executed.

3.5.5 CRU Multiple-Bit Instructions

General format:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
OP CODE						C				T_S		S			

The C field specifies the number of bits to be transferred. If C = 0, 16 bits will be transferred. The CRU base register (WR12, bits 3 through 14) defines the starting CRU bit address. The bits are transferred serially and the CRU address is incremented with each bit transfer, although the contents of WR12 is not affected. T_S and S provide multiple mode addressing capability for the source operand. If 8 or fewer bits are transferred (C = 1 through 8), the source address is a byte address. If 9 or more bits are transferred (C = 0, 9 through 15), the source address is a word address. If the source is addressed in the workspace register indirect auto increment mode, the workspace register is incremented by 1 if C = 1 through 8, and is incremented by 2 otherwise.

MNEMONIC	OP CODE	MEANING	RESULT COMPARED TO 0	STATUS BITS AFFECTED	DESCRIPTION
	0 1 2 3 4 5				
LDCR	0 0 1 1 0 0	Load communcation register	Yes	0-2,5†	Beginning with LSB of (SA), transfer the specified number of bits from (SA) to the CRU.
STCR	0 0 1 1 0 1	Store communcation register	Yes	0-2,5†	Beginning with LSB of (SA), transfer the specified number of bits from the CRU to (SA). Load unfilled bit positions with 0.

†ST5 is affected only if 1 ≤ C ≤ 8.

3.5.6 CRU Single-Bit Instructions

General format:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
OP CODE								SIGNED DISPLACEMENT							

CRU relative addressing is used to address the selected CRU bit.

MNEMONIC	OP CODE	MEANING	STATUS BITS AFFECTED	DESCRIPTION
	0 1 2 3 4 5 6 7			
SBO	0 0 0 1 1 1 0 1	Set bit to one	–	Set the selected CRU output bit to 1.
SBZ	0 0 0 1 1 1 1 0	Set bit to zero	–	Set the selected CRU output bit to 0.
TB	0 0 0 1 1 1 1 1	Test bit	2	If the selected CRU input bit = 1, set ST2.

3.5.7 Jump Instructions

General format:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
OP CODE								DISPLACEMENT							

Jump instructions cause the PC to be loaded with the value selected by PC relative addressing if the bits of ST are at specified values. Otherwise, no operation occurs and the next instruction is executed since PC points to the next instruction. The displacement field is a word count to be added to PC. Thus, the jump instruction has a range of –128 to 127 words from memory-word address following the jump instruction. No ST bits are affected by jump instruction.

MNEMONIC	OP CODE 0 1 2 3 4 5 6 7	MEANING	ST CONDITION TO LOAD PC
JEQ	0 0 0 1 0 0 1 1	Jump equal	ST2 = 1
JGT	0 0 0 1 0 1 0 1	Jump greater than	ST1 = 1
JH	0 0 0 1 1 0 1 1	Jump high	ST0 = 1 and ST2 = 0
JHE	0 0 0 1 0 1 0 0	Jump high or equal	ST0 = 1 or ST2 = 1
JL	0 0 0 1 1 0 1 0	Jump low	ST0 = 0 and ST2 = 0
JLE	0 0 0 1 0 0 1 0	Jump low or equal	ST0 = 0 or ST2 = 1
JLT	0 0 0 1 0 0 0 1	Jump less than	ST1 = 0 and ST2 = 0
JMP	0 0 0 1 0 0 0 0	Jump unconditional	unconditional
JNC	0 0 0 1 0 1 1 1	Jump no carry	ST3 = 0
JNE	0 0 0 1 0 1 1 0	Jump not equal	ST2 = 0
JNO	0 0 0 1 1 0 0 1	Jump no overflow	ST4 = 0
JOC	0 0 0 1 1 0 0 0	Jump on carry	ST3 = 1
JOP	0 0 0 1 1 1 0 0	Jump odd parity	ST5 = 1

3.5.8 Shift Instructions

General format:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
OP CODE								C				W			

If C = 0, bits 12 through 15 of WR0 contain the shift count. If C = 0 and bits 12 through 15 of WR0 = 0, the shift count is 16.

MNEMONIC	OP CODE 0 1 2 3 4 5 6 7	MEANING	RESULT COMPARED TO 0	STATUS BITS AFFECTED	DESCRIPTION
SLA	0 0 0 0 1 0 1 0	Shift left arithmetic	Yes	0-4	Shift (W) left. Fill vacated bit positions with 0.
SRA	0 0 0 0 1 0 0 0	Shift right arithmetic	Yes	0-3	Shift (W) right. Fill vacated bit positions with original MSB of (W).
SRC	0 0 0 0 1 0 1 1	Shift right circular	Yes	0-3	Shift (W) right. Shift previous LSB into MSB.
SRL	0 0 0 0 1 0 0 1	Shift right logical	Yes	0-3	Shift (W) right. Fill vacated bit positions with 0's.

3.5.9 Immediate Register Instructions

General format:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
OP CODE											N	W			
IOP															

MNEMONIC	OP CODE 0 1 2 3 4 5 6 7 8 9 10	MEANING	RESULT COMPARED TO 0	STATUS BITS AFFECTED	DESCRIPTION
AI	0 0 0 0 0 0 1 0 0 0 1	Add immediate	Yes	0-4	(W) + IOP → (W)
ANDI	0 0 0 0 0 0 1 0 0 1 0	AND immediate	Yes	0-2	(W) AND IOP → (W).
CI	0 0 0 0 0 0 1 0 1 0 0	Compare immediate	Yes	0-2	Compare (W) to IOP and set appropriate status bits
LI	0 0 0 0 0 0 1 0 0 0 0	Load immediate	Yes	0-2	IOP → (W)
ORI	0 0 0 0 0 0 1 0 0 1 1	OR immediate	Yes	0-2	(W) OR IOP → (W)

3.5.10 Internal Register Load Immediate Instructions

General format:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
OP CODE											N				
IOP															

MNEMONIC	OP CODE 0 1 2 3 4 5 6 7 8 9 10	MEANING	DESCRIPTION
LWPI	0 0 0 0 0 0 1 0 1 1 1	Load workspace pointer immediate	IOP → (WP), no ST bits affected
LIMI	0 0 0 0 0 0 1 1 0 0 0	Load interrupt mask	IOP, bits 12 thru 15 → ST12 thru ST15

3.5.11 Internal Register Store Instructions

General format:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
OP CODE											N	W			

No ST bits are affected.

MNEMONIC	OP CODE 0 1 2 3 4 5 6 7 8 9 10	MEANING	DESCRIPTION
STST	0 0 0 0 0 0 1 0 1 1 0	Store status register	(ST) → (W)
STWP	0 0 0 0 0 0 1 0 1 0 1	Store workspace pointer	(WP) → (W)

3.5.12 Return Workspace Pointer (RTWP) Instruction

General format:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	0	0	0	0	1	1	1	0	0	N				

The RTWP instruction causes the following transfers to occur:

(WR15) → (ST)
(WR14) → (PC)
(WR13) → (WP)

3.5.13 External Instructions

General format:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
OP CODE											N				

External instructions cause the three most-significant address lines (A0 through A2) to be set to the below-described levels and the CRUCLK line to be pulsed, allowing external control functions to be initiated.

MNEMONIC	OP CODE 0 1 2 3 4 5 6 7 8 9 10	MEANING	STATUS BITS AFFECTED	DESCRIPTION	ADDRESS BUS A0	 A1	 A2
IDLE	0 0 0 0 0 0 1 1 0 1 0	Idle	–	Suspend TMS 9900 instruction execution until an interrupt, $\overline{\text{LOAD}}$, or $\overline{\text{RESET}}$ occurs	L	H	L
RSET	0 0 0 0 0 0 1 1 0 1 1	Reset	12–15	0 → ST12 thru ST15	L	H	H
CKOF	0 0 0 0 0 0 1 1 1 1 0	User defined		----	H	H	L
CKON	0 0 0 0 0 0 1 1 1 0 1	User defined		----	H	L	H
LREX	0 0 0 0 0 0 1 1 1 1 1	User defined		----	H	H	H

3.6 TMS 9900 INSTRUCTION EXECUTION TIMES

Instruction execution times for the TMS 9900 are a function of:

1) Clock cycle time, $t_{c(\phi)}$

2) Addressing mode used where operands have multiple addressing mode capability

3) . Number of wait states required per memory access.

Table 3 lists the number of clock cycles and memory accesses required to execute each TMS 9900 instruction. For instructions with multiple addressing modes for either or both operands, the table lists the number of clock cycles and memory accesses with all operands addressed in the workspace-register mode. To determine the additional number of clock cycles and memory accesses required for modified addressing, add the appropriate values from the referenced tables. The total instruction-execution time for an instruction is:

$T = t_{c(\phi)} \quad (C + W \cdot M)$

where:

T = total instruction execution time;

$t_{c(\phi)}$ = clock cycle time;

C = number of clock cycles for instruction execution plus address modification;

W = number of required wait states per memory access for instruction execution plus address modification;

M = number of memory accesses.

TABLE 3
INSTRUCTION EXECUTION TIMES

INSTRUCTION	CLOCK CYCLES C	MEMORY ACCESS M	ADDRESS MODIFICATION†	
			SOURCE	DEST
A	14	4	A	A
AB	14	4	B	B
ABS (MSB = 0)	12	2	A	–
(MSB = 1)	14	3	A	–
AI	14	4	–	–
ANDI	14	4	–	–
B	8	2	A	–
BL	12	3	A	–
BLWP	26	6	A	–
C	14	3	A	A
CB	14	3	B	B
CI	14	3	–	–
CKOF	12	1	–	–
CKON	12	1	–	–
CLR	10	3	A	–
COC	14	3	A	–
CZC	14	3	A	–
DEC	10	3	A	–
DECT	10	3	A	–
DIV (ST4 is set)	16	3	A	–
DIV (ST4 is reset)*	92-124	6	A	–
IDLE	12	1	–	–
INC	10	3	A	–
INCT	10	3	A	–
INV	10	3	A	–
Jump (PC is changed)	10	1	–	–
(PC is not changed)	8	1	–	–
LDCR (C = 0)	52	3	A	–
(1≤C≤8)	20+2C	3	B	–
(9≤C≤15)	20+2C	3	A	–
LI	12	3	–	–
LIMI	16	2	–	–
LREX	12	1	–	–
RESET function	26	5	–	–
LOAD function	22	5	–	–
Interrupt context switch	22	5	–	–
LWPI	10	2	–	–
MOV	14	4	A	A
MOVB	14	4	B	B
MPY	52	5	A	–
NEG	12	3	A	–
ORI	14	4	–	–
RSET	12	1	–	–
RTWP	14	4	–	–
S	14	4	A	A
SB	14	4	B	B
SBO	12	2	–	–
SBZ	12	2	–	–
SETO	10	3	A	–
Shift (C≠0)	12+2C	3	–	–
(C=0, Bits 12–15 of WR0=0)	52	4	–	–
(C=0, Bits 12–15 of WRP-N≠0)	20+2N	4	–	–
SOC	14	4	A	A
SOCB	14	4	B	B
STCR (C=0)	60	4	A	–
(1≤C≤7)	42	4	B	–
(C=8)	44	4	B	–
(9≤C≤15)	58	4	A	–
STST	8	2	–	–
STWP	8	2	–	–
SWPB	10	3	A	–
SZC	14	4	A	A
SZCB	14	4	B	B
TB	12	2	–	–
X **	8	2	A	–
XOP	36	8	A	–
XOR	14	4	A	–
Undefined op codes: 0000-01FF,0320-033F,0C00-0FFF,0780-07FF	6	1	–	–

*Execution time is dependent upon the partial quotient after each clock cycle during execution.

**Execution time is added to the execution time of the instruction located at the source address minus 4 clock cycles and 1 memory access time.

†The letters A and B refer to the respective tables that follow.

ADDRESS MODIFICATION – TABLE A

ADDRESSING MODE	CLOCK CYCLES C	MEMORY ACCESSES M
WR (T_S or T_D = 00)	0	0
WR indirect (T_S or T_D = 01)	4	1
WR indirect auto-increment (T_S or T_D = 11)	8	2
Symbolic (T_S or T_D = 10, S or D = 0)	8	1
Indexed (T_S or T_D = 10, S or D ≠ 0)	8	2

ADDRESS MODIFICATION – TABLE B

ADDRESSING MODE	CLOCK CYCLES C	MEMORY ACCESSES M
WR (T_S or T_D = 00)	0	0
WR indirect (T_S or T_D = 01)	4	1
WR indirect auto-increment (T_S or T_D = 11)	6	2
Symbolic (T_S or T_D = 10, S or D = 0)	8	1
Indexed (T_S or T_D = 10, S or D ≠ 0)	8	2

As an example, the instruction MOVB is used in a system with $t_{c(\phi)}$ = 0.333 μs and no wait states are required to access memory. Both operands are addressed in the workspace register mode:

$T = t_{c(\phi)} \; (C + W \cdot M) = 0.333 \; (14 + 0 \cdot 4) \; \mu s = 4.662 \; \mu s.$

If two wait states per memory access were required, the execution time is:

$T = 0.333 \; (14 + 2 \cdot 4) \; \mu s = 7.326 \; \mu s.$

If the source operand was addressed in the symbolic mode and two wait states were required:

$T = t_{c(\phi)} \; (C + W \cdot M)$

$C = 14 + 8 = 22$

$M = 4 + 1 = 5$

$T = 0.333 \; (22 + 2 \cdot 5) \; \mu s = 10.656 \; \mu s.$

4. TMS 9900 ELECTRICAL AND MECHANICAL SPECIFICATIONS

4.1 ABSOLUTE MAXIMUM RATINGS OVER OPERATING FREE-AIR TEMPERATURE RANGE (UNLESS OTHERWISE NOTED)*

Supply voltage, V_{CC} (see Note 1)	–0.3 to 20 V
Supply voltage, V_{DD} (see Note 1)	–0.3 to 20 V
Supply voltage, V_{SS} (see Note 1)	–0.3 to 20 V
All input voltages (see Note 1)	–0.3 to 20 V
Output voltage (with respect to V_{SS})	–2 V to 7 V
Continuous power dissipation	1.2 W
Operating free-air temperature range	0°C to 70°C
Storage temperature range	–55°C to 150°C

*Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions beyond those indicated in the "Recommended Operating Conditions" section of this specification is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

NOTE 1: Under absolute maximum ratings voltage values are with respect to the most negative supply, V_{BB} (substrate), unless otherwise noted. Throughout the remainder of this section, voltage values are with respect to V_{SS}.

4.2 RECOMMENDED OPERATING CONDITIONS

	MIN	NOM	MAX	UNIT
Supply voltage, V_{BB}	−5.25	−5	−4.75	V
Supply voltage, V_{CC}	4.75	5	5.25	V
Supply voltage, V_{DD}	11.4	12	12.6	V
Supply voltage, V_{SS}		0		V
High-level input voltage, V_{IH} (all inputs except clocks)	2.2	2.4	V_{CC}+1	V
High-level clock input voltage, $V_{IH(\phi)}$	V_{DD}−2		V_{DD}	V
Low-level input voltage, V_{IL} (all inputs except clocks)	−1	0.4	0.8	V
Low-level clock input voltage, $V_{IL(\phi)}$	−0.3	0.3	0.6	V
Operating free-air temperature, T_A	0		70	°C

4.3 ELECTRICAL CHARACTERISTICS OVER FULL RANGE OF RECOMMENDED OPERATING CONDITIONS (UNLESS OTHERWISE NOTED)

PARAMETER			TEST CONDITIONS	MIN	TYP[†]	MAX	UNIT
I_I	Input current	Data bus during DBIN	$V_I = V_{SS}$ to V_{CC}		±50	±100	μA
		WE, MEMEN, DBIN, Address bus, Data bus during HOLDA	$V_I = V_{SS}$ to V_{CC}		±50	±100	
		Clock*	V_I = −0.3 to 12.6 V		±25	±75	
		Any other inputs	$V_I = V_{SS}$ to V_{CC}		±1	±10	
V_{OH}	High-level output voltage		I_O = −0.4 mA	2.4		V_{CC}	V
V_{OL}	Low-level output voltage		I_O = 3.2 mA			0.65	V
			I_O = 2 mA			0.50	
I_{BB}	Supply current from V_{BB}				0.1	1	mA
I_{CC}	Supply current from V_{CC}				50	75	mA
I_{DD}	Supply current from V_{DD}				25	45	mA
C_i	Input capacitance (any inputs except clock and data bus)		V_{BB} = −5, f = 1MHz, unmeasured pins at V_{SS}		10	15	pF
$C_{i(\phi1)}$	Clock-1 input capacitance		V_{BB} = −5, f = 1MHz, unmeasured pins at V_{SS}		100	150	pF
$C_{i(\phi2)}$	Clock-2 input capacitance		V_{BB} = −5, f = 1MHz, unmeasured pins at V_{SS}		150	200	pF
$C_{i(\phi3)}$	Clock-3 input capacitance		V_{BB} = −5, f = 1MHz, unmeasured pins at V_{SS}		100	150	pF
$C_{i(\phi4)}$	Clock-4 input capacitance		V_{BB} = −5, f = 1MHz, unmeasured pins at V_{SS}		100	150	pF
C_{DB}	Data bus capacitance		V_{BB} = −5, f = 1MHz, unmeasured pins at V_{SS}		15	25	pF
C_o	Output capacitance (any output except data bus)		V_{BB} = −5, f = 1MHz, unmeasured pins at V_{SS}		10	15	pF

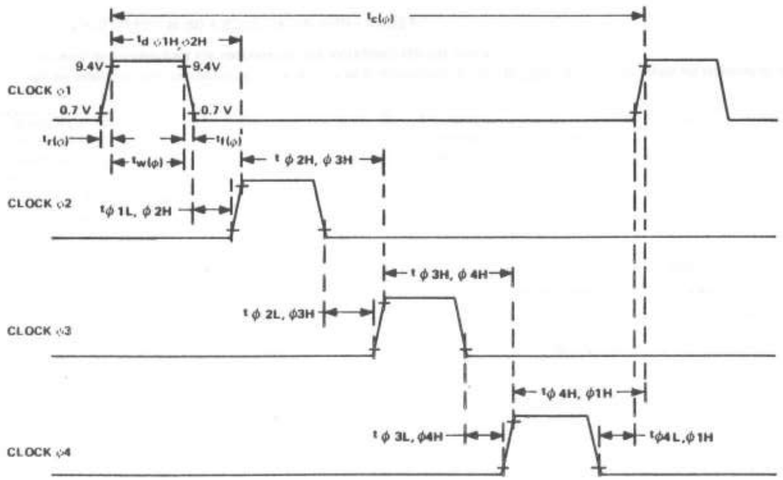
[†] All typical values are at T_A = 25°C and nominal voltages.
*D.C. Component of Operating Clock

4.4 TIMING REQUIREMENTS OVER FULL RANGE OF RECOMMENDED OPERATING CONDITIONS (SEE FIGURES 12 AND 13)

PARAMETER		MIN	NOM	MAX	UNIT
$t_{c(\phi)}$	Clock cycle time	0.3	0.333	0.5	μs
$t_{r(\phi)}$	Clock rise time	10	12		ns
$t_{f(\phi)}$	Clock fall time	10	12		ns
$t_{w(\phi)}$	Pulse width, any clock high	40	45	100	ns
$t_{\phi1L,\phi2H}$	Delay time, clock 1 low to clock 2 high (time between clock pulses)	0	5		ns
$t_{\phi2L,\phi3H}$	Delay time, clock 2 low to clock 3 high (time between clock pulses)	0	5		ns
$t_{\phi3L,\phi4H}$	Delay time, clock 3 low to clock 4 high (time between clock pulses)	0	5		ns
$t_{\phi4L,\phi1H}$	Delay time, clock 4 low to clock 1 high (time between clock pulses)	0	5		ns
$t_{\phi1H,\phi2H}$	Delay time, clock 1 high to clock 2 high (time between leading edges)	70	80		ns
$t_{\phi2H,\phi3H}$	Delay time, clock 2 high to clock 3 high (time between leading edges)	70	80		ns
$t_{\phi3H,\phi4H}$	Delay time, clock 3 high to clock 4 high (time between leading edges)	70	80		ns
$t_{\phi4H,\phi1H}$	Delay time, clock 4 high to clock 1 high (time between leading edges)	70	80		ns
t_{su}	Data or control setup time before clock 1	30			ns
t_h	Data hold time after clock 1	10			ns

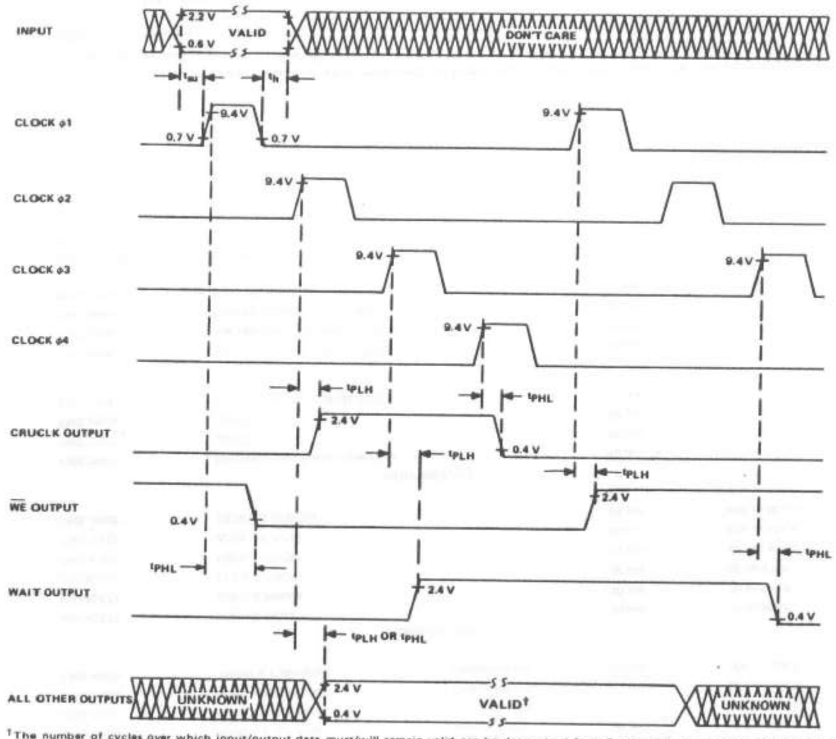
4.5 SWITCHING CHARACTERISTICS OVER FULL RANGE OF RECOMMENDED OPERATING CONDITIONS (SEE FIGURE 13)

PARAMETER		TEST CONDITIONS	MIN	TYP	MAX	UNIT
t_{PLH} or t_{PHL}	Propagation delay time, clocks to outputs	C_L = 200 pF		20	40	ns



NOTE: All timing and voltage levels shown on φ1 applies to φ2, φ3, and φ4 in the same manner.

FIGURE 12 – CLOCK TIMING



†The number of cycles over which input/output data must/will remain valid can be determined from Section 3.9. Note that in all cases data should not change during φ1.

FIGURE 13–SIGNAL TIMING

5. TMS 9900 PROTOTYPING SYSTEM

5.1 HARDWARE

The TMS 9900 prototyping system enables the user to generate and debug software and to debug I/O controller interfaces. The prototyping system consists of:

- 990/4 computer with TMS 9900 microprocessor
- 1024 bytes of ROM containing the bootstrap loader for loading prototyping system software, the front-panel and maintenance utility, and the CPU self-testing feature
- 16,896 bytes of RAM with provisions for expansion up to 57,334 bytes of RAM
- Programmable-write-protect feature for RAM
- Interface for Texas Instruments Model 733 ASR* Electronic Data Terminal with provisions for up to five additional interface moculdes

*Requires remote device control and 1200 baud EIA interface option on 733 ASR.

- Available with Texas Instruments Model 733 ASR Electronic Data Terminal
- 7-inch-high table-top chassis
- Programmer's front panel with controls for run, halt, single-instruction execute, and entering and displaying memory or register contents
- Power supply with the following voltages:
  5 V dc @ 20 A
  12 V dc @ 2 A
  −12 V dc @ 1 A
  −5 V dc @ 0.1 A
- Complete hardware and software documentation.

5.2 SYSTEM CONSOLE

The system console for the prototyping system is the 733 ASR, which provides keyboard entry, 30-character-per-second thermal printer, and dual cassette drives for program loading and storage.

5.3 SOFTWARE

The following software is provided on cassette for loading into the prototyping system:

- Debug Monitor – Provides full control of the prototyping system during program development and includes single instruction, multiple breakpoints, and entry and display capability for register and memory contents for debugging user software under 733 ASR console control.
- One-Pass Assembler – Converts source code stored on cassette to relocatable object on cassette and generates program listing. (Object is upward compatible with other 990 series assemblers).
- Linking Loader – Allows loading of absolute and relocatable object modules and links object modules as they are loaded.
- Source Editor – Enables user modification of both source and object from cassette with resultant storage on cassette.
- Trace Routine – Allows user to monitor status of computer at completion of each instruction.
- PROM Programming/Documentation Facility – Provides documentation for ROM mask generation, or communicates directly with the optional PROM Programmer Unit.

5.4 OPTIONS

The following optional equipment is offered for the prototyping system:

- Battery-pack/standby-power supply
- PROM programming unit and adapter boards
- Universal wire-wrap modules
- Expansion RAM modules
- Expansion EPROM modules
- I/O modules and other interfaces
- Rack-mounted version
- International ac voltage option

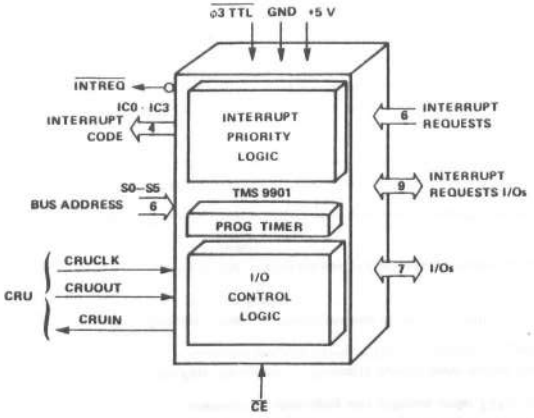
6. TMS 9900 SUPPORT CIRCUITS

MEMORY DEVICE	ORGANIZATION/FUNCTION	I/O STRUCTURE	PACKAGE	ACCESS TIME
		RAMS		
TMS 4036-2	64 x 8 static	Common bus	20 pin	450 ns MAX
TMS 4033	1024 x 1 static	Dedicated bus	16 pin	450 ns MAX
TMS 4039-2	256 x 4 static	Dedicated bus	22 pin	450 ns MAX
TMS 4042-2	256 x 4 static	Common bus	18 pin	450 ns MAX
TMS 4043-2	256 x 4 static	Common bus	16 pin	450 ns MAX
TMS 4050	4096 x 1 dynamic	Common bus	18 pin	300 ns MAX
TMS 4051	4096 x 1 dynamic	Dedicated bus	18 pin	300 ns MAX
TMS 4060	4096 x 1 dynamic	Dedicated bus	22 pin	300 ns MAX
TMS 4070	16384 x 1 dynamic	Dedicated bus	16 pin	300 ns MAX
		ROMS/PROMS		
SN74S371	256 x 8 ROM		20 pin	70 ns MAX
SN74S471	256 x 8 PROM		20 pin	70 ns MAX
SN74S472	512 x 8 PROM		20 pin	55 ns TYP
TMS 4700	1024 x 8 ROM		24 pin	450 ns MAX
TMS 4732	4096 x 8 ROM		24 pin	450 ns MAX
TMS 4908	1024 x 8 EPROM		24 pin	450 ns MAX
		PERIPHERALS		
TMS 9901	Programmable System Interface		40 pin	
TMS 9902	UART		18 pin	
TMS 9903	USRT		20 pin	
TIM 9904	Clock Generator (SN74LS362)		20 pin	
TIM 9905	Data multiplexer (SN74LS251)		16 pin	
TIM 9906	Addressable latch (SN74LS259)		16 pin	
TIM 9907	Priority encoder (SN74148)		16 pin	
SN74S412	8-bit I/O port		24 pin	
SN74LS138	3 to 8 Decoder		16 pin	
TMS 6011	UART		40 pin	
SN74S241	Bidirectional bus driver		20 pin	

7. SYSTEM DESIGN EXAMPLES

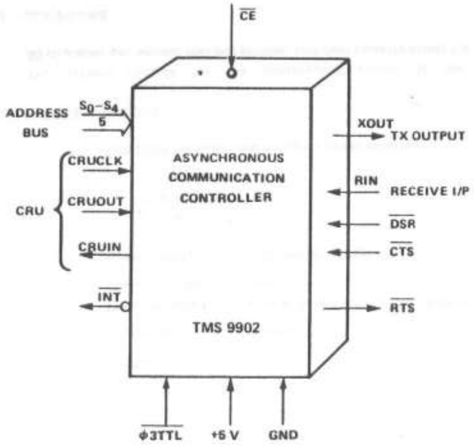
Figure 18 illustrates a typical minimum TMS 9900 system. Eight bits of input and output interface are implemented. The memory system contains 1024 x 16 ROM and 256 x 16 RAM memory blocks. The total package count for this system is 13 packages.

A maximum TMS 9900 microprocessor system is illustrated in Figure 19. ROM and RAM are both shown for a total of 65,536 bytes of memory. The I/O interface supports 4096-output bits and 4096-input bits. Fifteen external interrupts are implemented in the interrupt interface. The clock generator and control section contains memory decode logic, synchronization logic, and the clock electronics. Bus buffers, required for this maximally configured system, are indicated on the system buses.



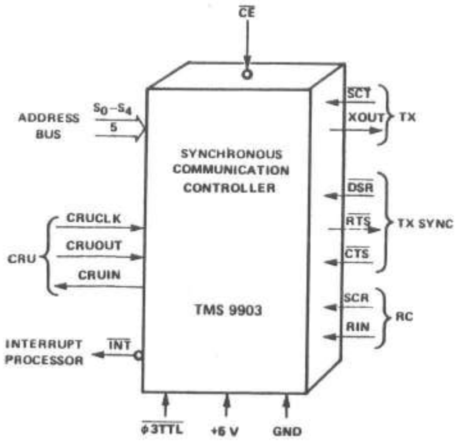
- TMS 9900 CPU PERIPHERAL
- 22 INTERRUPT AND I/O PORTS
  - 6 DEDICATED INTERRUPT INPUTS
  - 7 DEDICATED I/O PORTS
  - 9 PROGRAMMABLE AS INTERRUPT OR I/O
  - INT 3 PROGRAMMABLE AS TIMER INTERRUPT FOR INTERVALS FROM 21 μs TO 349 ms
- EXPANDABLE FOR INTERRUPT AND I/O EXPANSION
- 5 V POWER SUPPLY
- N-CHANNEL PROCESS

FIGURE 14 – TMS 9901 PROGRAMMABLE INTERRUPT AND I/O CONTROLLER



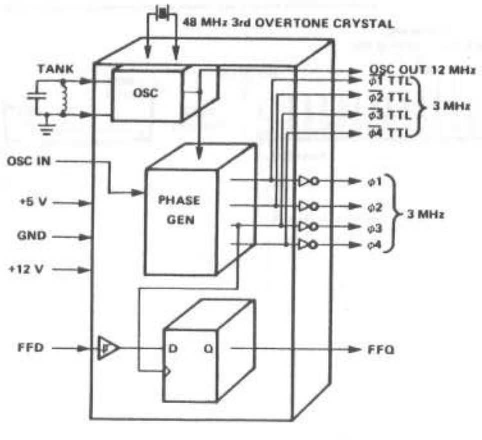
- TMS 9900 CPU PERIPHERAL
- PROGRAMMABLE DATA RATES 110 TO 76,800 BAUD
- PROGRAMMABLE CHARACTER LENGTH
  - 5–8 BITS
  - 1–1½–2 STOP BITS
  - ODD-EVEN-NO PARITY
- ON-CHIP INTERVAL TIMER 64 μs TO 16,384 μs
- SINGLE 5 V SUPPLY
- N-CHANNEL SILICON-GATE PROCESS
- 18 PIN 0.3" DIP

FIGURE 15 – TMS 9902 ASYNCHRONOUS COMMUNICATIONS CONTROLLER



- TMS 9900 CRU PERIPHERAL
- DC TO 250 K BITS/SEC
- PROGRAMMABLE SYNC. REGISTER AND CHARACTER LENGTH
- BI-SYNC AND SDLC COMPATIBLE
- ON-CHIP INTERVAL TIMER 64 μs TO 16,384 μs
- SINGLE 5 V SUPPLY
- N-CHANNEL SILICON-GATE PROCESS
- 20 PIN 0.3" DIP

FIGURE 16 – TMS 9903 SYNCHRONOUS COMMUNICATIONS CONTROLLER



- SINGLE-CHIP OSCILLATOR AND CLOCK DRIVER
- CRYSTAL CONTROLLED
- LOW-POWER SCHOTTKY PROCESS
- 20 PIN DIP
- NON-OVERLAPPING 12 V 4φ CLOCK
- NON-OVERLAPPING 5 V 4φ CLOCK
- SYNCHRONIZED RESET CIRCUIT
- +12, +5 V POWER SUPPLIES

FIGURE 17 - TIM 9904 (74LS362) CLOCK DRIVER

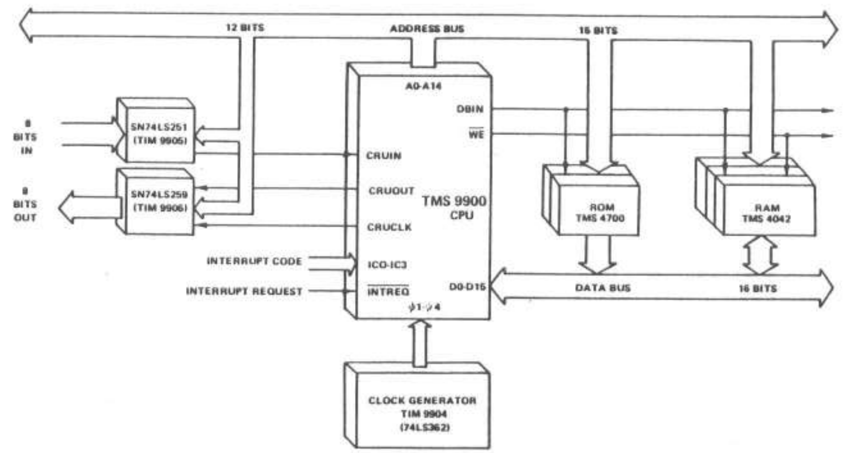


FIGURE 18 - MINIMUM TMS 9900 SYSTEM

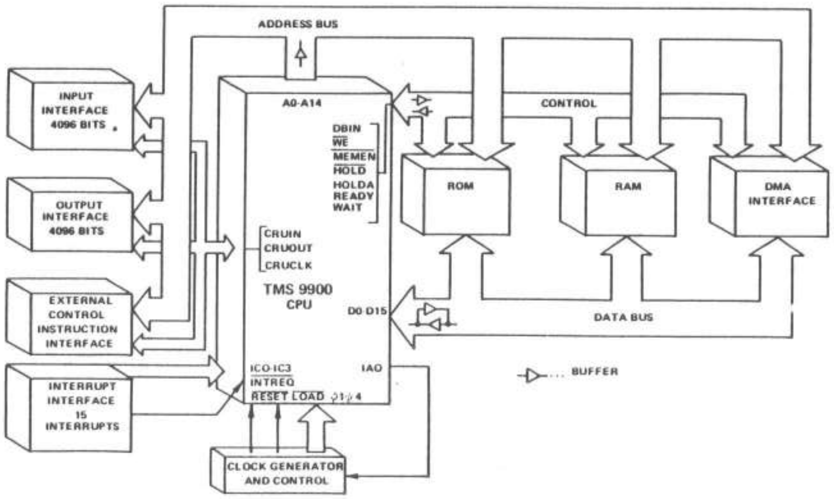
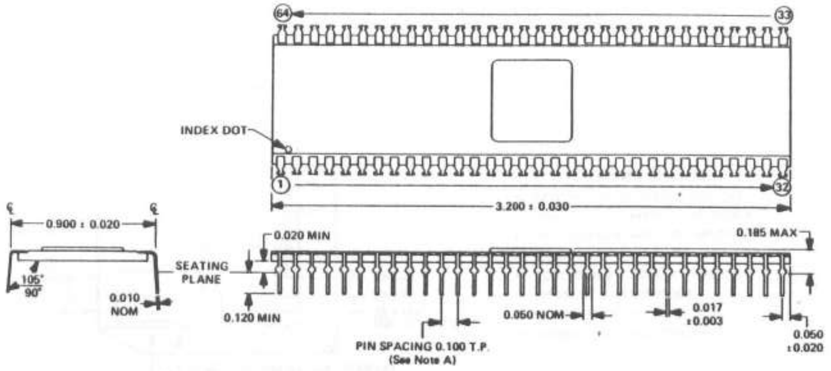


FIGURE 19 - MAXIMUM TMS 9900 SYSTEM

8. MECHANICAL DATA



NOTE A. Each pin centerline is located within 0.010 of its true longitudinal position.