National Semiconductor

PRELIMINARY

NS405-Series Display/Terminal Management Processor (TMP)

General Description

The NS405 is a CRT terminal controller on a chip. It is a microcomputer system which replaces the following LSI circuits commonly found in a CRT data terminal:

- Microcomputer
- CRT Controller
- DMA Controller
- Character Generator
- UART
- Baud Rate Generator
- Interrupt Controller
- Parallel I/O Controller
- Timer

In addition the NS405 includes powerful attribute logic, two graphics display modes, and fast video output circuits.

The NS405 is primarily intended for use in low-cost terminals, but contains many features which make it a superior building block for "smart" terminals and word processing systems.

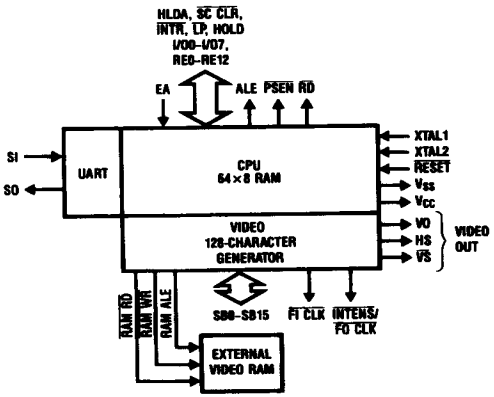
The NS405 interfaces easily to the display monitor, keyboard, display memory, and I/O ports. The architecture and instruction set are derived from the 8048-series microcontrollers. The instruction set has been enhanced and the architecture tailored to allow the NS405 CPU to efficiently manage a large display memory and an extensive interrupt environment.

The TMP can be used to easily and inexpensively add a display to many systems where it was previously impractical, it is not limited to terminal applications.

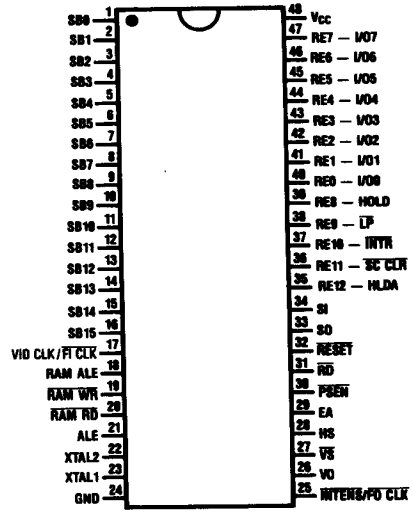
Features

- Enhanced 8048 instruction set and architecture
- Up to 8k x 8 ROM external with ROM expand bus
- On-board RAM 64 x 8
- Programmable display format
- On-board video memory management unit
- 16-bit bidirectional display memory bus (direct video and attribute RAM interface)
- Built-in timer
- Real-time clock (may be programmed for 1 Hz)
- Video control signals
- Eight independent attributes
- Pixel and block graphics display modes
- Programmable cursor characteristics
- Programmable CRT refresh rate
- Light pen feature
- UART, programmable baud rate up to 19.2k baud
- Character generator (128 characters 7 x 11 max)
- Single 5-volt supply @ 110 mA (typ)
- Up to 18 MHz video dot rate (12 MHz CPU clock)
- 48-pin package
- 8-bit parallel I/O port (multiplexed with external ROM)
- Extensive I/O expansion capabilities
- Up to 64k by 8 or 16 video RAM

Block and Connection Diagrams



TL/DD/5526–1



Top View

TL/DD/5526–2

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, contact the National Semiconductor Sales Office/ Distributors for availability and specifications.

Temperature Under Bias	0°C to +70°C
Storage Temperature	−65°C to +150°C
All Input or Output Voltages with Respect to V_{SS}*	−0.5V to +7.0V
Power Dissipation	1.5W
ESD	2000V

*EA, SI and VSYNC may be subjected to V_{SS} + 15V.

Note: *Absolute maximum ratings indicate limits beyond which permanent damage may occur. Continuous operation at these limits is not intended; operations should be limited to those conditions specified under DC Electrical Characteristics.*

DC Electrical Characteristics

T_A = 0°C to +70°C, V_{CC} = +5V ±10%, V_{SS} = 0V, unless otherwise specified

Symbol	Parameter	Test Conditions	Min	Max	Units
V_{IL1}	Input Low Voltage (All Except XTAL1, XTAL2, $\overline{RESET}$)		−0.5	0.8	V
V_{IH1}	Input High Voltage (All Except XTAL1, XTAL2, $\overline{RESET}$)		2.0	V_{CC}	V
V_{IL2}	Input Low Voltage (XTAL1, XTAL2, $\overline{RESET}$)		−0.5	0.6	V
V_{IH2}	Input High Voltage (XTAL1, XTAL2, $\overline{RESET}$)		3.8	V_{CC}	V
V_{OL}	Output Low Voltage (All Except $\overline{INTENS}$, VO)	I_{OL} = 2.0 mA		0.4	V
V_{OH}	Output High Voltage (All Except $\overline{INTENS}$, VO)	I_{OH} = −125 μA	2.4	V_{CC}	V
V_{OL}	Output Low Voltage ($\overline{INTENS}$, VO)	I_{OL} = 5.0 mA		0.4	V
V_{OH}	Output High Voltage ($\overline{INTENS}$, VO)	I_{OH} = −500 μA	2.4		V
I_{IL}	Input Leakage Current (EA, $\overline{INT}$, SI)	$V_{SS} \le V_{IN} \le V_{CC}$		±10	μA
I_{OL}	Output Leakage Current (ROM Expand Bus, High Impedance State)	$V_{CC} \ge V_{IN} \ge V_{SS}$ + 0.45		±10	μA
I_{OL}	Output Leakage Current (System Bus, High Impedance State)	$V_{CC} \ge V_{IN} \ge V_{SS}$ + 0.45		±100	μA
I_{CC}	Total Supply Current	T_A = 25°C		150	mA

AC Electrical Characteristics

T_A = 0°C to +70°C, V_{CC} = +5V ±10%, V_{SS} = 0V, unless otherwise specified

Symbol	Parameter	Min	Max	Units
CPU AND ROM EXPAND BUS TIMING				
F_{XTAL}	Crystal Frequency	3	18	MHz
F_{CPU}	CPU Frequency	3	12	MHz
t_{CY}	CPU Cycle Time	1.25	7.5	μs
t_{DF}	Video Dot Time	55.5	333.3	ns
t_{LL}	ALE Pulse Width (Note 1)	125		ns
t_{AL}	Address Setup to ALE (Note 1)	55		ns
t_{LA}	Address Hold from ALE (Note 1)	40		ns
t_{CC}	Control Pulse Width $\overline{PSEN}$, $\overline{RD}$ (Note 1)	250		ns
t_{DR}	Data Hold (Notes 1, 4)	0	100	ns
t_{RD}	$\overline{PSEN}$, $\overline{RD}$ to Data In (Note 1)		220	ns
t_{AD}	Address Setup to Data In (Note 1)		360	ns
t_{AFC}	Address Float to $\overline{RD}$, $\overline{PSEN}$ (Notes 1, 5)	0		ns
t_{CAF}	$\overline{PSEN}$ to Address Float (Notes 1, 5)	−10	+10	ns
t_{DAL}	Data Setup to ALE (RE0–7, 11, 12) (Note 1)	55		ns
t_{ALD}	Data Hold from ALE (RE0–7, 11, 12) (Note 1)	40		ns
t_{CIS}	Control Input Setup to ALE (RE8, 9, 10) (Note 1)	240		ns
t_{CIH}	Control Input Hold from ALE (RE8, 9, 10) (Notes 1, 4)	75	125	ns

AC Electrical Characteristics

T_A = 0°C to +70°C, V_{CC} = +5V ±10%, V_{SS} = 0V, unless otherwise specified (Continued)

Symbol	Parameter	Min	Max	Units
SYSTEM BUS TIMING				
t_{EL}	RAM ALE Low Time (Note 1)	250		ns
t_{EH}	RAM ALE High Time (Note 1)	100		ns
t_{AS}	Address Setup to RAM ALE (Note 1)	20		ns
t_{AH}	Address Hold from RAM ALE (Note 1)	10		ns
t_{RR}	$\overline{\text{RAM RD}}$ Width (Note 1)	210		ns
t_{AR}	Address Setup to $\overline{\text{RAM RD}}$ (Note 1)	80		ns
t_{RRD}	Data Access from $\overline{\text{RAM RD}}$ (Note 1)		140	ns
t_{RDR}	Data Hold from $\overline{\text{RAM RD}}$ (Notes 1, 4)	0	60	ns
t_{WFI}	$\overline{\text{FIFO In Clock}}$ Width (Note 1)	210		ns
t_{WW}	$\overline{\text{RAM WR}}$ Strobe Width (Note 1)	130		ns
t_{AW}	Address Setup to $\overline{\text{RAM WR}}$ (Note 1)	120		ns
t_{DW}	Data Setup to $\overline{\text{RAM WR}}$ (Note 1)	10		ns
t_{WD}	Data Hold from $\overline{\text{RAM WR}}$ (Note 1)	20		ns
VIDEO TIMING				
t_{DF}	Dot Period = $\frac{1}{f_c}$ (Note 1)	55		ns
t_{VID}	Video Blank Time (Note 1)	5	15	ns
t_{VI}	Skew, $\overline{\text{Intensity}}$ to Dot 0 (Note 1)	−15	15	ns
t_{FOV}	$\overline{\text{FIFO Out Clock}}$ to Dot 0 (Note 1)		15	ns
t_{WFOH}	$\overline{\text{FIFO Out Clock}}$ Width High (Note 1, Note 2)	55	165	ns

*1/3 CPU cycle.

**1 Dot time is 55 ns.

Note 1: Control outputs C_L = 80 pF; ROM Expand Bus outputs C_L = 150 pF; System Bus outputs C_L = 100 pF; V_{OUT} & $\overline{\text{INTENS}}$ outputs C_L = 50 pF; F_{XTAL} = 18 MHz; F_{CPU} = 12 MHz. XTAL1 & XTAL2 driven externally per *Figure 12b* with 50% duty cycle.

Note 2: $\overline{\text{FO CLK}}$ duty cycle is shown above.

Note 3: Hold request is latched. It is honored at the start of the next vertical retrace.

Note 4: Max spec. listed for user information only, to prevent bus contention. Maximum value not tested.

Note 5: Not tested.

Character Cell Width	$\overline{\text{FIFO Out}}$ HIGH	$\overline{\text{FIFO Out}}$ LOW
6	1 dot	5 dots
7	2 dots	5 dots
8	2 dots	6 dots
9	3 dots	6 dots
10	3 dots	7 dots

Input Hold Times

T_A = 25°C, V_{CC} = +5V ±10%, V_{SS} = 0V

Input	Min Active Time
Reset	50 ms (power up) 5 CPU Cycles (after power up)
$\overline{\text{External Interrupt}}$	2 CPU Cycle
$\overline{\text{Light Pen}}$	1 CPU Cycle
I/O Input	1 CPU Cycle
Hold Request	1 CPU Cycle (Note 3)

FIFO

Fall through should not be greater than 4 character times (character time = $1/f_{XTAL}$ × #dots/cell).

Throughput rate must be at least the character rate (character rate = 1/character time).

Capacitance $T_A = 25°C$, $V_{CC} = V_{SS} = 0V$

Symbol	Parameter	Test Conditions	Min	Max	Units
C_{IN}	Input Capacitance	F_C = 1 MHz (Note 5)		10	pF
C_{OUT}	Output and Reset	Unmeasured Pins Returned to V_{SS} (Note 5)		20	pF

AC Electrical Characteristics in CPU Cycle Time

CPU AND ROM EXPAND BUS TIMING (FOR REFERENCE ONLY)

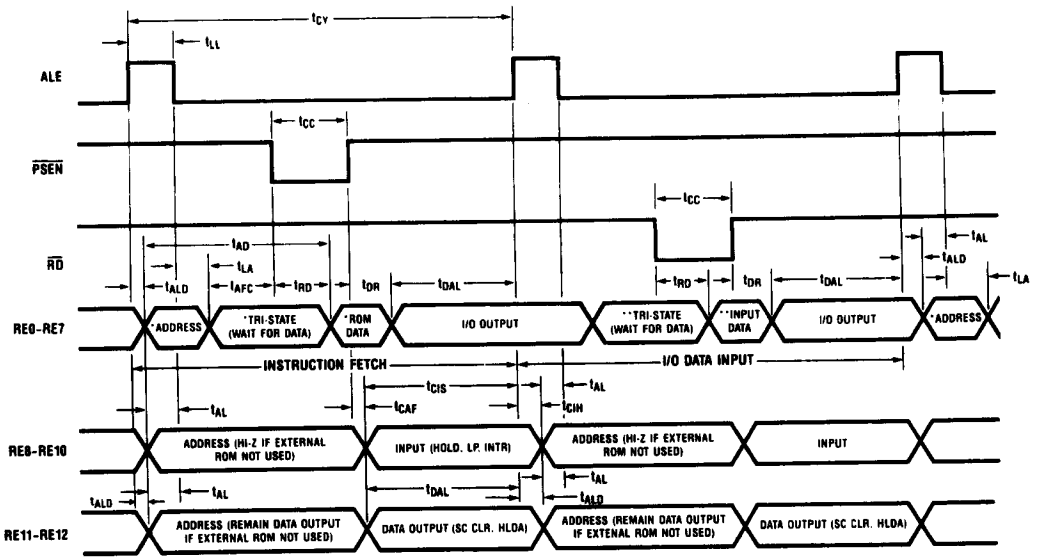
Symbol	Parameter		Typ
t_{LL}	ALE Pulse Width		14 $t_{CY/60}$
t_{AL}	Address Setup to ALE		8 $t_{CY/60}$
t_{LA}	Address Hold from ALE		6 $t_{CY/60}$
t_{CC}	Control Pulse Width	$\overline{PSEN}$ $\overline{RD}$	24 $t_{CY/60}$ 36 $t_{CY/60}$
t_{CY}	CPU Cycle Time		60 $t_{CY/60} = 15/f_{CPU} = \frac{15}{f_{XTAL} \div 1 \text{ or } \div 1.5}$
t_{DR}	Data Hold		-2 $t_{CY/60}$
t_{RD}	Control Pulse to Data In	$\overline{PSEN}$ $\overline{RD}$	18 $t_{CY/60}$ 30 $t_{CY/60}$
t_{AD}	Address Setup to Data In		32 $t_{CY/60}$
t_{AFC}	Address Float to	$\overline{PSEN}$ $\overline{RD}$	2 $t_{CY/60}$ 2 $t_{CY/60}$
t_{CAF}	PSEN to Address Float		0 $t_{CY/60}$
t_{DAL}	Data Setup to ALE	RE0–7 RE8–10 RE11–12	6 $t_{CY/60}$ -2 $t_{CY/60}$ 16 $t_{CY/60}$
t_{ALD}	Data Hold from ALE	RE0–7 RE8–12	2 $t_{CY/60}$ 6 $t_{CY/60}$

SYSTEM BUS TIMING (FOR REFERENCE ONLY)

Symbol	Parameter	Ticks	
		Min	Max
t_{EL}	RAM ALE Low Time	14 $t_{CY/60}$ − 42 ns	
t_{EH}	RAM ALE High Time	6 $t_{CY/60}$ − 25 ns	
t_{AS}	Address Setup to RAM ALE	4 $t_{CY/60}$ − 60 ns	
t_{AH}	Address Hold from RAM ALE	2 $t_{CY/60}$ − 40 ns	
t_{RCY}	Read or Write Cycle Time		
t_{RR}	$\overline{RAM\ RD}$ Width	12 $t_{CY/60}$ − 40 ns	
t_{AR}	Address Setup to $\overline{RAM\ RD}$	6 $t_{CY/60}$ − 45 ns	
t_{RRD}	Data Access from $\overline{RAM\ RD}$		10 $t_{CY/60}$ − 70 ns
t_{RDR}	Data Hold from $\overline{RAM\ RD}$		
t_{WFI}	$\overline{FIFO\ In\ Clock}$ Width	12 $t_{CY/60}$ − 40 ns	
t_{WW}	$\overline{RAM\ WR}$ Strobe Width	8 $t_{CY/60}$ − 27 ns	
t_{AW}	Address Setup to $\overline{RAM\ WR}$	10 $t_{CY/60}$ − 90 ns	
t_{DW}	Data Setup to $\overline{RAM\ WR}$	2 $t_{CY/60}$ − 30 ns	
t_{WD}	Data Hold from $\overline{RAM\ WR}$	2 $t_{CY/60}$ − 20 ns	

Timing Waveforms

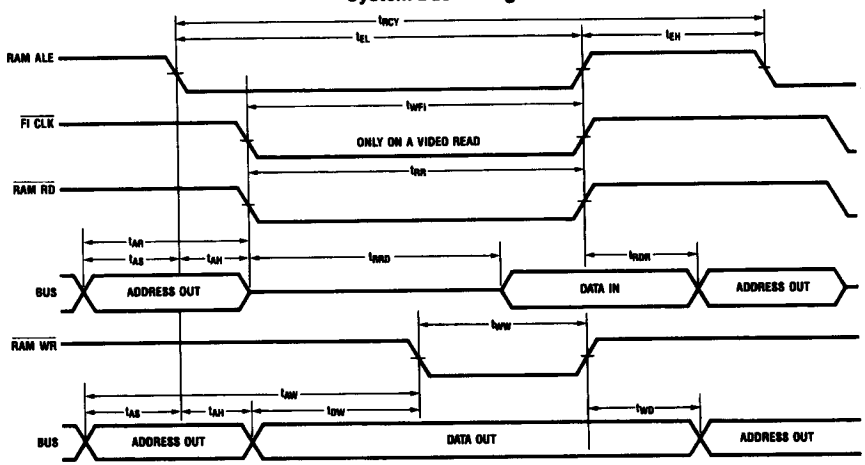
ROM Expand Bus Timing
(In Port Instruction is Shown)



TL/DD/5526–3

*Remain I/O OUTPUT if External ROM not used.
**I/O Data input or 2nd ROM byte of 2 byte instruction. Otherwise remain I/O OUTPUT.

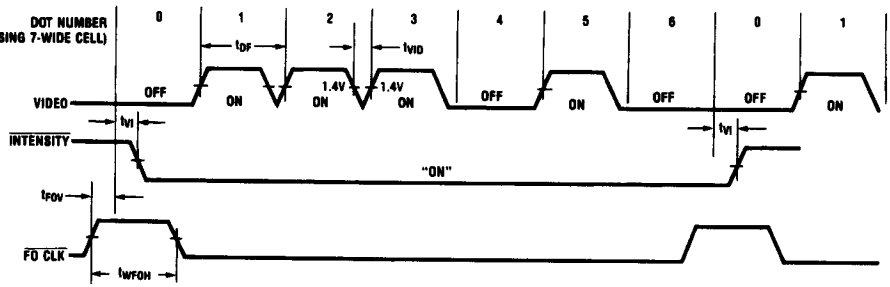
System Bus Timing



TL/DD/5526–4

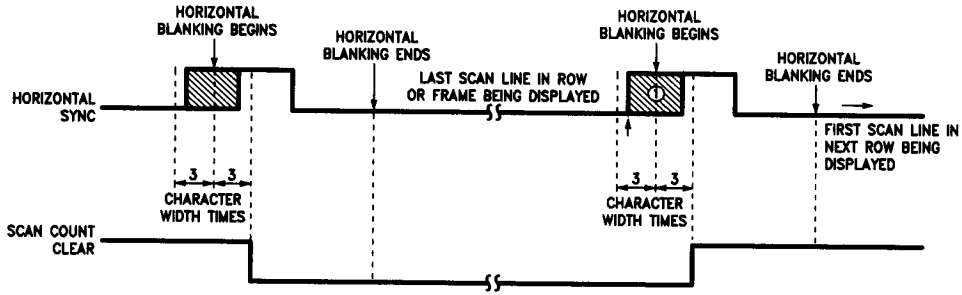
Timing Waveforms (Continued)

Video Timing



TL/DD/5526-5

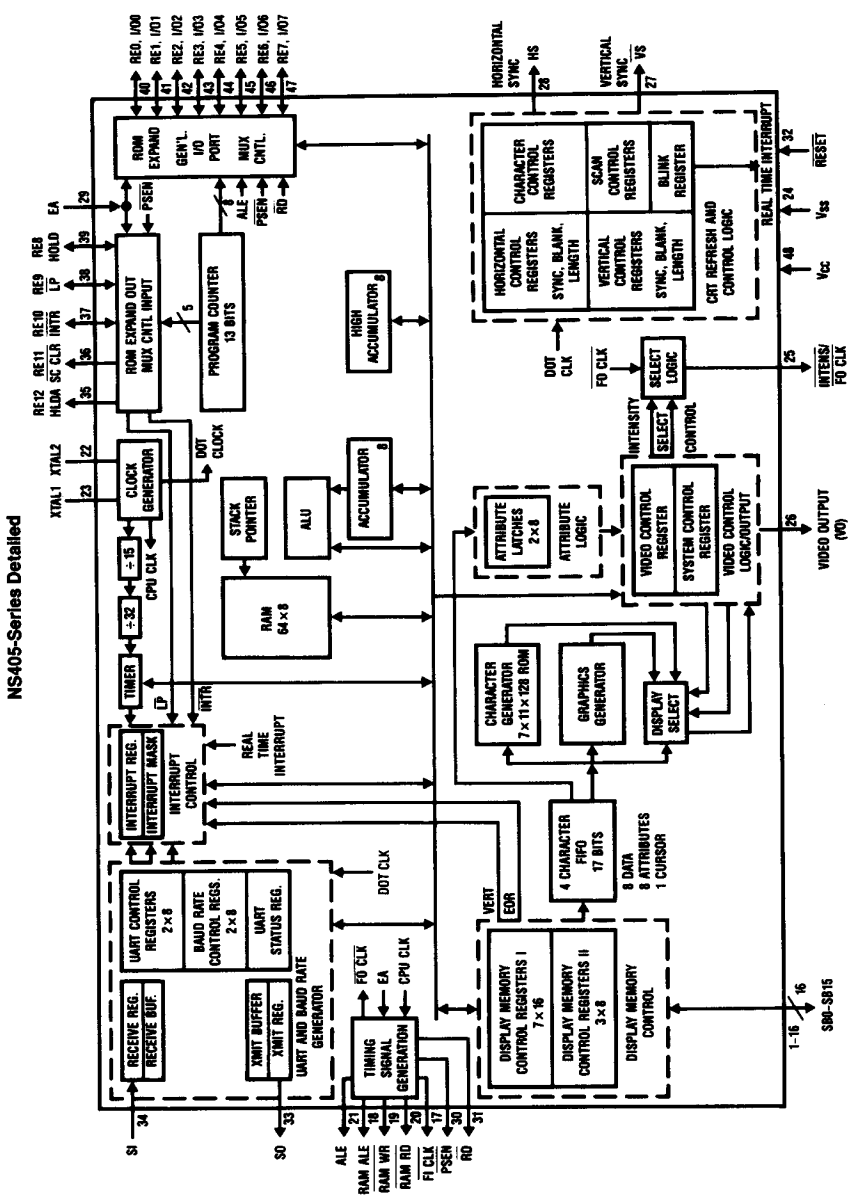
Scan Count Clear Timing



TL/DD/5526-6

For external character generation this edge is used to clock CLEAR into scan line counter. The edge must come before Scan Count Clear goes away, but not before the video controller has brought in all necessary display information for the last scan line.

Block Diagram



1.0 Functional Pin Descriptions

1.1 SUPPLIES

Pin	Name	Function
48	V_{CC} — Power	5V ±10%
24	V_{SS} — Ground Reference	

1.2 INPUT SIGNALS

23, 22	XTAL1, XTAL2 — Crystal 1, 2:	Crystal connections for clock oscillator (3–18 MHz).
29	EA — External Access:	Pull HIGH (V_{IH_2})
32	$\overline{\text{RESET}}$	An active low input that initializes the processor. The $\overline{\text{RESET}}$ input is also used for internal ROM verification.
34	SI — Serial Input:	Drives receiver section of UART (true data).

1.3 OUTPUT SIGNALS

33	SO — Serial Output:	Driven by transmitter section of UART (true data).
21	ALE — Address Latch Enable:	ROM address is available on the ROM Expand Bus and may be latched on the falling edge of ALE. Port output data may be latched on the rising edge of ALE. ALE pulses are always present, even if EA is tied low.
30	$\overline{\text{PSEN}}$ — Program Store Enable:	Enable external ROM output drivers when low. $\overline{\text{PSEN}}$ is idle (high) when the CPU fetches from internal ROM.
31	$\overline{\text{RD}}$ — Read Port Data:	Accept Port input data on ROM Expand Bus RE0–RE7 while low. ROM Expand Bus is in high impedance state while $\overline{\text{RD}}$ is low.
28	HS — Horizontal Sync	The rising edge of HS is controlled by the Horizontal Sync Begin Register and the falling edge is controlled by the Horizontal Sync End Register. HS is disabled (low) if bit 5 of the Video Control Register = 0.
27	$\overline{\text{VS}}$ — Vertical Sync Output:	The falling edge of $\overline{\text{VS}}$ is controlled by the Vertical Sync Begin Register and the rising edge is controlled by the Vertical Sync End Register. $\overline{\text{VS}}$ is at TRI-STATE if bit 5 of the Video Control Register = 0.
26	VO — Video Output:	High = beam on, low = beam off. VO is disabled (low) if bit 5 of the Video Control Register = 0.
25	$\overline{\text{INTENS}}$/$\overline{\text{FO CLK}}$	(Shared pin) $\overline{\text{INTENS}}$ Signal under attribute control may be used to switch the bistable brightness of display characters. $\overline{\text{FIFO Out Clock}}$ may be used to clock data from an external FIFO in synchronism with data from the internal FIFO. Both CANNOT be used simultaneously.
17	VID CLK/$\overline{\text{FI CLK}}$ — Video Dot Clock Out/ FIFO IN CLOCK	(Shared pin) The rising edge of the Video Dot Clock may be used to clock the data out of the video output pin. FIFO In Clock may be used to clock data from an extended attribute RAM into an external FIFO in synchronism with the data loaded into the internal FIFO. Both CANNOT be used simultaneously.
18	RAM ALE — RAM Address Latch Enable:	RAM address is available on the System Bus and may be latched on the falling edge of RAM ALE. Only operational when Display RAM accesses being performed. Otherwise high.
20	$\overline{\text{RAM RD}}$ — RAM Read:	Enable display RAM data onto the System Bus when $\overline{\text{RAM RD}}$ is low.
19	$\overline{\text{RAM WR}}$ — RAM Write:	Data to RAM is available on the System Bus and may be written at the rising edge of $\overline{\text{RAM WR}}$.

1.4 BUS — I/O

1–8	SB0–SB7 — System Bus 0–7:	Display RAM address is output while RAM ALE is high and may be latched on the falling edge of RAM ALE. System Bus accepts data input while $\overline{\text{RAM RD}}$ is low and outputs data while $\overline{\text{RAM WR}}$ is low.
9–16	SB8–SB15 — System Bus 8–15:	Normally, Display RAM address is output and held on these pins for the full read or write cycle. However, if bit 4 of the System Control Register is set, these pins function bidirectionally like SB0–SB7 to allow 16-bit data words for attribute operation.
35–47	RE0–12 — ROM Expand Bus 0–12:	Used for program ROM expansion as described below. Time multiplexed with I/O port and system control signals. I/O port and system control signals only if no external ROM used.
40–47	RE0–RE7	Low order ROM address is output and may be latched on the falling edge of ALE. Enable ROM data to this Bus when $\overline{\text{PSEN}}$ is low. Enable I/O port input data to the Bus when $\overline{\text{RD}}$ is low. Use the rising edge of ALE to latch port output data.

1.0 Functional Pin Description (Continued)

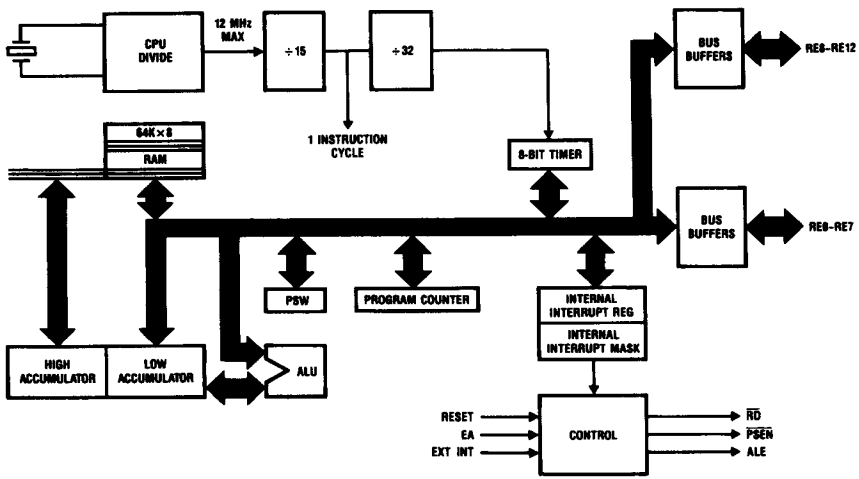
Pin	Name	Function
39–35	RE8–RE12	Five most significant bits of the ROM address are output during ALE and remain stable until data is read in during $\overline{PSEN}$. These pins are multiplexed with the HLDA, $\overline{INTR}$, $\overline{LP}$, $\overline{SC\ CLR}$, and HOLD signals.
37	$\overline{INTR}$ — Interrupt: RE10	An active low input that interrupts the processor if the external interrupt is enabled. Because it shares a pin with RE10, $\overline{INTR}$ may be driven directly only if no external ROM is used (EA is low). Otherwise must be driven through a 3.9k resistor.*
38	$\overline{LP}$ — Light Pen Interrupt: RE9	An active low input that interrupts the processor if internal interrupts are enabled and bit 5 in the Interrupt Mask Register is set. Because it shares a pin with RE9, $\overline{LP}$ may be driven directly only if EA is low. Otherwise, must be driven through a 3.9k resistor.*
39	HOLD — HOLD request: RE8	When high, requests that the NS405 enter the Hold mode. When in the Hold mode the System Bus will be in a high impedance state. The Hold mode is granted at the beginning of the next vertical retrace. Because it shares a pin with RE8, HOLD may be driven directly only if EA is low. Otherwise, must be driven through a 3.9k resistor.*
35	HLDA — Hold Acknowledge: RE12	This output is asserted in response to Hold and provides handshake capability with another processor (active high). For more detailed information see Section 3.0 Slave Processing. Because HLDA shares a pin with RE12, the HLDA state is preset only during the interval preceding the rising edge of ALE. However, if no external ROM is used, HLDA is a steady state output and need not be latched externally.
36	$\overline{SC\ CLR}$ — Scan Count Clear: RE11	This output clears an external scan counter when used with an external character generator. It is a low going pulse which occurs during the horizontal retrace preceding the first scan line of each character row. Because $\overline{SC\ CLR}$ shares a pin with the RE11, the correct $\overline{SC\ CLR}$ state is present only during the interval preceding the rising edge of ALE. However, if no external ROM is used, $\overline{SC\ CLR}$ is a steady state output and need not be latched externally.

*Unused control inputs must be terminated

2.0 Functional Description

2.1 CPU

The CPU of the NS405 is patterned after the 8048 single chip microcomputer (see *Figure 1*).



TL/DD/5526–8

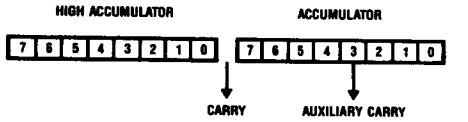
FIGURE 1. NS405 Series CPU Block Diagram

2.0 Functional Description (Continued)

2.1.1 Accumulator — High Accumulator

In addition to the regular 8-bit Accumulator, there is an 8-bit High Accumulator extension to facilitate the 16-bit operations required for display memory management. The HACC/ACC pair is usually used in conjunction with the 16-bit RAM pointer registers (RA, R0 and RB, R1, CURSOR, HOME, BEGD and ENDD) to effect video data transfers. In addition, external attribute memory is loaded in a 16-bit transfer operation. Any instruction which causes a carry or borrow out of the low accumulator will affect the high accumulator (see *Figure 2*).

Auxiliary carry is used only when converting the accumulator contents from binary to BCD (binary coded decimal) using the DA A instruction. The auxiliary carry flag can be cleared by moving a zero into bit 6 of the program status word.

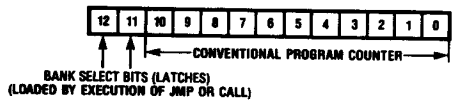


TL/DD/5526–9

FIGURE 2. CPU Accumulator

2.1.2 Program Counter (PC)

The Program Counter is a 13-bit wide register which provides program addressing for the CPU. The lower 11 bits operate like a conventional program counter while the upper 2 bits are actually latches. These 2 latches are automatically loaded from the bank select flip-flops (PSW bits 3, 4) whenever a JMP or CALL instruction is executed. The bank select flip-flops in turn are only modified upon the execution of a Select Memory Bank Instruction or modification of the PSW (see *Figure 3*).



TL/DD/5526–10

FIGURE 3. TMP Program Counter

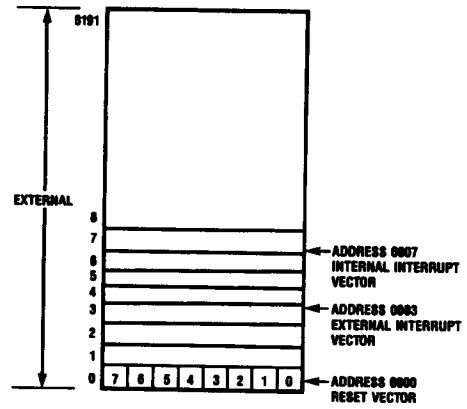
2.1.3 Program Memory

Memory is subdivided into 2k banks with accesses limited to the currently selected bank unless a Bank Change sequence has been executed. Upon reaching the end of a memory bank, the program counter will wrap around and point to the beginning of the current bank.

Each bank is further subdivided into pages of 256 bytes each, with 8 pages in every bank. The conditional JUMP instructions are restricted to operate within the memory page that they reside in.

Because of the sequence which the CALL instruction executes when pushing and loading the PC, it is possible to easily call and return from subroutines located in different memory banks (see *Figure 4*).

Upon executing an RET or RETR instruction for a call from one memory bank into another, a SEL MBx instruction should be excuted to restore the memory bank select flip-flops to their original bank. However, no SEL MBx is needed after an interrupt since the flip-flops were never modified.



TL/DD/5526–11

FIGURE 4. Program Memory Map

2.1.4 Program Status Word Bit Assignments

Bit Position	Contents
0	Stack Pointer Bit, S0
1	Stack Pointer Bit, S1
2	Stack Pointer Bit, S2
3*	Memory Bank Select Bit 0
4*	Memory Bank Select Bit 1
5*	Register Bank Select Bit (0 = Bank 0, 1 = Bank 1)
6*	Auxiliary Carry. A carry from Bit 3 to Bit 4 generated by an add operation. Used only by the decimal adjust (DA A) instruction.
7*	Carry. A bit indicating the preceding operation resulted in an overflow or an underflow from the 8-bit accumulator.

***Note 1:** Bits 3 through 7 are saved on the stack by subroutine calls or interrupts. Bits 3 and 4 are restored upon execution of an RET instruction, whereas all 5 bits are restored by RETR.

Note 2: F0 is not saved on the stack (as in an 8048).

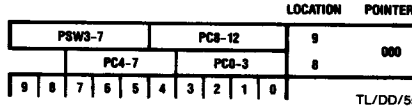
Note 3: Bits 0–5 cleared on a RESET.

2.1.5 Stack Pointer (SP)

The stack pointer is an independent 3-bit counter which points to designated locations in the internal RAM that holds subroutine return parameters. The stack itself is located in RAM locations 8–23 (see *Figure 5*).

Each entry in the stack takes up two bytes and contains both the PC and status bits. When reset to zero, the stack pointer actually points to locations 8 and 9 in RAM. Since the stack pointer is a simple up/down counter, an overflow will cause the deepest stack entry to be lost (the counter overflows from 111 to 000 and underflows from 000 to 111).

Note: If the level of subroutine nesting is less than eight (8), the unneeded stack locations may be used as RAM.



TL/DD/5526–12

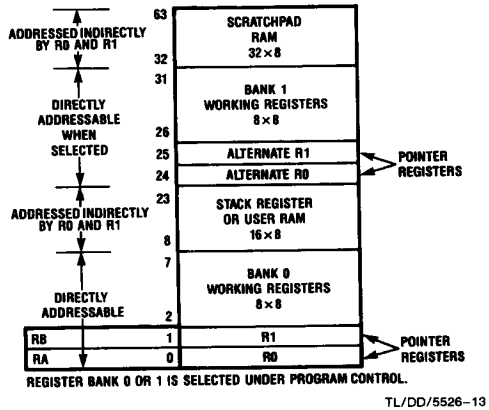
Note: The odd numbered RAM bytes in the stack area have two (2) extra bits to allow for storage of the bank select switch bits. This feature allows interrupt routines and subroutines to be located outside the current 2k program memory bank.

FIGURE 5. Typical Stack Composition

2.0 Functional Description (Continued)

2.1.6 Data Memory (On-Chip RAM)

The data memory nominally consists of 64 8-bit locations and is utilized for working registers, the subroutine stack, pointer registers and scratch pad. There are two sets of working/pointer registers (R0–R7) which are selected by the Select RAM Bank instruction. The stack area is located in locations 8–23. Locations 32–63 contain the scratch pad memory. To facilitate 16-bit Video Memory Management there are two 8-bit extension registers (RA and RB) which are associated with the R0 and R1 registers respectively of whichever RAM bank is currently selected (see *Figure 6*). i.e., There is only one RA register and only one RB register.

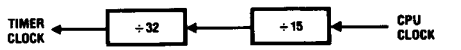


TL/DD/5526–13

FIGURE 6. RAM Memory Map

2.1.7 Timer

The On-Board Timer is an 8-bit up counter which sets the Timer Overflow Flag and generates an internal interrupt (if enabled) whenever it overflows from FF to zero. The Timer may be stopped, started, loaded and read from by the CPU. The Timer clock is derived from the CPU clock as shown in *Figure 7*. Whenever a Start Timer instruction is executed the ÷32 is initialized to its zero state to insure a full count measurement. After overflow the timer keeps counting until the next FF to zero overflow at which time the overflow flag will be set and another interrupt generated. The overflow flag can only be reset through the JTF and JNTF instructions.



TL/DD/5526–14

FIGURE 7. Timer Clock Generation

2.1.8 Interrupts

The interrupt circuitry handles two generic classes of interrupt conditions called Internal and External. Either class has its own master control which can be activated through software enable and disable instructions. On an interrupt service the currently executing instruction is completed, then two CPU cycles are used as the program counter and bits 3–7 of the PSW are pushed onto the stack and stack pointer is incremented.

Then the interrupt vector address (3 or 7) is loaded into the PC and service started. Whenever an interrupt condition is being serviced all other interrupts of either class are locked out until a RETR instruction is executed to conclude interrupt service. If both an external and internal interrupt arrive at the same time, the external interrupt is recognized first.

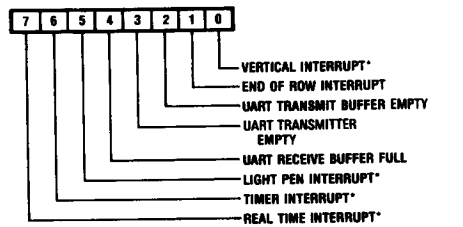
2.1.8.1 External Interrupt

The External Interrupt consists solely of the shared $\overline{\text{INTR}}$/RE10 pin. External interrupts on this pin will be detected if the setup and hold times as shown in the timing diagrams are met. This pin is a level sampled interrupt which means that as long as the pin is low during the sampling window an interrupt will be generated. In addition, the $\overline{\text{INTR}}$ pin is the only external pin whose logic state can be directly tested through software.

2.1.8.2 Internal Interrupts

The Internal Interrupts consist of seven internal operational conditions plus the light pen arranged in an 8-bit wide register as shown in *Figure 8*. Activation of an internal interrupt condition causes a corresponding register bit to be set, *Figure 9*. Each internal interrupt may be individually masked out through the Interrupt Mask register which has the same bit assignments as the Interrupt register and can be loaded from the accumulator. A zero in the Interrupt Mask register inhibits the interrupt and a one enables it. Further interrupt processing is as shown. To determine which of the eight internal conditions caused the interrupt the CPU must read the Interrupt register into the accumulator. To acknowledge receipt of the interrupt certain bits are automatically cleared on a read while others are reset upon service of the particular interrupt.

The conditions under which each of the interrupts are generated and cleared are as follows:



TL/DD/5526–16

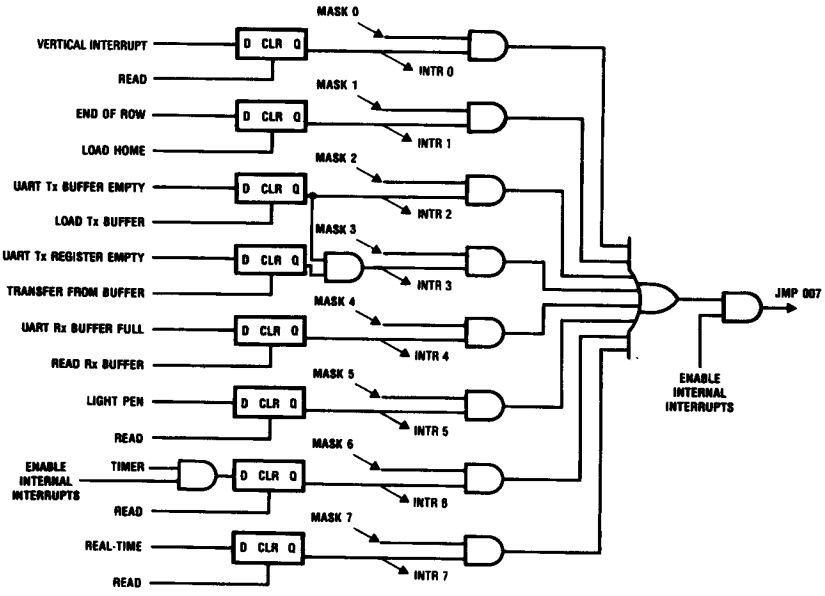
Note: The interrupt flags indicated by an asterisk (*) are cleared when the Interrupt Register is read.

FIGURE 8. Internal Interrupt Register

Bit

0 Vertical Interrupt—Generates an interrupt at the end of the display row designated by the Vertical Interrupt Register. Interrupt bit cleared on a CPU read of the interrupt register. If VIR > Vertical Length Register no interrupt will be generated.

2.0 Functional Description (Continued)



TL/DD/5526–15

FIGURE 9. Internal Interrupt Processing

Bit

1 End of Row Interrupt—Generates an interrupt at the end of each display row when the Current Row Start Register is updated for the next row. Used in conjunction with the Row Sequencing Control Bit (5) in the System Control Register to implement Row Pointer Look-Up Tables and Horizontally Split Screens. Interrupt bit cleared on a CPU write to the Home Register. Does not generate interrupts for those rows blanked during vertical blanking.

2 UART Transmit Buffer Empty—Generates an interrupt when the Transmit Buffer empties out after dumping a character into the Transmit Shift Register. Interrupt bit cleared on a CPU write to the Transmit Buffer.

3 Transmitter Empty—Generates an interrupt when BOTH the Transmit Buffer and Transmit Shift Register are empty. The interrupt bit is cleared when the CPU loads the transmit buffer.

4 UART Receiver Buffer Full—Generates an interrupt when the Receiver Buffer fills up with a character from the Receive Shift Register. Interrupt bit cleared on a CPU read of the Receiver Buffer.

5 Light Pen Interrupt—Generates an interrupt on each falling edge detected on the shared $\overline{LP}$/RE9 pin. Since only falling edges generate interrupts and the input is sampled each CPU Cycle, a high level must be sampled between falling edges in order to be considered a new interrupt. This interrupt is used to latch the light pen position registers. For further information see Light Pen Description. Interrupt bit cleared on a CPU read of the interrupt register.

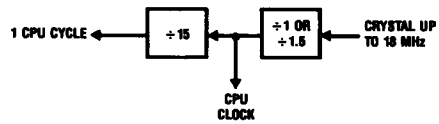
Bit

6 Timer Interrupt—Generates an interrupt when the internal 8-bit Timer overflows from FF to 00. Interrupt bit cleared on a CPU read of the interrupt register.

7 Real-Time Interrupt—Generates interrupts at a software programmable frequency that is generally in the Hertz range. (See CPU Clock Generation.) Thus permitting the implementation of a real-time clock or timer. Interrupt bit cleared on a CPU read of the interrupt register.

2.1.9 Clock Generation

All chip clocks are derived from the one external crystal connected between pins 22 and 23. This master clock also doubles as the video dot clock. The crystal frequency is constrained to lie within the range of 3 to 18 MHz. The CPU clock is derived from the crystal clock by either using it directly or by dividing down by a factor of 1.5 *(Figure 10)*.



TL/DD/5526–17

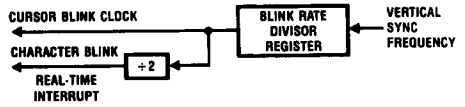
FIGURE 10. CPU Clock Generation

The choice is software programmable through bit 0 in the System Control Register. The exact selection is made in consideration of the fact that the CPU clock must lie within the range of 3 to 12 MHz. In addition, the choice of divide by modes will also impact the display character cell width due to the nature of the video controller. Specifically with ÷1.5

2.0 Functional Description (Continued)

the cell width must be ≥ 8 dots wide whereas with ÷1 the cell width must be ≥ 6 dots wide.

The low clock rates necessary to implement Cursor Blinking, Character Blinking and the Real-Time Interrupt are derived by passing the vertical sync frequency through a 5-bit Blink Rate Divisor Register, *(Figure 11)*. The resultant frequency is used as the Cursor Blink Clock. This clock is then further divided by 2 to yield the Character Blink and Real-Time Interrupt Clocks. For example, to get a 1 Hz real time interrupt, with a 60 Hz system, set the 5 bit Divisor Register to 30 in order to yield a 2 Hz signal which is then divided by 2.

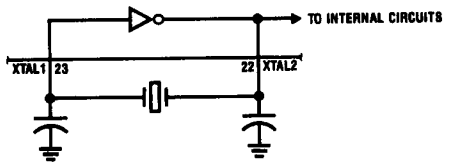


TL/DD/5526–18

FIGURE 11. Blink Clock Generation

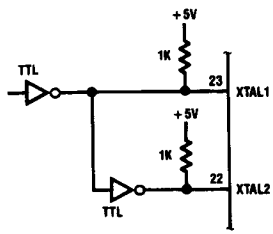
2.1.10 Oscillator Operation

The on-board oscillator circuit consists of a phase inverter which, when used with an external parallel resonant tank, *(Figure 12a)*, will yield the required oscillator clock. Crystals should be specified for AT cut and parallel resonant operation with the desired load capacitance (typically 20 pF). If one desires to externally generate the clock and input it to the chip, he may do so by driving XTAL1 (pin 23) and XTAL2 (pin 22) as shown in *Figure 12b*.



TL/DD/5526–19

FIGURE 12a. TMP Oscillator



TL/DD/5526–20

Note: Use AS TTL devices if faster than 12 MHz.

FIGURE 12b. External Oscillator Mode

2.2 DISPLAY MEMORY CONTROLLER

The video display data resides in the external Video Memory which is managed by the Display Memory Controller (DMC) through the System Bus. Either the CPU or the Video Controller may access the display memory by presenting its requests to the DMC. A maximum of three Video Memory accesses (Reads or Writes) can be performed by the DMC during each CPU instruction execution cycle. Because the CPU can access the Video Memory, one may expand CPU I/O or data memory by memory mapping into the Video Memory space. Up to 64k locations may be addressed over the 16-bit System Bus. Data word widths may be 8 or 16 bits depending upon whether external character attribute selection is used. The actual bus multiplexing mode is controlled by bit 4 in the System Control register. The Video Controller has the highest priority in obtaining Video Memory accesses with the CPU getting in on a space available basis. If all memory accesses are being taken by the Video Controller (rarely), the CPU is put into a wait state should it try to access video memory. To ease accessing requirements and boost throughput the Video Controller utilizes a 4-level data FIFO which is normally kept full of display data.

2.2.1 Display Memory Control Registers

In order to facilitate the management of video data for such features as a Screen scroll, memory paging and row lookup the DMC utilizes a number of registers which address the video RAM space. Each of these pointers is 16 bits wide and writable or readable from the 16-bit HACC/ACC pair as the case may be. There are 2 video data accessing modes as determined by bit 5 in the SCR, Sequential and Table Lookup. The functions of the pointer registers vary depending upon the accessing mode selected. Their designators are:

HOME = Home address register. Read and write.

BEGD = Beginning of diplay RAM. Write only.

ENDD = End of display RAM. Write only.

CURS = Cursor address register. Read, Write, Increment, Decrement.

SROW = Status section register. Write only.

CRSR = Current row start register. Not directly accessed.

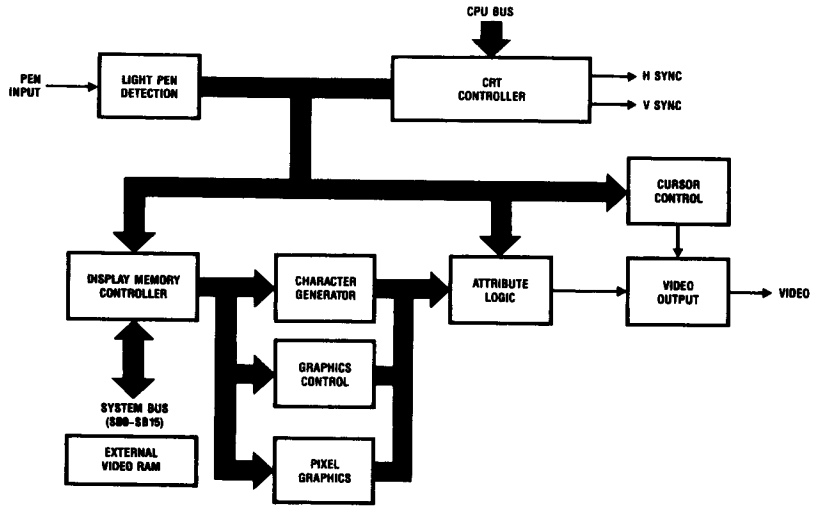
2.2.2 Sequential Access Mode

In this mode display data is accessed from sequential address locations in the video memory until the data requirements for the current screen field are fulfilled. The location from which the first display character is taken is the one pointed to by the HOME register. By modifying the contents of HOME one may implement a row scroll or paging operation. The BEGD and ENDD are used to control the wrap-around condition when HOME gets near the end of available display RAM as determined by ENDD. In this instance, when sequential accessing brings us to the end of memory as pointed to by ENDD, the controller wraps around by jumping back to the beginning of display memory as pointed to by BEGD. The value in ENDD should be the last location in display memory + 1. Also the size of the display memory between BEGD and ENDD (ENDD − BEGD) must be an integral number of display rows. The CURS in both accessing modes merely identifies the current cursor position in display memory so that the cursor characteristics can be inserted into the video at the appropriate character position.

In addition to the display of normal video data one may elect to have a special status section displayed using data from a separate section of video memory. The status section would consist of an integral number of display rows on the bottom of the screen. This feature operates by reloading the video RAM pointer with the contents of SROW when the desired row position at which to start the status section comes up. The particular row at which the status display starts is defined in the Timing Chain. Once the video RAM pointer is jumped to SROW, data accessing again proceeds sequentially from there until the data requirements for the current field are satisfied.

2.0 Functional Description (Continued)

TMP Video Section



TL/DD/5526–21

Whether a status section is used or not, upon accessing all of the data necessary to display a field, the video RAM pointer is reset to HOME in preparation for the display of a new field.

2.2.3 Table Lookup Mode

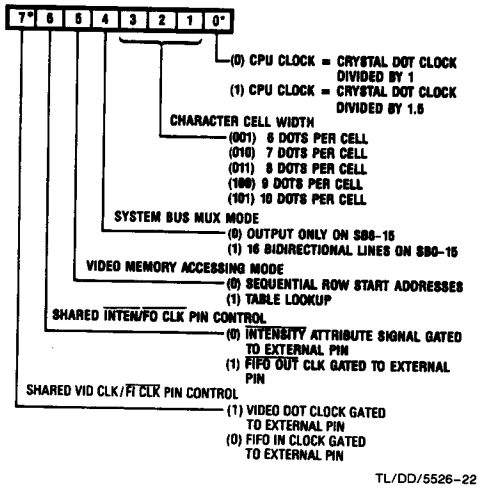
The CRSR (transparent to the user) is a pointer to the address of the first character in a display row. It is required because each time a scan line is displayed, all display characters in the row must be accessed anew. Since a row is made up of a number of scan lines, we must recover the address of the first character in the row for each scan in the row. After a row is done, the CRSR is normally advanced to point to the first character in the next row.

In table look-up mode the starting memory location of the next row is loaded into the CRSR from the HOME register at the end of each row. The HOME register was presumably updated by the CPU since the last end of row.

A CRSR load also generates the internal End of Row interrupt which the CPU will use as a signal to reload HOME. Finally, reloading HOME will clear out the End of Row interrupt. If the status section feature is used, upon reaching the begin status row location the CRSR will be loaded with SROW instead of HOME for that row. After which CRSR will revert back to load from HOME for the remaining rows on the screen.

2.3 SYSTEM CONTROL REGISTER

Through the System Control Register (SCR) the user specifies several important chip operational conditions. It is an 8-bit write only register which is loaded from the CPU accumulator.



TL/DD/5526–22

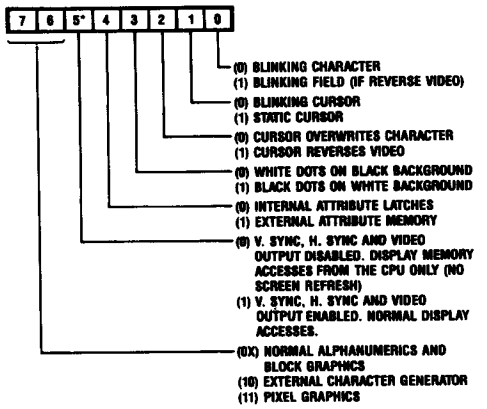
*Bit 0 is set to 1 by RESET and bit 7 is set to 0 by RESET.

FIGURE 13. System Control Register

2.4 VIDEO CONTROL REGISTER

Through the Video Control Register (VCR) the user specifies several video display features to the chip. It is an 8-bit write only register which is loaded from the CPU accumulator.

2.0 Functional Description (Continued)



TL/DD/5526–23

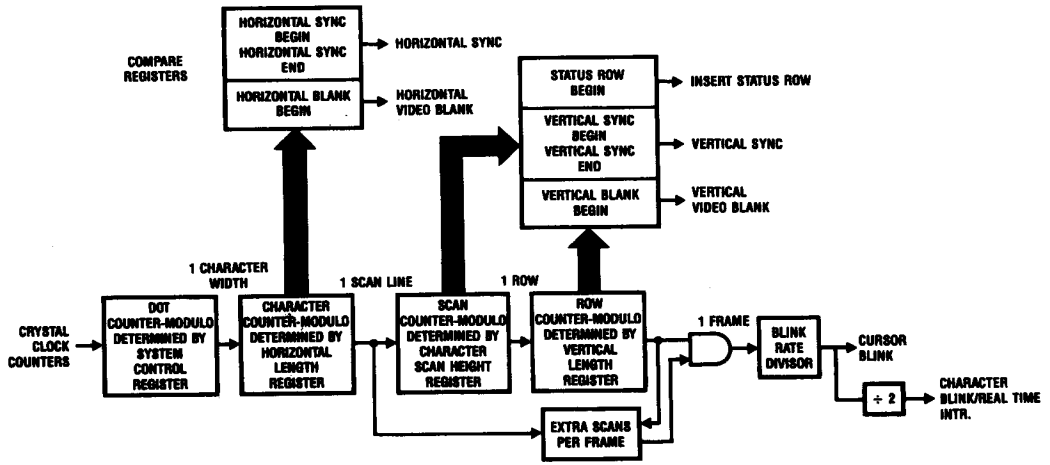
*Bit 5 is set to 0 by RESET.

FIGURE 14. Video Control Register

2.5 CRT REFRESH LOGIC

All video timing and clocking signals are derived from a series of counters and comparators called the Video Timing Chain. The chain is driven by the dot/crystal clock and ultimately divides down to the very slow blink clock, *(Figure 15)*. By having the program initialize the registers in the chain a user may specify all aspects of video generation.

The chain also controls the size and placement of the cursor and underline attribute within a character cell as well as the cell partitioning for block graphics display. All totaled, the chain consists of 14 wire only registers. They are loaded indirectly by using the Timing Chain Pointer (TCP), a 4-bit pointer to registers in the chain, and the MOV @TCP, A instruction.



TL/DD/5526–24

FIGURE 15. TMP Video Timing Chain

2.5.1 TMP Timing Chain Registers

TCP	Horizontal Timing
0	Horizontal Length Register — HLR 7 bits — Total number of character cells in a horizontal scan and retrace. — Enter desired count − 1
1	Horizontal Blank Begin Register — HBR 7 bits (Characters/Row) — Character position in horizontal scan after which horizontal blanking begins. — Enter desired number of displayed characters/row − 1.
2	Horizontal Sync Begin Register — HSBR 7 bits — Character position in horizontal scan after which horizontal sync begins (rising edge), HSBR ≤ HLR. — Enter desired count + 2.

2.0 Functional Description (Continued)

2.5.1 TMP Timing Chain Registers (Continued)

TCP — **Horizontal Timing**

3 Horizontal Sync End Register — HSER 7 bits
- — Character position in horizontal scan after which horizontal sync ends (falling edge), HSER ≤ HLR.
- — Enter desired count + 2.

Note: The polarity of the horizontal sync signal can be inverted by switching the values in the two horizontal sync registers.

TCP — **Character Height Definition**

4 High Nibble — Character Scan Height Register — CSHR 4 bits (see *Figure 16a*)
- — Scan line height of a character cell.
- — Enter desired number of scan lines − 1.

4 Low Nibble — Extra Scans/Frame — ES/F 4 bits
- — Number of extra scans to be added to a frame if desired.
- — Enter desired number of extra scans −1.
- — To get no extra scans make ES/F = CSHR. ES/F must be ≤ CSHR.

TCP — **Vertical Timing**

5 Vertical Length Register — VLR 5 bits
- — Total number of display and retrace rows in a frame.
- — Enter desired number of rows − 1.

6 Vertical Blank Register — VBR 5 bits (Rows/Screen)
- — Row position in vertical scan after which vertical blanking begins, VBR < VLR.
- — Enter desired number of displayed rows − 1.

7 High Nibble — Vertical Sync Begin Register — VSBR 4 bits
- — Scan line position in first blank row at which vertical sync begins (falling edge). Sync starts 1 char time after blanking for that line starts (except when VSBR = CSHR sync will start 1 char time after blanking of the last displayed scan line).
- — Enter desired scan line position − 1.

7 Low Nibble — Vertical Sync End Register — VSER 4 bits
- — Scan line position after start of vertical sync at which vertical sync ends (rising edge). Sync ends 1 char time after horizontal blanking for that scan line start.
- — Enter desired scan line position − 1.

Note: If VSER = VSBR there will be no vertical sync signal.

8 Status Row Begin Register — SRBR 5 bits
- — Row count after which the status row is inserted.
- — Enter desired row position − 1.

TCP — **Cursor and Graphics Control**

9 Upper 5 Bits — Blink Rate 5 bits
- — Divider driven by the vertical sync frequency to yield the slow cursor, character and real-time blink rates.
- — Enter desired divisor − 1.

9 Lower 3 Bits — Blink Duty Cycle 3 bits
- — Approximate ON time of blink signal.
- — 000 = shortest, 111 = longest (100 = 50% duty cycle).

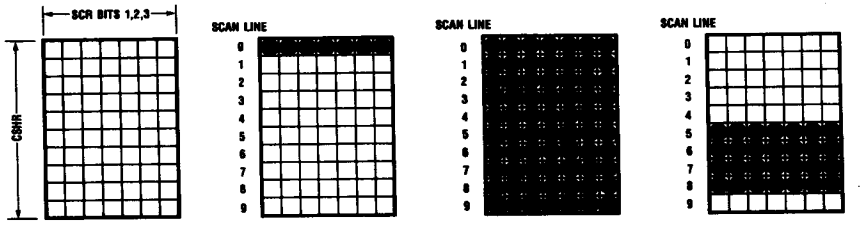
10 Graphics Column Register — GCR 8 bits
- — Assign dot positions to left, middle and right character cell columns for block graphics operation.

11 Graphics Row Register — GRR 8 bits
- — Defines scan count at which middle row for block graphics characters begins (upper nibble) and at which bottom row begins (lower nibble). The middle row (upper nibble) must be ≥ 1.
- — Enter desired scan count − 1.

12 Underline Size Register — USR 8 bits (see *Figures 16a, b, c*)
- — Defines the beginning (upper nibble) and ending (lower nibble) scan lines for the underline attribute. Values must be ≤ CSHR.

13 Cursor Size Register — CSR 8 bits (see *Figures 16a, b, c*)
- — Defines the beginning (upper nibble) and ending (lower nibble) scan lines for the cursor. Values must be ≤ CSHR.

2.0 Functional Description (Continued)



TL/DD/5526–25

a. Character Cell Format Specification

b. Underline Size Register = 90

c. Cursor Size Register = 9A (0B, 0C, 0D, 0E, 0F) may also be used

d. Cursor Size Register = 48

FIGURE 16. Underline and Cursor Register Operation

Note: The internal cursor flip-flop gets set to ON whenever a scan line corresponding to the begin cursor nibble is reached, and gets set to cursor OFF whenever a scan line corresponding to the end cursor nibble is reached. The cursor attributes are inserted whenever the character position being displayed corresponds to the one pointed to by the cursor address register. A similar situation applies for characters with the underline attribute selected. Therefore, care should be taken when setting the ES/F register and setting the cursor and underline sizes. In particular the ES/F value should not be between the upper nibble and lower nibble values of the underline size register or between the upper nibble and lower nibble values of the cursor size register. To use the cursor as a pointer without displaying it, set the lower nibble of the cursor size register to a value less than CSHR and the upper nibble to a value greater than CSHR.

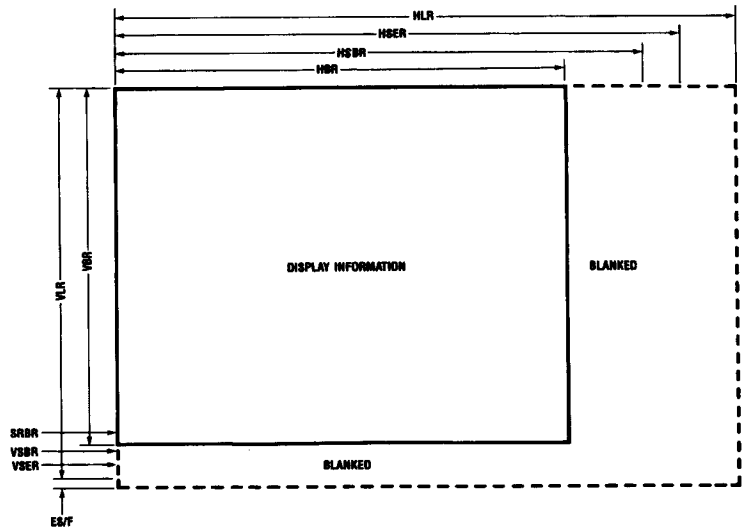
2.5.2 TIMING CHAIN LOAD VALUE EXAMPLE

It is desired to have a display field of 80 columns by 25 rows with the last screen row being a status row. It has been determined that 25 character width times will be necessary to complete horizontal retrace and that Horizontal sync should be positioned to start a full seven character times after blanking and end twenty characters after blanking to give us a total sync width of 13 character times. (See *Figure 17* for example.)

Additionally, vertical retrace will take 23 scan line times to complete with vertical sync starting three scan line times after vertical blanking begins and occupying a total period of 11 scan lines.

It is desired to make the character cells 12 scan lines tall. The cursor will be a block shape and occupy the bottom 11 scan lines in a cell. The underline attribute will actually be a strike through dash occupying the 4th scan line from the top in a cell.

Our line width is 80 displayed characters plus 25 for retrace making HLR = 80 + 25 − 1 = 104. Blanking will start after the 80th character so HBR = 80 − 1 = 79. To achieve seven character times after horizontal blanking, HSBR = 87 + 2 = 89. To achieve twenty character times after blanking HSER = 100 + 2 = 102 (note 102 − 89 = 13 total). Cell height is 12 lines so CSHR = 12 − 1 = 11. Since there are 12 scan lines per cell or row, vertical retrace will require 23/12 = 1 row and 11 scan lines. This makes our total row count VLR = 25 + 1 − 1 = 25 and ES/F = 11 − 1 = 10. Thus, timing chain location 4 would be coded: 1011 1010. We will display 25 rows so VBR = 25 − 1 = 24. Vertical sync will start at the beginning of the fourth scan



TL/DD/5526–26

FIGURE 17. Typical Video Screen Format Specification

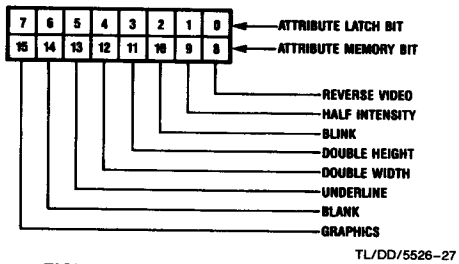
2.0 Functional Description (Continued)

line of the row after blanking begins so VSBR = 4 − 1 = 3. It will run for 11 scan lines or specifically the 4, 5, 6, 7, 8, 9, 10, 11, 12, 1, 2 ending at the beginning of the 3rd so VSER = 3 − 1 = 2. The status row will be after the 24th so SRBR = 24 − 1 = 23. To specify the underline and cursor sizes one must remember that the first scan line is numbered 0. To get our 11 line block cursor we begin after the 0 line and end at the end of the 11 line making CSR = 0000 1011. The underline dash will be USR = 0011 0100. Note that the CSHR determines the scan counter modulo and if a scan compare register value (ES/F, VSBR, VSER, USR, CSR) is never reached, the signal end or begin will never be initiated.

2.6 ATTRIBUTES

Eight independent attributes may be inserted itno the video dot stream to affect display characters on either an individual or global basis. The eight attributes along with their control word bit assignments are detailed in *Figure 18*. The scope with which a particular set of attributes affects the display depends upon whether attribute control is internal or external as determined by bit 4 in the VCR.

Attributes are present if the corresponding bit is a ZERO (low).



TL/DD/5526–27

FIGURE 18. Attribute Bit Assignments

2.6.1 Internal Attribute Selection

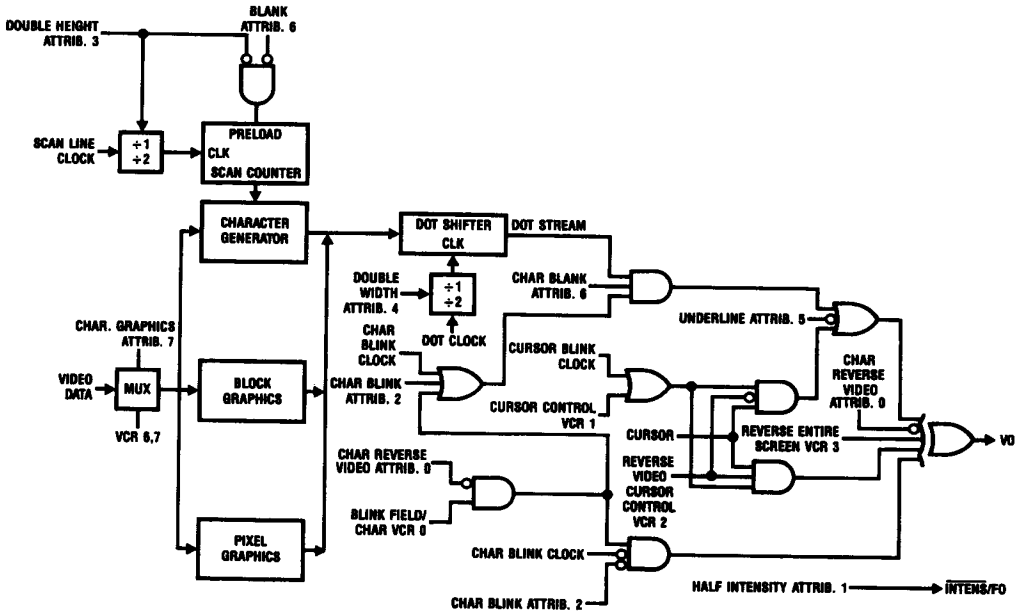
In internal mode attribute control comes from one of two internal attribute latches designated AL0 and AL1, either of which is directly loadable from the CPU accumulator. The choice of which of the two is used for a particular display character is determined by bit 7 (MSB) in the display memory data byte with 0 = AL0 and 1 = AL1. (Characters are represented in display memory as ASCII values occupying the low 7 bits of each 8-bit byte thus leaving bit 7 free for attribute control.)

2.6.2 External Attribute Selection

In external mode each display character has associated with it, a dedicated attribute field in the form of a high 8-bit extension to the regular display memory character byte. To use this mode the system bus msut be configured for 16-bit bidirectional operation (SCR bit 4 = 1) and external attributes must be selected (VCR bit 4 = 1).

2.6.3 Attribute Processing

Each of the eight attributes may be independently enabled thus yielding a number of possible combinations. The exact processing involved is shown in *Figure 19*. Note that attributes are always present. Whether any of them are active depends upon the particular control bit being enabled in the latch or memory.



TL/DD/5526–28

FIGURE 19. TMP Attribute Processing

2.0 Functional Description (Continued)

2.6.4 Attribute Operation

Reverse Video: A character and its surrounding cell are reversed in video from what was selected for the rest of the screen.

Half Intensity: To use the half intensity function the shared $\overline{\text{INTENSITY/FO CLK}}$ pin (25) must be selected for INTENSITY operation by setting SCR bit 6 low. In operation the half intensity pin will be low whenever a character for which the attribute is active is being displayed. To perform the actual attenuation function external circuitry must be connected between the $\overline{\text{INTEN}}$ and Video Output pins. In fact the signal may be used for another purpose such as switching between two colors.

Blink: A character or the field around it blinks as selected by VCR bit 0.

Double Height: A designated character is stretched out so that it will occupy a 2-row tall space. This attribute is implemented by slowing down by half the scan line stepping to the internal character generator. To use this attribute the desired double high character must be placed into the two display memory locations corresponding to the top and bottom row positions. For both locations the double high attribute is set. In addition the Blank attribute for the bottom character is also set to tell the controller it is the bottom half of a double high character. The double high attribute has no effect on element graphics or on pixel graphics displays. If an external character generator is used special circuitry must be employed to implement double high characters.

Double Width: A designated character is stretched out so that it will occupy a 2-character cell wide space. This attribute is implemented by slowing down by half the clock to the video dot shifter. To use this attribute the desired double wide character must be placed in the left character position and the double wide attribute bit set. The following character position (right) can have any character as it will be ignored.

Underline: If set this attribute causes the underline figure to be added to the video dot stream. Since the underline, like the cursor, can be specified as to position and size in the character cell, the underline can be an overline, block, strike through or any one of a number of effects. The underline overwrites any dot where it overlaps the character.

Blank/Double High Bottom: A character is inhibited from being displayed while still allowing it to be stored in the display memory. If this attribute and the double height attribute are set for the same character, the normal blank function is disabled for that character position and the character is displayed as the bottom half of a double height character.

Graphics: This attribute determines whether the video memory data byte as accessed by the display memory controller is routed through the character generator or block graphics control logic. If routed through the block graphics logic (attribute active) the effect on the video display will be as described in the Block Graphics section. Note that because Block Graphics mode is selected as an attribute it may be mixed in with normal alphanumerics characters. Also all other attributes with the exception of double height operate on the block graphics characters.

2.7 CHARACTER GENERATOR

The internal character generator holds 128 characters in a 7 x 11 matrix. The standard character sets are addressed using 7-bit ASCII codes stored in the display memory. When operating with fonts smaller than the maximum of 7 x 11, zeroes are encoded into the unused bits. When putting out a character the video controller always starts character generation on the second scan line of a row, leaving the first scan line blank. Similarly, the first (left) column in a character cell is blanked with character generation starting on the second column. Therefore, the specified cell size must be one greater in height and width than the display characters (including descenders) otherwise they will be chopped off. If the character cells are larger than the internal 7 x 11 matrix, blank dots will be put out after exhausting the internal generator (See *Figure 20* for example.)

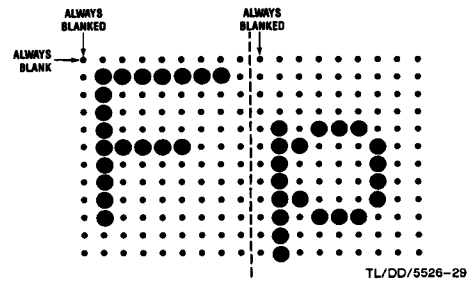


FIGURE 20. Character Cell Format

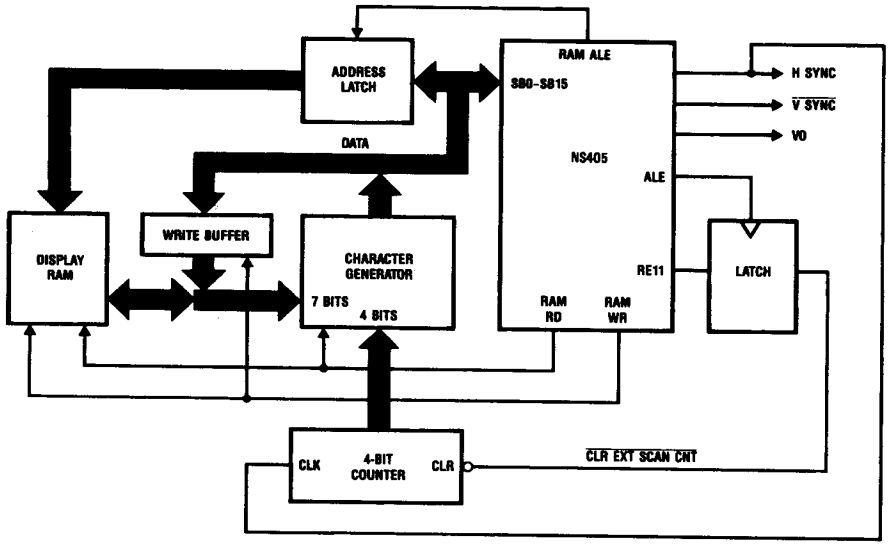
2.7.1 External Character Generation

The chip may be used with an external character generator by switching over to a pixel graphic display mode with modified address stepping as controlled by VCR bits 6, 7. In this mode an external character generator supplies pixel data to the chip as depicted in *Figure 21*. Character addressing comes from the display memory and scan line stepping from a 4-bit counter clocked by the Horizontal Sync. Scan line synchronization is achieved by using the Scan Count Clear signal coming out on RE11, pin 36. After the display of a row it pulses low to initialize the scan line counter for the start of a new row. In pixel mode both the character and any spacing between characters must be encoded into the external character generator. In addition, the chip will access and use at most 8 bits of pixel data for each character cell. However, if the cell width is specified to be 9 or 10, the ninth and tenth dots will repeat what was coded into the first. Therefore, assuming at least one dot spacing between characters, external fonts can at most be seven dots wide.

No limitations apply to the height of a character as long as the external generator can supply all of the scan lines as specified by the CSHR. As in regular pixel mode the LSB brought in is the first dot put out.

Since the eighth data bit is used for character generation it cannot effectively be used for internal attribute latch selection although one of the latches will be selected every data byte. Therefore, both internal attribute latches must be loaded with the same values. If external attribute operation is specified the full 8-bit high order attribute field is available for usage.

2.0 Functional Description (Continued)

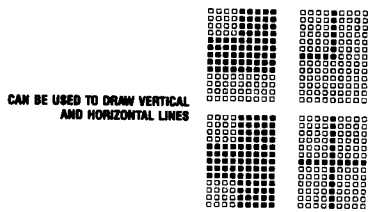


TL/DD/5526–30

FIGURE 21. External Character Set Implementation

2.8 BLOCK GRAPHICS

Block graphics is an alternative display mode to normal alphanumerics which is selected through attribute bit 7. Example *(Figure 22)*. It can operate on a character cell by character cell basis (see Attributes) and words by rerouting display memory bytes through the Block graphics logic instead of the internal character generator.



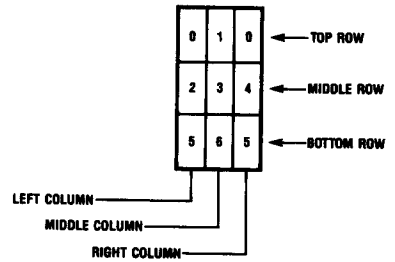
TL/DD/5526–31

FIGURE 22. Example Block Graphics Display Patterns

The Graphics Logic operates by partitioning the character cell space into nine possible areas as shown in *Figure 23* and then using the seven lower bits in the display data byte to turn these areas on or off. In this way one can draw contiguous lines or simple geometric figures while at the same time displaying alphanumeric characters in other cells.

The partitioning of the cell is controlled by two timing chain registers which specify two Horizontal and two Vertical cut off points to the graphics logic. Through these two registers one can make the sections as large or as small as desired, even eliminating sections entirely. Note that data bits 0 and 5 each control two sections as depicted in *Figure 23*.

2.8.1 Graphics Partitioning



TL/DD/5526–32

FIGURE 23. Block Graphics Cell Partitioning

The registers defining the graphics areas function as follows:

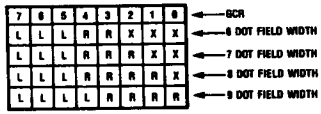
The Graphics Row Register — 8 bits (GRR) is divided into the following two (2) registers:

- Graphics Middle Row, (GMR):
  Defines the scan count at which the middle row begins (4 most significant bits of GRR).
- Graphics Bottom Row, (GBR):
  Defines the scan count at which the bottom row begins (4 least significant bits of GRR).

See *Figure 24.1a* for row example.

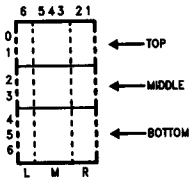
2.0 Functional Description (Continued)

The Graphics Column Register — 8 bits (GCR) controls vertical partitioning through bit patterns as follows: (See *Figure 24.*)



TL/DD/5526-33

FIGURE 24. Block Graphics Column Partitioning



TL/DD/5526-44

GRR = 24
GCR = 60 (0110 0XXX)
cell size = 6 x 7

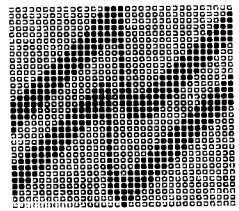
FIGURE 24.1a Block Graphics Example

For all bits in the Graphics Column Register, a one assigns that bit position to the middle column. A zero in an L bit position assigns that bit position to the left column. A zero in an R bit position assigns that bit position to the right column. There is always at least one middle dot although the left and right sections may be eliminated entirely. For 10 dot wide cells the 10th bit will repeat the 9th bit. An easy way to determine the column partitioning is to fill the GCR with all ones, thereby making it one large middle section. Then, starting from the outermost L and R bit positions, put zeros in until the left and right sections are the sizes needed.

2.9 PIXEL GRAPHICS

When bits 6 and 7 of the Video Control Register are both set to 1, the character generator and block graphics circuits are disabled. Video output directly reflects the contents of the display memory byte on a pixel (dot) per bit basis with data output LSB first. Example *(Figure 25).*

Nine bits at a time are accessed from each video memory location with as many bits being used as defined in the character cell width specification. If a cell width of 10 is specified



TL/DD/5526-34

FIGURE 25. Example Pixel Graphics

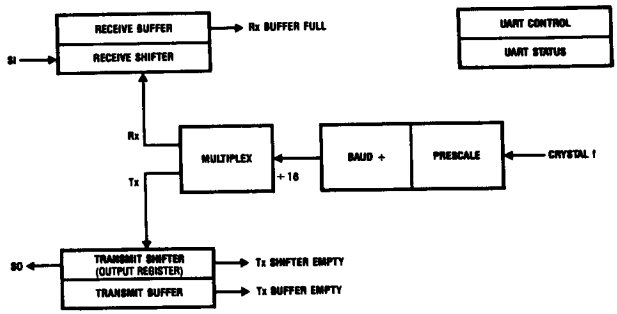
the 10th bit will merely repeat the 9th bit. Attributes are still operable in pixel mode, on a data byte basis, with internal and external operation possible. With internal attribute latch operation the same values must be loaded into both latches since the usual latch select bit is now being used for pixel control. Unless, however, only a 7 dot wide cell is used leaving the 8th bit free. With external attribute operation we are now limited to a 7-bit attribute field since pixel data can now occupy 9 of the 16 bus bits. Because of this the LSB attribute, Reverse Video is totally disabled from operation in Pixel Graphic mode. This also applies to internal attribute latch operation. Note, however, that reverse entire screen video is still operable. Address sequencing through the video memory is sequential with as many data bytes being read in as is necessary to satisfy the pixel requirements of the screen.

2.10 LIGHT PEN

Activation of the light pen interrupt causes the horizontal and vertical screen position of the currently displayed character to be latched into the Horizontal Light Pen Register HPEN (7 bits) and Vertical Light Pen Register VPEN (5 bits) respectively. Both HPEN and VPEN may be read into the CPU accumulator. The values latched remain in VPEN and HPEN until another light pen interrupt latches new values.

2.11 UART

The UART features full duplex operation with double buffered Receive and Transmit sections. Baud rate generation is fully programmable through a 2-stage divider chain. CPU control of the UART is extensive with polled or interrupt driven operation possible.



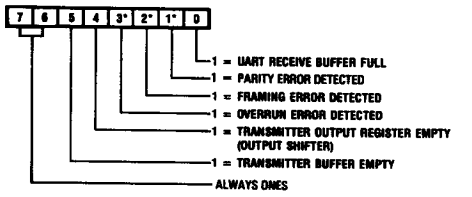
TL/DD/5526-35

FIGURE 26. TMP UART Block Diagram

2.0 Functional Description (Continued)

2.11.1 UART Control

UART Status Register (STAT): Contains error and status bits which reflect the internal state of the UART. Read into CPU accumulator. Bits 0, 5 are the same as those found in the internal interrupt register.



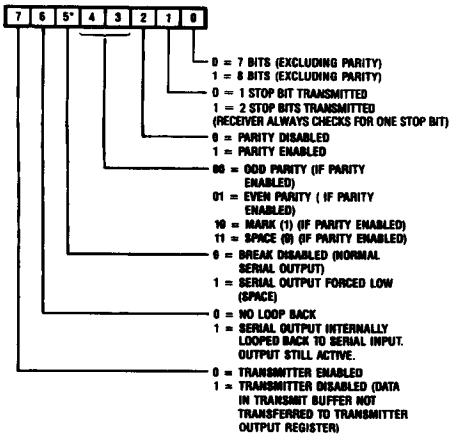
TL/DD/5526–36

UART Status Register bits 1, 2, 3 are only cleared on a chip reset or a read of the UART Receive Buffer. If another word were to come in before the Receive Buffer could be read the errors associated with the new word would add to those already present. The receipt of a new word can cause the three bits to go from a 0 to a 1, but not from a 1 to a 0.

FIGURE 27. UART Status Register

Note: The Transmit Output Register Empty flag is set to one whenever the transmitter is idle. The flag is reset to zero when a data character is transferred from the Transmit Buffer to the Output Register. This transfer does not occur until the next rising edge of the internal UART Transmit Clock. The Transmitter Output Register Empty flag occurs at the beginning of the last stop bit.

UART Control Register (UCR): Contains control bits which configure the format of transmitted data and tests made upon received data. Written to from CPU accumulator.



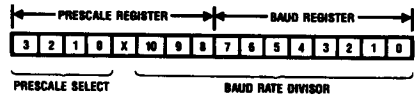
TL/DD/5526–37

*Bit 5 set to 0 by RESET.

FIGURE 28. UART Control Register

2.11.2 Baud Clock Generation

The basic BAUD clock is derived from the crystal frequency through a two-stage divider chain consisting of a 3.5–11 prescale and an 11-bit binary counter. *(Figure 29)*. The divide factors are specified through 2 write only registers shown in *Figure 30*. Note that the 11-bit Baud Rate Divisor spills over into the Prescale Select Register. The correspondences between the 4-bit Prescale Select and Prescale factors is shown in Table I. There are many ways to calculate the two divisor factors but one particularly effective method would be to try to achieve a 1.8432 MHz frequency coming out of the first stage then use the BAUD Rate Divisor factors shown in Table II.



TL/DD/5526–39

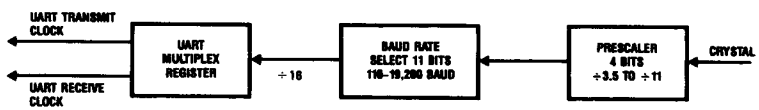
FIGURE 30. UART BAUD Clock Divider Registers

TABLE I. Prescale Factors

Prescale Select	Prescale Factor
0000	3.5
0001	4
0010	4.5
0011	5
0100	5.5
0101	6
0110	6.5
0111	7
1000	7.5
1001	8
1010	8.5
1011	9
1100	9.5
1101	10
1110	10.5
1111	11

TABLE II. Baud Rate Divisors (1.8432 MHz Input)

Baud Rate	Baud Rate Divisor (N − 1)
110 (110.03)	1046
134.5 (134.58)	855
150	767
300	383
600	191
1200	95
1800	63
2400	47
3600	31
4800	23
7200	15
9600	11
19200	5



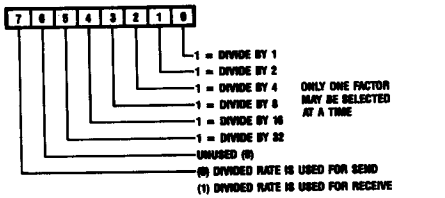
TL/DD/5526–38

FIGURE 29. UART BAUD Clock Generation

2.0 Functional Description (Continued)

The frequency coming out of the BAUD Rate Divisor is then passed through the UART Multiplex Register. Through the UART Multiplex Register one can specify that the Transmitter or Receiver clock be the same or a power of two multiple of the other.

UART Multiplex Register (UMX): Contains the bits which determine the divisor which is used to count down from the primary baud rate when different rates are used for send and receive (eight bits).



TL/DD/5526–40

FIGURE 31. UART Multiplex Register

The actual baud rate may be found from:

$BR = Fc/(16*N*P*D)$

Where:

BR is the Baud Rate

Fc is the external crystal frequency

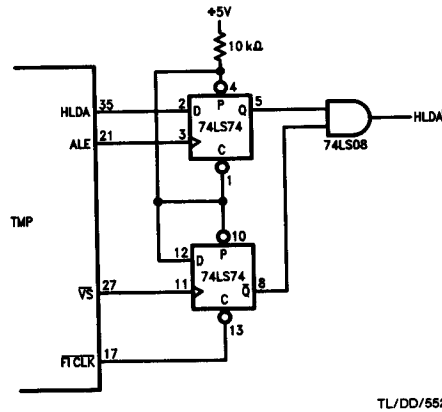
N is one plus the value of the Baud Rate Divisor contained in the Baud Rate Select Register and the Prescale Select Register.

P is the Prescale Divide Factor Selected by the value in the Prescale Select Register.

D is the Multiplex Register Divide Factor

3.0 Slave Processing

The TMP may be used as a slave video controller by having a host system perform Direct Memory Accesses into the display RAM. To assist in implementing such a system the chip features two DMA control pins—HOLD (Hold Request) and HLDA (Hold Acknowledge). These two signals come out on shared ROM Expand Bus pins RE8 and RE12. To request a DMA access a host would activate HOLD (active high and await the acknowledging HLDA from the TMP before proceeding with the DMA. The TMP only allows DMA operations during the vertical blanking period and will activate HLDA in response to a HOLD shortly after vertical blanking starts. In DMA mode all 16 TMP System Bus drivers are tri-stated while the bus control signals RAM ALE, RAM RD, RAM WR go to their inactive (high) states. A HOLD request must arrive two CPU cycles before vertical blanking starts; otherwise it will miss that retrace cycle and will have to wait until the next one, one frame later. Once DMA mode is entered, it is maintained for the duration of vertical blanking regardless of the state of HOLD. Near the end of vertical blanking the DMA mode will terminate in preparation for the display of the next frame, but the HLDA will NOT turn off. Specifically, this will occur one scan time before the end of vertical blanking. It is up to the designer to be sure that the host is off the BUS before this happens or suffer bus contention with the video controller. He can do this by either predetermining the length of time the host has to remain on the bus, or by using the end of vertical sync (as shown in *Figure 32*) to signal the end of a safe DMA period. If during DMA the CPU attempts to do a display memory access it would be put into a wait state until DMA is concluded and normal memory accessing is resumed.



TL/DD/5526–45

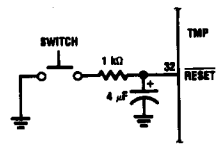
Vertical sync should be programmed to end as late as possible, but must end at least one scan time before the end of vertical blanking.

FIGURE 32

4.0 Reset

The TMP will reset if the $\overline{\text{RESET}}$ (32) pin is held at a logic low (< 0.8V) for at least five CPU cycle times. This pre-supposes that the V_{CC} is up, stable and within operational limits (+5V ±10%) and that the oscillator is running. For a power on reset, time must be allowed for the power supplies to stabilize (typically 50 ms) and the oscillator to start up. If power supply noise or ripple causes V_{CC} to exceed the +5V ±10% limits neither reset nor operation is guaranteed.

Internally, the $\overline{\text{RESET}}$ pin has a depletion load pullup that typically acts as a 30 μA current source from V_{CC} in the voltage range of interest. A typical reset circuit with a 0.5 second reset pulse is shown in *Figure 33*.



TL/DD/5526–41

FIGURE 33. Typical Reset Circuit

4.0 Reset (Continued)

During RESET a number of internal registers are initialized as follows:

4.1 CPU

CPU Clock divide = 1.5 (SCR bit 0 = 1)
Shared VIDCLK/$\overline{\text{FI CLK}}$ = 0 (SCR bit 7 = 0, $\overline{\text{FI CLK}}$ gated to external pin)
Program Counter = 0
Stack Pointer = 0
Program Memory Bank = 0
RAM Register Bank = 0
Timer Stopped
Instruction Register cleared
F0 and F1 cleared

4.2 INTERRUPTS

Internal and External Interrupts disabled
Internal Interrupt Register set to 000011X0

4.3 UART

Receiver initialized to look for start bit
Status Register set to 11110000
Transmitter initialized to wait for OUT XMTR instruction
Control Register bit 5 = 0 (No BREAK)

4.4 VIDEO

Video generation shutdown (VCR bit 5 = 0)
FIFO Cleared Out
Timing Chain Character Counter = 0
Timing Chain Scan Counter = 0
Timing Chain Row Counter = 0
Timing Chain Blink Counter = 0
} IN TEST MODE ONLY

4.5 PIN STATES AT RESET

Pins 1–8 (SB0–7)	In TRI-STATE during reset and until either the CPU executes a MOVX instruction or bit 5 of the VCR is set.
Pins 9–16 (SB8–15)	If bit 4 of the SCR is set, SB8–15 will behave like SB0–7. If bit 4 of the SCR is cleared, SB8–15 will act as outputs (any of which may be either high or low). Note that bit 4 of the SCR may be one or zero at power-up.
Pin 17 (VID CLK/$\overline{\text{FI CLK}}$)	High during reset and until bit 5 of the VCR is set.
Pin 18 (RAM ALE)	High during reset and until the CPU executes a MOVX instruction or bit 5 of the VCR is set.
Pin 19 ($\overline{\text{RAM WR}}$)	High during reset and until the CPU executes a MOVX (of the output to display RAM variety) instruction.
Pin 20 ($\overline{\text{RAM RD}}$)	High during reset and until either the CPU executes a MOVX instruction or bit 5 of the VCR is set.
Pin 21 (ALE)	Pulses continuously.
Pin 22 (XTAL 2)	Crystal input or master clock input.
Pin 23 (XTAL 1)	Crystal input.
Pin 24 (Gnd.)	
Pin 25 ($\overline{\text{INTENS}}$/$\overline{\text{FO CLK}}$)	May be either high or low during reset.
Pin 26 (VO)	Low (because of asserted blanking signals) from reset until bit 5 of the VCR is set.
Pin 27 ($\overline{\text{VS}}$)	In TRI-STATE mode upon RESET, enabled when bit 5 of the VCR is set.
Pin 28 (HS)	Low from reset until bit 5 of the VCR is set.
Pin 29 (EA)	Input only. (must be tied HIGH (V_{IH_2}))

4.0 Reset (Continued)

Pin 30 ($\overline{PSEN}$)	Active during reset.
Pin 31 ($\overline{RD}$)	High during reset and until an IN PORT instruction is executed.
Pin 32 ($\overline{RESET}$)	Input only.
Pin 33 (SO)	High during reset and until an OUT XMTR instruction is executed.
Pin 34 (SI)	Input only.
Pin 35 (RE12/HLDA)	If HOLD is low: low during reset. If HOLD is high: low at falling edge of ALE and during $\overline{PSEN}$, may be low or high at rising edge of ALE.
Pin 36 (RE11/$\overline{SC\ CLR}$)	If reset asserted: low at falling edge of ALE and during $\overline{PSEN}$, sampled value of internal Scan Count Clear signal is output at rising edge of ALE.
Pin 37 (RE10/$\overline{INTR}$) Pin 38 (RE9/$\overline{LPEN}$) Pin 39 (RE8/HLDR)	If reset asserted: low at falling edge of ALE and during $\overline{PSEN}$. Always in TRI-STATE at rising edge of ALE.
Pins 40–47 (RE0–7; I/O0–7)	If reset asserted: low at falling edge of ALE, in TRI-STATE during $\overline{PSEN}$, and may be either high or low at the rising edge of ALE.
Pin 48 (V_{CC})	

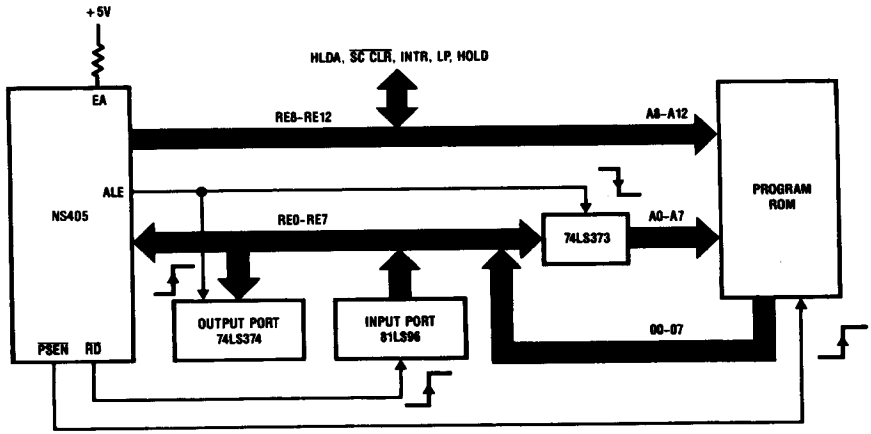
5.0 Extra Attributes

One may want to expand the external attribute field by adding more bits so that functions such as color (Red—Green—Blue drive) or grey scale may be implemented. Like the eight attributes which the chip handles internally these extra attributes would operate on a character cell basis. To add attribute bits one would have to duplicate the internal 4 level character/attribute FIFO externally using fast MSI chips. To assist in handling the external FIFO circuitry the TMP features two FIFO clocking signals on pins 17 and 25. The FIFO IN Clock ($\overline{FI\ CLK}$) is used to strobe attribute data into the external FIFO circuits in synchronism with the internal TMP FIFO. Its timing is identical to $\overline{RAM\ RD}$ but is only active when the video does a display RAM read to load its FIFO. The FIFO OUT Clock ($\overline{FO\ CLK}$) pulses for 1–3 bit times each time the video starts the display of a new character cell. The external FIFO would use the rising edge of this signal to clock out or latch the attribute output.

In order for the TMP CPU to access the additional attribute bits special bus gating arrangements would have to be worked out on the System Bus (Video Data Bus is at most 16 bits wide). Unless one were to run with internal attributes or only use a few of the external attributes in which case the unused bits could be used with the external FIFO. Whenever using the $\overline{FO\ CLK}$ the Intensity attribute is disabled since they both share the same pin.

6.0 TMP BUS Interfacing

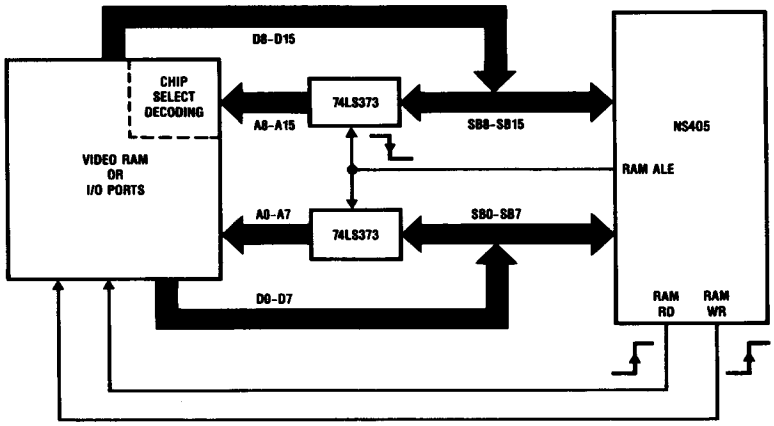
The two external buses on the TMP, ROM Expand and System are easily interfaced to as shown in *Figures 34* and *35*. Important bus information output from the chip is latched using the rising or falling edges of the various control signals. I/O port information is read in through a TRI-STATE® buffer chip such as an 81LS96.



TL/DD/5526–42

FIGURE 34. TMP ROM Expand BUS

6.0 TMP BUS Interfacing (Continued)



TL/DD/5526-43

FIGURE 35. TMP System Bus

TMP Registers (Excluding Timing Chain Registers)

TMP Registers		Associated Intructions			
		CPU SECTION			
A	= Accumulator — 8 bits	ADD A,Rr	MOV A,Rr	XCH A,Rr	
#data	= data immediate	ADD A,#data	MOV A,@Rr	XCH A,@Rr	
Rr	= Register	ADD A,@Rr	MOV A,#data	XCHD A,@Rr	
@Rr	= Register pointed to by R0 or R1	ADDC A, Rr	MOV Rr,A	XRL A,Rr	
		ADDC A,#data	MOV Rr,#data	XRL A,@Rr	
		ADDC A,@Rr	MOV @Rr,A	XRL A,#data	
		ANL A,Rr	MOV @Rr,#data	JBn addr	
		ANL A,#data	MOVP A,@A	JNZ addr	
		ANL A,@Rr	MOVP3 A,@A	JZ addr	
		CLR A	RL A	DJNZ Rr,addr	
		CPL A	RLC A		
		DAA	RR A		
		DEC A	RRC A		
		DEC Rr	ORL A,Rr		
		INC A	ORL A,@Rr		
		INC Rr	ORL A,#data		
		INC @Rr	SWAP A		
*HACC	= High Accumulator — 8 bits	*MOV A,HACC	*MOV HACC,A		
C	= Carry Bit	CLR C	CPL C	JNC addr	JC addr
*LONG R0	= Register Pair, R0, RA	*DECL R0		*INCL R0	*MOVL A,R0
		*MOVL R0,A		*MOVX A,@R0	*MOVX @R0,A
*LONG R1	= Register Pair R1, RB	*DECL R1		*INCL R1	*MOVL A,R1
		*MOVL R1,A		*MOVX A,@R1	*MOVX @R1,A
T	= Timer — 8 bits	MOV A,T		MOV T,A	STOP T
		STRT T		*JNTF addr	JTF addr
F0	= Flag 0	CLR F0	CPL F0	JF0 addr	*JNF0 addr
F1	= Flag 1	CLR F1	CPL F1	JF1 addr	*JNF1 addr
INTR	= Interrupt Register — 8 bits	MOV A,INTR		JNXI addr	JXI addr
		*DIS II		DIS XI	*EN II
		EN XI			

TMP Registers (Excluding Timing Chain Registers) (Continued)

TMP Registers		Associated Instructions		
CPU SECTION (Continued)				
MASK	= Internal Interrupt MasK — 8 bits	*MOV MASK,A		
PSW	= Program Status Word — 8 bits	MOV A,PSW	MOV PSW,A	
PORT	= 8 bit I/O Port	ANL PORT,#data	IN PORT	
		ORL PORT,#data	OUT PORT	
Miscellaneous Instructions		CALL addr	JMP addr	JMPP @A
		NOP	RET	RETR
		SEL MB0	SEL MB1	*SEL MB2
		*SEL MB3	SEL RB0	SEL RB1
VIDEO MANAGEMENT				
		Associated Instructions		
SCR	= System Control Register — 8 bits		*MOV SCR,A	
VCR	= Video Control Register — 8 bits		*MOV VCR,A	
HOME	= Home Address Register — 16 bits		*MOV A,HOME	*MOV HOME,A
CURS	= Cursor Address Register — 16 bits	*DEC CURS	*INC CURS	*MOVX A,@CURS
		*MOV CURS,A	*MOV A,CURS	*MOVX @CURS,A
BEGD	= Beginning of Display RAM Register — 16 bits		*MOV BEGD,A	
ENDD	= End of Display RAM Register — 16 bits		*MOV ENDD,A	
SROW	= Status Row Register — 16 bits		*MOV SROW,A	
AL0	= Attribute Latch 0 — 8 bits		*MOV AL0,A	
AL1	= Attribute Latch 1 — 8 bits		*MOV AL1,A	
HPEN	= Horizontal Light Pen Register — 7 bits		*MOV A,HPEN	
VPEN	= Vertical Light Pen Register — 5 bits		*MOV A,VPEN	
VINT	= Vertical Interrupt Register — 5 bits		*MOV VINT,A	
UART CONTROL				
PSR	= Prescale Register (UART) — 8 bits		*MOV PSR,A	
BAUD	= Baud Rate Select Register — 8 bits		*MOV BAUD,A	
UCR	= UART Control Register — 8 bits		*MOV UCR,A	
UMX	= UART Multiplex Register — 8 bits		*MOV UMX,A	
STAT	= Status Latch (UART) — 6 bits		*MOV A,STAT	
RCVR	= UART Receive Buffer — 8 bits		*IN RCVR	
XMTR	= UART Transmit Buffer — 8 bits		*OUT XMTR	
TCP	= Timing Chain Pointer		*MOV TCP,A	
@TCP	= Register Pointed to by TCP		*MOV @TCP,A	

*New instruction added to 8048 subset.

Symbol Definitions

Symbol	Definition
AC	Auxiliary Carry Flag
addr	Program Memory Address
b	Bit Designator (b = 0 − 7)
BS	RAM Bank Switch
data	Number or Expression (8 bits)
DBF	Program Memory Bank Select Bits (2)
EXI	External Interrupt Pin
F0, F1	Internal Flags
P	I/O Port (8 bits)

Symbol	Definition
PC	Program Counter
SP	Stack Pointer
TF	Timer Flag
#	Prefix for Immediate Data
@	Prefix for Indirect Address
( )	Contents of Register
(( ))	Contents of Memory Location pointed to by designated register
←	Replaced by

Instruction Set

Mnemonic	Machine Code	Function	Description	Cycles	Bytes	Flags				
						C	AC	HACC	F0	F1
ADD A, Rr	0 1 1 0 1 r r r	(A) ← (A) + (Rr) for r= 0 − 7	Add contents of designated register to the Accumulator (8-bit operation)	1	1	*	*	*		
ADD A, #data	0 0 0 0 0 0 1 1 d7 d6 d5 d4 d3 d2 d1 d0	(A) ← (A) + data	Add immediate the specified data to the Accumulator (8-bit operation)	2	2	*	*	*		
ADD A, @ Rr	0 1 1 0 0 0 0 r	(A) ← (A) + ((Rr)) for r = 0 − 1	Add indirect the contents of data memory pointed to by Rr to the Accumulator (8-bit operation)	1	1	*	*	*		
ADDC A, Rr	0 1 1 1 1 r r r	(A) ← (A) + (C) + (Rr) for r = 0 − 7	Add with carry the contents of the designated register to the Accumulator (8-bit operation)	1	1	*	*	*		
ADDC A, # data	0 0 0 1 0 0 1 1 d7 d6 d5 d4 d3 d2 d1 d0	(A) ← (A) + (C) + data	Add immediate with carry the specified data to the Accumulator (8-bit operation)	2	2	*	*	*		
ADDC A, @ Rr	0 1 1 1 0 0 0 r	(A) ← (A) + (C) + ((Rr)) for r = 0 − 1	Add indirect with carry the contents of data memory pointed to by Rr to the Accumulator (8-bit operation)	1	1	*	*	*		
ANL A, Rr	0 1 0 1 1 r r r	(A) ← (A) AND (Rr) for r = 0 − 7	Logical AND contents of designated register with Accumulator (8-bit operation)	1	1					
ANL A, # data	0 1 0 1 0 0 1 1 d7 d6 d5 d4 d3 d2 d1 d0	(A) ← (A) AND data	Logical AND specified Immediate Data with Accumulator (8-bit operation)	2	2					
ANL A, @ Rr	0 1 0 1 0 0 0 r	(A) ← (A) AND ((Rr)) for r = 0 − 1	Logical AND indirect the contents of data memory pointed to by Rr with Accumulator (8-bit operation)	1	1					
ANL PORT, # data	0 1 1 1 0 0 1 1 d7 d6 d5 d4 d3 d2 d1 d0	(P) ← (P) AND data	Logical AND immediate specified data with output port (8-bit operation)	2	2					
CALL addr	a10 a9 a8 1 0 1 0 0 a7 a6 a5 a4 a3 a2 a1 a0	((SP)) ← (PC0−12) ((SP)) ← (PSW3−7) (SP) ← (SP) + 1 (PC8−10) ← addr 8−10 (PC0−7) ← addr 0−7 (PC11−12 ← DBF 0, 1	Call designated subroutine	2	2					

Instruction Set (Continued)

Mnemonic	Machine Code	Function	Description	Cycles	Bytes	Flags				
						C	AC	HACC	F0	F1
CLR A	0 0 1 0 0 1 1 1	(A) ← 0	Clear the Accumulator	1	1					
CLR C	1 0 0 1 0 1 1 1	(C) ← 0	Clear carry bit	1	1	*				
CLR F0	1 0 0 0 0 1 0 1	(FO) ← 0	Clear Flag 0	1	1				*	
CLR F1	1 0 1 0 0 1 0 1	(F1) ← 0	Clear Flag 1	1	1					*
CPL A	0 0 1 1 0 1 1 1	(A) ← NOT (A)	Complement the contents of the Accumulator (8-bit operation)	1	1					
CPL C	1 0 1 0 0 1 1 1	(C) ← NOT (C)	Complement carry bit	1	1	*				
CPL F0	1 0 0 1 0 1 0 1	(F0) ← NOT (F0)	Complement Flag 0	1	1				*	
CPL F1	1 0 1 1 0 1 0 1	(F1) ← NOT (F1)	Complement Flag 1	1	1					*
DA A	0 1 0 1 0 1 1 1		Decimal Adjust the contents of the Accumulator (8-bit operation)	1	1	*	*			
DEC A	0 0 0 0 0 1 1 1	(HACC, A) ← (HACC, A) − 1	Decrement by 1 the contents of HACC/ ACC	1	1	*		*		
DEC CURS	0 0 0 0 1 0 1 0	(CURS) ← (CURS) − 1	Decrement by 1 the contents of the Cursor Address Register	1	1					
DEC Rr	1 1 0 0 1 r r r	(Rr) ← (Rr) − 1	Decrement by 1 the contents of the designated register (8-bit operation)	1	1	*				
DECL Rr	0 0 0 0 1 0 0 r	(Rr) ← (Rr) − 1 for r = 0 − 1	Decrement by 1 the contents of the designated 16-bit register pair	1	1					
DIS II	0 0 1 1 0 1 0 1		Disable internal interrupts	1	1					
DIS XI	0 0 0 1 0 1 0 1		Disable external interrupts	1	1					

DJNZ Rr, addr	1 1 1 0 1 r r r
	a7 a6 a5 a4 a3 a2 a1 a0
		(Rr) ← (Rr) − 1 for r = 0 − 7
		If (Rr) ≠ 0 do (PC0−7) ← addr
		If (Rr) = 0 do (PC) ← PC + 2
			Decrement the specified register and Jump if not zero to designated address within page (8-bit decrement)	2	2					

EN II	0 0 1 0 0 1 0 1		Enable internal interrupts.	1	1					
EN XI	0 0 0 0 0 1 0 1		Enable external interrupt.	1	1					
INC A	0 0 0 1 0 1 1 1	(HACC, A) ← (HACC, A) + 1	Increment by 1 the contents of HACC/A.	1	1	*		*		
INC CURS	0 0 1 1 1 0 1 0	(CURS) ← (CURS) + 1	Increment by 1 the contents of the Cursor Address Register.	1	1					

Instruction Set (Continued)

Mnemonic	Machine Code								Function	Description	Cycles	Bytes	Flags				
													C	AC	HACC	F0	F1
INC Rr	0	0	0	1	1	r	r	r	(Rr) ← (Rr) + 1 for r = 0 − 7	Increment by 1 the contents of the designated register (8-bit increment)	1	1	*				
INC @ Rr	0	0	0	1	0	0	0	r	((Rr)) ← ((Rr)) + 1 for r = 0 − 1	Increment in direct the contents of data memory pointed to by Rr (8-bit increment)	1	1	*				
INCL Rr	0	0	1	1	1	0	0	r	(Rr) ← (Rr) + 1 for r = 0 − 1	Increment by 1 the contents of the designated 16-bit register pair	1	1					
IN PORT	1	1	1	0	0	0	0	1	(A) ← (P)	Input data from port into Accumulator (8-bit transfer)	2	1					
IN RCVR	1	1	1	0	0	0	0	0	(A) ← (RCVR)	Input contents of UART Receive buffer into Accumulator (8-bit transfer). Also, clears Receive Buffer Full interrupt.	1	1					
JBb addr	b2	b1	b0	1	0	0	1	0	(PC0–7) ← addr if (b) = 1 (PC) ← (PC) + 2 if (b) = 0 for b = 0 − 7	Jump to specified address within page if Accumulator bit is set	2	2					
	a7	a6	a5	a4	a3	a2	a1	a0									
JC addr	1	1	1	1	0	1	1	0	(PC0–7) ← addr if C = 1 (PC) ← (PC) + 2 if C = 0	Jump to specified address within page if Carry flag is set	2	2					
	a7	a6	a5	a4	a3	a2	a1	a0									
JF0 addr	1	0	0	1	0	1	1	0	(PC0–7) ← addr if F0 = 1 (PC) ← (PC) + 2 if F0 = 0	Jump to specified address within page if Flag F0 is set	2	2					
	a7	a6	a5	a4	a3	a2	a1	a0									
JF1 addr	0	1	1	1	0	1	1	0	(PC0–7) ← addr if F1 = 1 (PC) ← (PC) + 2 if F1 = 0	Jump to specified address within page if Flag F1 is set	2	2					
	a7	a6	a5	a4	a3	a2	a1	a0									
JMP addr	a10	a9	a8	0	0	1	0	0	(PC8–10) ← addr 8–10 (PC0–7) ← addr 0–7 (PC11–12) ← DBF 0, 1	Direct Jump to specified address within 2k Bank	2	2					
	a7	a6	a5	a4	a3	a2	a1	a0									
JMPP @ A	1	0	1	0	0	0	1	1	(PC0–7) ← ((A))	Jump indirect within page to the address specified in the memory location pointed to by the Accumulator	2	1					
JNC addr	1	1	1	0	0	1	1	0	(PC0–7) ← addr if C = 0 (PC) ← (PC) + 2 if C = 1	Jump within page to specified address if Carry flag is 0	2	2					
	a7	a6	a5	a4	a3	a2	a1	a0									

Instruction Set (Continued)

Mnemonic	Machine Code	Function	Description	Cycles	Bytes	Flags				
						C	AC	HACC	F0	F1
JNF0 addr	1 0 0 0 0 1 1 0 a7 a6 a5 a4 a3 a2 a1 a0	(PC0–7) ← addr if F0 = 0 (PC) ← (PC) + 2 if F0 = 1	Jump within page to specified address if F0 is 0	2	2					
JNF1 addr	0 1 1 0 0 1 1 0 a7 a6 a5 a4 a3 a2 a1 a0	(PC0–7) ← addr if F1 = 0 (PC) ← (PC) + 2 if F1 = 1	Jump within page to specified address if F1 is 0	2	2					
JNTF addr	0 0 0 0 0 1 1 0 a7 a6 a5 a4 a3 a2 a1 a0	(PC0–7) ← addr if TF = 0 (PC) ← (PC) + 2 if TF = 1, (TF) ← 0	Jump within page to specified address if Timer flag is reset. If not, continue and reset TF	2	2					
JNXI addr	1 0 1 0 0 1 1 0 a7 a6 a5 a4 a3 a2 a1 a0	(PC0–7) ← addr if EXI = LOW (PC) ← (PC) + 2 if EXI = HIGH	Jump within page to specified address if External Interrupt pin is LOW	2	2					
JNZ addr	1 1 0 1 0 1 1 0 a7 a6 a5 a4 a3 a2 a1 a0	(PC0–7) ← addr if A ≠ 0 (PC) ← (PC) + 2 if A = 0	Jump within page to specified address if Accumulator is not 0	2	2					
JTF addr	0 0 0 1 0 1 1 0 a7 a6 a5 a4 a3 a2 a1 a0	(PC0–7) ← addr if TF = 1, (TF) ← 0 (PC) ← (PC) + 2 if TF = 0	Jump within page to specified address if Timer flag is set. If jump taken Timer flag reset	2	2					
JXI addr	1 0 1 1 0 1 1 0 a7 a6 a5 a4 a3 a2 a1 a0	(PC0–7) ← addr if EXI = HIGH (PC) ← (PC) + 2 if EXI = LOW	Jump within page to specified address if External Interrupt pin is HIGH	2	2					
JZ addr	1 1 0 0 0 1 1 0 a7 a6 a5 a4 a3 a2 a1 a0	(PC0–7) ← addr if A = 0 (PC) ← (PC) + 2 if A ≠ 0	Jump within page to specified address if Accumulator is 0	2	2					
MOV A, CURS	1 0 0 1 1 0 1 1	(HACC/A) ← (CURS)	Copy the contents of the Cursor Address Register into the HACC/A (16-bit transfer)	1	1			*		
MOV A, HACC	1 1 1 0 0 0 1 0	(A) ← (HACC)	Copy contents of the High Accumulator into the Low Accumulator (8-bit transfer)	1	1					
MOV A, HOME	1 0 0 1 1 0 1 0	(HACC/A) ← (HOME)	Copy the contents of the Home Address register into the HACC/A (16-bit transfer)	1	1			*		
MOV A, HPEN	0 0 1 1 1 1 1 1	(A0–6) ← (HPEN) (A7) ← 0	Copy the contents of the Horizontal Light Pen Register into the Accumulator (7-bit transfer, A7 cleared)	1	1					

Instruction Set (Continued)

Mnemonic	Machine Code	Function	Description	Cycles	Bytes	Flags				
						C	AC	HACC	F0	F1
MOV A, INTR	1 0 0 0 1 1 0 0	(A) ← (INTR)	Copy the contents of the Interrupt Register into the Accumulator (8-bit transfer)	1	1					
MOV A, PSW	1 1 0 0 0 1 1 1	(A) ← (PSW)	Copy contents of the Program Status word into the Accumulator (8-bit transfer)	1	1					
MOV A, Rr	1 1 1 1 1 r r r	(A) ← (Rr) for r = 0 − 7	Copy the contents of the designated Register into the Accumulator (8-bit transfer)							
MOV A, STAT	1 0 0 1 1 1 0 0	(A0–5) ← (STAT) (A6–7) ← 11	Copy the contents of the UART Status Latch into the Accumulator (6-bit transfer, A6 and A7 set)	1	1					
MOV A, T	0 1 0 0 0 0 1 0	(A) ← (T)	Copy the contents of the Timer into the Accumulator (8-bit transfer)	1	1					
MOV A, VPEN	0 0 1 1 1 1 1 0	(A0–4) ← (VPEN) (A5–7) ← O	Copy contents of the Vertical Light Pen Register into the Accumulator (5-bit transfer, A5–A7 cleared)	1	1					
MOV A, @ Rr	1 1 1 1 0 0 0 r	(A) ← ((Rr)) for r = 0 − 1	Copy indirect the contents of data memory pointed to by Rr into the Accumulator (8-bit transfer)	1	1					
MOV A, # data	0 0 1 0 0 0 1 1 d7 d6 d5 d4 d3 d2 d1 d0	(A) ← data	Load immediate the specified data into the Accumulator (8-bit load)	2	2					
MOV AL0, A	0 0 1 1 1 1 0 0	(AL0) ← (A)	Copy the contents of the Accumulator into Attribute Latch 0 (8-bit transfer)	1	1					
MOV AL1, A	0 0 1 1 1 1 0 1	(AL1) ← (A)	Copy the contents of the Accumulator into Attribute Latch 1 (8-bit transfer)	1	1					
MOV BAUD, A	0 0 0 0 0 0 1 0	(BAUD) ← (A)	Copy the contents of the Accumulator into the UART Baud Rate Select Register (8-bit transfer)	1	1					
MOV BEGD, A	0 0 0 0 1 1 0 1	(BEGD) ← (HACC/A)	Copy the contents of HACC/A into the Beginning of Display RAM Register (16-bit transfer)	1	1					

Instruction Set (Continued)

Mnemonic	Machine Code	Function	Description	Cycles	Bytes	Flags				
						C	AC	HACC	F0	F1
MOV CURS, A	1 0 0 0 1 0 1 1	(CURS) ← (HACC/A)	Copy the contents of HACC/A into the Cursor Address Register (16-bit transfer)	1	1					
MOV ENDD, A	0 0 0 0 1 1 0 0	(ENDD) ← (HACC/A)	Copy the contents of HACC/A into the End of Display RAM Register (16-bit transfer)	1	1					
MOV HACC, A	1 1 0 0 0 0 1 0	(HACC) ← (A)	Copy the contents of the Low Accumulator into the High Accumulator (8-bit transfer)	1	1			*		
MOV HOME, A	1 0 0 0 1 0 1 0	(HOME) ← (HACC/A)	Copy the contents of HACC/A into the Home Address Register (16-bit transfer)	1	1					
MOV MASK, A	1 0 0 0 0 0 1 0	(MASK) ← (A)	Copy the contents of the Accumulator into the Interrupt Mask Register (8-bit transfer)	1	1					
MOV PSR, A	0 0 1 0 0 0 1 0	(PSR) ← (A)	Copy the contents of the Accumulator into the UART Prescale Register (8-bit transfer)	1	1					
MOV PSW, A	1 1 0 1 0 1 1 1	(PSW) ← (A)	Copy contents of the Accumulator into the Program Status Word (8-bit transfer)	1	1	*	*			
MOV Rr, A	1 0 1 0 1 r r r	(Rr) ← (A) for r = 0 − 7	Copy contents of the Accumulator into the designated register (8-bit transfer)	1	1					
MOV SCR, A	0 1 0 1 0 1 0 1	(SCR) ← (A)	Copy contents of the Accumulator into the System Control Register (8-bit transfer)	1	1					
MOV SROW, A	0 0 0 0 1 1 1 0	(SROW) ← (HACC/A)	Copy the contents of HACC/A into the Status Row Register (16-bit transfer)	1	1					
MOV T, A	0 1 1 0 0 0 1 0	(T) ← (A)	Copy the contents of the Accumulator into the Timer (8-bit transfer)	1	1					
MOV TCP, A	1 0 0 0 0 1 1 1	(TCP) ← (A)	Copy the contents of the Accumulator into the Timing Chain Pointer	1	1					

Instruction Set (Continued)

Mnemonic	Machine Code	Function	Description	Cycles	Bytes	Flags				
						C	AC	HACC	F0	F1
MOV UCR, A	0 0 0 0 0 0 0 1	(UCR) ← (A)	Copy the contents of the Accumulator into the UART Control Register (8-bit transfer)	1	1					
MOV VCR, A	0 1 0 0 0 1 0 1	(VCR) ← (A)	Copy the contents of the Accumulator into the Video Control Register (8-bit transfer)	1	1					
MOV VINT, A	1 0 1 0 0 0 1 0	(VINT) ← (A)	Copy the contents of the Accumulator into the Vertical Interrupt Register	1	1					
MOV Rr, # data	1 0 1 1 1 r r r d7 d6 d5 d4 d3 d2 d1 d0	(Rr) ← data for r = 0 − 7	Load immediate the specified data into the designated register (8-bit load)	2	2					
MOV @ Rr, A	1 0 1 0 0 0 0 r	((Rr)) ← (A) for r = 0 − 1	Copy indirect the contents of the Accumulator into the data memory location pointed to by Rr (8-bit transfer)	1	1					
MOV @ Rr, # data	1 0 1 1 0 0 0 r d7 d6 d5 d4 d3 d2 d1 d0	((Rr)) ← data for r = 0 − 1	Load indirect the specified immediate data into the data memory location pointed to by Rr (8-bit load)	2	2					
MOV @ TCP, A	1 0 1 1 0 1 1 1	((TCP)) ← (A) (TCP) ← (TCP) + 1	Copy indirect the contents of the Accumulator into the Timing Chain Register pointed to by TCP. Contents of TCP incremented by 1	1	1					
MOV UMX, A	0 0 1 1 0 0 1 1	(UMX) ← (A)	Copy the contents of the Accumulator into the UART Multiplex Register (8-bit transfer)	1	1					
MOVL A, R0	1 0 0 1 1 0 0 0	(HACC/A) ← (RA, R0)	Copy the contents of RA, R0 into HACC/A (16-bit transfer)	1	1			*		
MOVL A, R1	1 0 0 1 1 0 0 1	(HACC/A) ← (RB, R1)	Copy the contents of RB, R1 into HACC/A (16-bit transfer)	1	1			*		
MOVL R0, A	1 0 0 0 1 0 0 0	(RA, R0) ← (HACC/A)	Copy the contents of HACC/A into RA, R0 (16-bit transfer)	1	1					
MOVL R1, A	1 0 0 0 1 0 0 1	(RB, R1) ← (HACC/A)	Copy the contents of HACC/A into RB, R1 (16-bit transfer)	1	1					

Instruction Set (Continued)

Mnemonic	Machine Code	Function	Description	Cycles	Bytes	Flags				
						C	AC	HACC	F0	F1
MOVP A, @ A	1 0 1 1 0 0 1 1	(PC0–7) ← (A) (A) ← ((PC)) (PC0–7) ← (old PC0–7) + 1	Replace low 8 bits of PC with A. Load indirect within page the contents of the memory location pointed to by new PC into Accumulator. Restore PC with old value plus 1. Operates in all memory banks.	2	1					
MOVP3 A, @ A	1 1 1 1 0 0 1 1	(PC0–7) ← (A) (PC8–10) ← 011 (A) ← ((PC)) (PC) ← (old PC) + 1	Replace low 8 bits of PC with A. Next 3 bits replaced with 011. Load indirect within page 3 the contents of the memory location pointed to by new PC into the Accumulator. Restore PC with old value plus 1. Operates in all memory banks.	2	1					
MOVX A, @ CURS	1 0 0 1 1 1 0 1	(HACC/A) ← ((CURS))	Copy indirect the contents of display memory as pointed to by CURS into HACC/A (16-bit transfer)	Min. 2	1			*		
MOVX A, @ R0	1 0 0 1 0 0 0 0	(HACC/A) ← ((RA, R0))	Copy indirect the contents of display memory as pointed to by RA, R0 into HACC/A (16-bit transfer)	Min. 2	1			*		
MOVX A, @ R1	1 0 0 1 0 0 0 1	(HACC/A) ← ((RB, R1))	Copy indirect the contents of display memory as pointed to by RB, R1 into HACC/A (16-bit transfer)	Min. 2	1			*		
MOVX @ CURS, A	1 0 0 0 1 1 0 1	((CURS)) ← (HACC/A)	Copy indirect the contents of HACC/A into the display memory location as pointed to by CURS (16-bit transfer)	Min. 2	1					
MOVX @ R0, A	1 0 0 0 0 0 0 0	((RA, R0)) ← (HACC/A)	Copy indirect the contents of HACC/A into the display memory location as pointed to by RA, R0 (16-bit transfer)	Min. 2	1					

Instruction Set (Continued)

Mnemonic	Machine Code	Function	Description	Cycles	Bytes	Flags				
						C	AC	HACC	F0	F1
MOVX @ R1, A	1 0 0 0 0 0 0 1	((RB, R1)) ← (HACC/A)	Copy indirect the contents of HACC/A into the display memory location pointed to by RB, R1 (16-bit transfer)	Min. 2	1					
NOP	0 0 0 0 0 0 0 0		No Operation	1	1					
ORL A, Rr	0 1 0 0 1 r r r	(A) ← (A) OR (Rr) for r = 0 – 7	Logical OR contents of designated register with Accumulator (8-bit transfer)	1	1					
ORL A, @ Rr	0 1 0 0 0 0 0 r	(A) ← (A) OR ((Rr)) for r = 0 – 1	Logical OR indirect the contents of the data memory location pointed to by Rr with Accumulator (8-bit operation)	1	1					
ORL A, # data	0 1 0 0 0 0 1 1 d7 d6 d5 d4 d3 d2 d1 d0	(A) ← (A) OR data	Logical OR the specified immediate data with the Accumulator (8-bit operation)	2	2					
ORL PORT, # data	0 1 1 0 0 0 1 1 d7 d6 d5 d4 d3 d2 d1 d0	(P) ← (P) OR data	Logical OR immediate specified data with output port	2	2					
OUT PORT	1 1 0 0 0 0 0 1	(P) ← (A)	Output the contents of the Accumulator to the I/O Port (8-bit transfer)	2	1					
OUT XMTR	1 1 0 0 0 0 0 0	(XMTR) ← (A)	Copy the contents of the Accumulator into the UART Transmit Buffer (8-bit transfer). Also clears Transmit Buffer empty interrupt	1	1					
RET	1 0 0 0 0 0 1 1	(SP) ← (SP) – 1 (PC0–12) ← ((SP))	Return from subroutine without restoring Program Status Word bits 5–7	2	1					
RETR	1 0 0 1 0 0 1 1	(SP) ← (SP) – 1 (PC0–12) ← ((SP)) (PSW 3–7) ← ((SP))	Return from Subroutine restoring Program Status Word (use for all returns from interrupts)	2	1	*	*			
RL A	1 1 1 0 0 1 1 1	(A_{n+1}) ← (An) for n = 0 – 6 (A0) ← (A7)	Rotate Accumulator left by 1 bit without carry	1	1					
RLC A	1 1 1 1 0 1 1 1	(A_{n+1}) ← (An) for n = 0 – 6 (A0) ← (C) (C) ← (A7)	Rotate Accumulator left by 1 bit through carry	1	1	*				

7

Instruction Set (Continued)

Mnemonic	Machine Code	Function	Description	Cycles	Bytes	Flags				
						C	AC	HACC	F0	F1
RR A	0 1 1 1 0 1 1 1	(An) ← A_{n+1} for n = 0 − 6	Rotate Accumulator right by 1 bit without carry	1	1					
RRC A	0 1 1 0 0 1 1 1	(An) ← A_{n+1} for n = 0 −6 (A7) ← (C) (C) ← (A0)	Rotate Accumulator right by 1 bit through carry	1	1	*				
SEL MB0	1 1 0 0 0 1 0 1	(DBF) ← 00	Select Bank 0 (0–2047) of Program Memory	1	1					
SEL MB1	1 1 0 1 0 1 0 1	(DBF) ← 01	Select Bank 1 (2048–4095) of Program Memory	1	1					
SEL MB2	1 1 1 0 0 1 0 1	(DBF) ← 10	Select Bank 2 (4096–6143) of Program Memory	1	1					
SEL MB3	1 1 1 1 0 1 0 1	(DBF) ← 11	Select Bank 3 (6144–8191) of Program Memory	1	1					
SEL RBn	1 1 n 0 0 0 1 1	(BS) ← n for n = 0 − 1	Select Data RAM Bank (0–7) or 1 (24–31)	1	1					
STOP T	0 1 1 0 0 1 0 1		Stop Timer	1	1					
STRT T	0 1 1 1 0 1 0 1		Start Timer	1	1					
SWAP A	0 1 0 0 0 1 1 1	(A4–A7) ⟷ (A0–A3)	SWAP 4 bit nibbles in Accumulator	1	1					
XCH A, Rr	0 0 1 0 1 r r r	(A) ⟷ (Rr) for r = 0 − 7	Exchange the Accumulator and contents of designated register (8-bit transfer)	1	1					
XCH A, @ Rr	0 0 1 0 0 0 0 r	(A) ⟷ ((Rr)) for r = 0 − 1	Exchange indirect the contents of the Accumulator and the data memory location pointed to by Rr (8-bit transfer)	1	1					
XCHD A, @ Rr	0 0 1 1 0 0 0 r	(A0–3) ⟷ ((Rr)) 0−3 for r = 0 − 1	Exchange indirect the low 4 bits of the Accumulator and the data memory location pointed to by Rr (4-bit transfer)	1	1					
XRL A, Rr	1 1 0 1 1 r r r	(A) ← (A) XOR (Rr) for r = 0 − 7	Logical XOR contents of designated register with Accumulator (8-bit transfer)	1	1					
XRL A, @ Rr	1 1 0 1 0 0 0 r	(A) ← (A) XOR ((Rr)) for r = 0 − 1	Logical XOR indirect the contents of the data memory location pointed to by Rr with the Accumulator	1	1					
XRL A, # data	1 1 0 1 0 0 1 1 d7 d6 d5 d4 d3 d2 d1 d0	(A) ← (A) XOR data	Logical XOR the immediate specified data with the Accumulator	2	2					

TMP Opcode Chart

MSN \ LSN	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP	MOV UCR, A	MOV BAUD, A	ADD A, #data	JMP (page 0)	EN XI	JNTF	DEC A	DECL R0	DECL R1	DEC CURS		MOV ENDD, A	MOV BECD, A	MOV SROW, A	
1	INC @R0	INC @R1	JB0	ADDC A, #data	CALL (page 0)	DIS XI	JTF	INC A	INC R0	INC R1	INC R2	INC R3	INC R4	INC R5	INC R6	INC R7
2	XCH A, @R0	XCH A, @R1	MOV PSR, A	MOV A, #data	JMP (page 1)	EN II		CLR A	XCH A, R0	XCH A, R1	XCH A, R2	XCH A, R3	XCH A, R4	XCH A, R5	XCH A, R6	XCH A, R7
3	XCHD A, @R0	XCHD A, @R1	JB1	MOV UMX, A	CALL (page 1)	DIS II		CPL A	INCL R0	INCL R1	INC CURS		MOV AL0, A	MOV AL1, A	MOV A, VPEN	MOV A, HFEN
4	ORL A, @R0	ORL A, @R1	MOV A,T	ORL A, #data	JMP (page 2)	MOV VCR, A		SWAP A	ORL A, R0	ORL A, R1	ORL A, R2	ORL A, R3	ORL A, R4	ORL A, R5	ORL A, R6	ORL A, R7
5	ANL A, @R0	ANL A, @R1	JB2	ANL A, #data	CALL (page 2)	MOV SCR, A		DA A	ANL A, R0	ANL A, R1	ANL A, R2	ANL A, R3	ANL A, R4	ANL A, R5	ANL A, R6	ANL A, R7
6	ADD A, @R0	ADD A, @R1	MOV T,A	ORL PORT, #data	JMP (page 3)	STOP T	JNF1	RRC A	ADD A, R0	ADD A, R1	ADD A, R2	ADD A, R3	ADD A, R4	ADD A, R5	ADD A, R6	ADD A, R7
7	ADDC A, @R0	ADDC A, @R1	JB3	ANL PORT, #data	CALL (page 3)	STRT T	JF1	RR A	ADDC A, R0	ADDC A, R1	ADDC A, R2	ADDC A, R3	ADDC A, R4	ADDC A, R5	ADDC A, R6	ADDC A, R7
8	MOVX @R0, A	MOVX @R1, A	MOV MASK, A	RET	JMP (page 4)	CLR F0	JNF0	MOV TCP, A	MOVL R0, A	MOVL R1, A	MOV HOME, A	MOV CURS, A	MOV A, INTR	MOVX @CURS, A		
9	MOVX A, @R0	MOVX A, @R1	JB4	RETR	CALL (page 4)	CPL F0	JF0	CLR C	MOVL A, R2	MOVL A, R1	MOV A, HOME	MOV A, CURS	MOV A, STAT	MOVX A, @ CURS		
A	MOV @R0, A	MOV @R1, A	MOV VINT, A	JMPP @A	JMP (page 5)	CLR F1	JNXI	CPL C	MOV R0, A	MOV R1, A	MOV R2, A	MOV R3, A	MOV R4, A	MOV R5, A	MOV R6, A	MOV R7, A
B	MOV @R0, #data	MOV @R1, #data	JB5	MOVP A, @A	CALL (page 5)	CPL F1	JXI	MOV @TCP, A	MOV R0, #data	MOV R1, #data	MOV R2, #data	MOV R3, #data	MOV R4, #data	MOV R5, #data	MOV R6, #data	MOV R7, #data
C	OUT XMTR	OUT PORT	MOV HACC, A	SEL RB0	JMP (page 6)	SEL MB0	JZ	MOV A, PSW	DEC R0	DEC R1	DEC R2	DEC R3	DEC R4	DEC R5	DEC R6	DEC R7
D	XRL A, @R0	XRL A, @R1	JB6	XRL A, #data	CALL (page 6)	SEL MB1	JNZ	MOV PSW, A	XRL A, R0	XRL A, R1	XRL A, R2	XRL A, R3	XRL A, R4	XRL A, R5	XRL A, R6	XRL A, R7
E	IN RCVR	IN PORT	MOV A, HACC	SEL RB1	JMP (page 7)	SEL MB2	JNC	RL A	DJNZ R0	DJNZ R1	DJNZ R2	DJNZ R3	DJNZ R4	DJNZ R5	DJNZ R6	DJNZ R7
F	MOV A, @R0	MOV A, @R1	JB7	MOVP3 A, @A	CALL (page 7)	SEL MB3	JC	RLC A	MOV A, R0	MOV A, R1	MOV A, R2	MOV A, R3	MOV A, R4	MOV A, R5	MOV A, R6	MOV A, R7

Ordering Information

ORDER PART NUMBERS

ROMless	NS405-A12N NS405-B12N NS405-C12N	NS405-B18N