Philips Components-Signetics

Document No.	853-0095
ECN No.	85292
Date of Issue	August 26, 1986
Status	Product Specification
Application Specific Product	

SCN8049 Series

SCN8049, SCN8050, SCN8039, SCN8040 single-chip 8-bit microcontroller

DESCRIPTION

The SCN8049 Series Microcontrollers are self-contained, 8-bit processors which contain the system timing, control logic, RAM data memory, ROM program memory (8048/49/50 only), and I/O lines necessary to implement dedicated control functions. All SCN8049 Series devices are pin and program compatible, differing only in the size of the on-board program ROM and data RAM, as follows:

TYPE	RAM SIZE	ROM SIZE
SCN8049	128 x 8	2k x 8
SCN8050	256 x 8	4k x 8
SCN8039	128 x 8	–
SCN8040	256 x 8	–

Program memory can be expanded externally up to a maximum total of 4k bytes without paging. Data memory can also be expanded externally. I/O capabilities can be expanded using standard devices or the 8243 I/O expander.

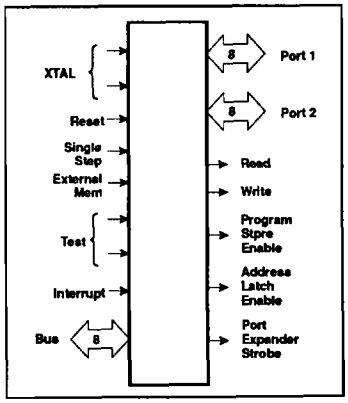
The SCN8049 Series processors are designed to be efficient control processors as well as arithmetic processors. They provide an instruction set which allows the user to directly set and reset individual lines within its I/O ports as well as test individual bits within the accumulator. A large variety of branch and table look-up instructions make these processors very efficient in implementing standard logic functions. Also, special attention has been given to code efficiency. Over 70% of the instructions are a single byte long and all others are only 2 bytes long.

An on-chip 8-bit counter is provided which can count, under program control, either internal clock pulses (with a divide by 32 prescaler) or external events. The counter can be programmed to cause an interrupt on terminal count.

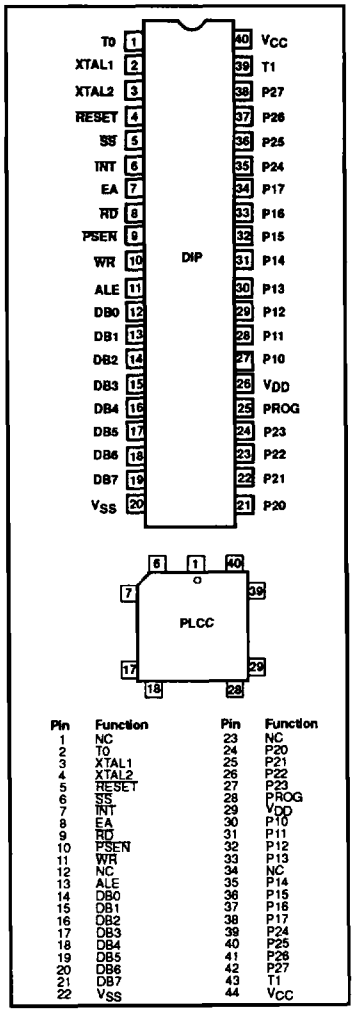
FEATURES

- 8-bit CPU, ROM, RAM, I/O in a 40-pin package
- 24 quasi-bidirectional I/O lines
- Two test inputs
- Internal counter/timer
- Single-level vectored interrupts: external, counter/timer
- Over 90 instructions, 70% single byte
- 1.36µs or 2.5µs instruction cycle, all instructions one or two cycles
- Expandable memory and I/O
- Low voltage standby
- TTL compatible inputs and outputs
- Single +5V power supply

LOGIC SYMBOL



PIN CONFIGURATIONS

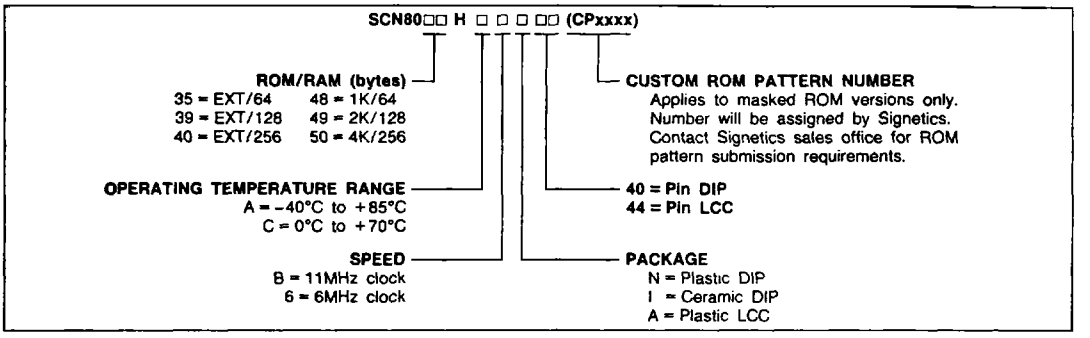


Pin	Function	Pin	Function
1	NC	23	NC
2	T0	24	P20
3	XTAL1	25	P21
4	XTAL2	26	P22
5	RESET	27	P23
6	SS	28	PROG
7	INT	29	VDD
8	EA	30	P10
9	RD	31	P11
10	PSEN	32	P12
11	WR	33	P13
12	NC	34	NC
13	ALE	35	P14
14	DB0	36	P15
15	DB1	37	P16
16	DB2	38	P17
17	DB3	39	P24
18	DB4	40	P25
19	DB5	41	P26
20	DB6	42	P27
21	DB7	43	T1
22	VSS	44	VCC

SCN8049, SCN8050, SCN8039, SCN8040 Single-chip 8-bit microcontroller

SCN8049 Series

ORDERING INFORMATION



PIN DESCRIPTION

MNEMONIC	PIN NO. DIP	PIN NO. PLCC	TYPE	NAME AND FUNCTION
V_{SS}	20	22		Circuit ground potential.
V_{DD}	26	29		Low power standby.
V_{CC}	40	44		**Main Power Supply:** +5V during operation.
PROG	25	28	O	Output strobe for 8243 I/O expander.
P10 - P17	27 - 34	30 - 33, 35 - 38	I/O	**Port 1:** 8-bit quasi-bidirectional port.
P20 - P27	21 - 24, 35 - 38	24 - 27, 39 - 42	I/O	**Port 2:** 8-bit quasi-bidirectional port. P20-23 contain the four high-order program counter bits during an external program memory fetch and serve as a 4-bit I/O expander bus for 8243.
DB0 - DB7	12 - 19	14 - 21	I/O	**Data Bus:** True bidirectional port which can be written or read synchronously using the $\overline{RD}$, $\overline{WR}$ strobes. The port can also be statically latched. Contains the eight low-order program counter bits during an external program memory fetch and receives the addressed instruction under the control of $\overline{PSEN}$. Also contains the address and data during an external RAM data store instruction, under control of ALE, $\overline{RD}$ and $\overline{WR}$.
T0	1	2	I	Input pin testable using the conditional transfer instructions JT0 and JNT0. T0 and be designated as a clock output using the ENT0 CLK instruction.
T1	39	43	I	Input pin testable using the JT1 and JNT1 instructions. Can be designated the timer/counter input using the STRT CNT instruction.
XTAL1	2	3	I	**Crystal 1:** One side of the crystal input for internal oscillator. Also input for external source (non-TTL V_{IH}).
XTAL2	3	4	I	**Crystal 2:** Other side of crystal input.
$\overline{INT}$	6	7	I	**Interrupt:** Initiates an interrupt if interrupt is enabled. Interrupt is disabled after a reset. Also testable with conditional jump instruction. Interrupt must remain low for at least three machine cycles for proper operation.
$\overline{RESET}$	4	5	I	**Reset:** Used to initialize the microcomputer. Active low. Internal pullup ~75KΩ. During program verification the address is latched by a "0" to "1" transition on $\overline{RESET}$ and the data at the addressed location is output on BUS.
$\overline{RD}$	8	9	O	**Read:** Output strobe activated during a bus read. Can be used to enable data onto the bus from an external device. Used as a read strobe to external data memory.
$\overline{WR}$	10	11	O	**Write:** Output strobe during a bus write. Used as write strobe to external data memory.
ALE	11	13	O	**Address Latch Enable:** Occurs once during each cycle and is useful as a clock output. The negative edge of ALE strobes address into external data and program memory.
$\overline{PSEN}$	9	10	O	**Program Store Enable:** Output occurs only during a fetch to external program memory.
$\overline{SS}$	5	6	I	**Single Step:** Can be used in conjunction with ALE to "single step" the processor through each instruction.
EA	7	8	I	**External Access:** Forces all program memory fetches to reference external memory. Useful for emulation and debug, and essential for testing and program verification.

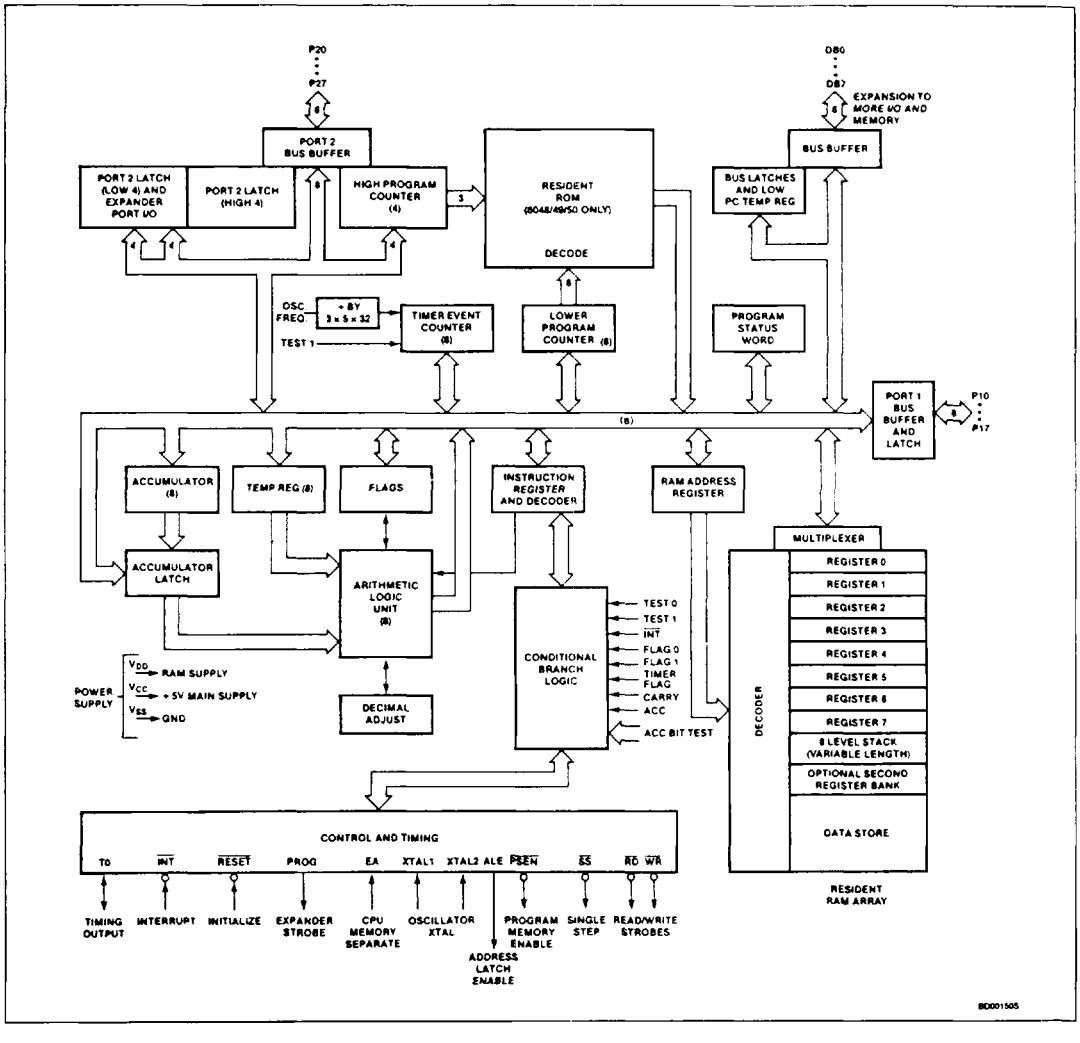
NOTE:
Each pin on these ports can be assigned, under program control, to be an input or an output. A pin is designated as an input by writing a logic "1" to the pin. $\overline{RESET}$ sets all pins to the input mode. Each pin has an internal pullup of approximately 50kΩ.

SCN8049, SCN8050, SCN8039, SCN8040
Single-chip 8-bit microcontroller

FUNCTIONAL DESCRIPTION

The following is a general functional description of the SCN8049 Series microcomputers. Refer to the block diagram below.

BLOCK DIAGRAM



PROGRAM MEMORY

Resident program memory consists of up to 4K bytes of ROM. The program memory is divided into pages of 256 bytes each. As shown in the memory map, Figure 1, program memory is also divided into two 2048-byte banks, MB0 and MB1. A total of 4096 bytes can be addressed directly. If more memory is required, an I/O port can be used to address locations over 4095.

There are three locations in program memory of special importance. These locations contain the first instruction to be executed upon the occurrence of one of three events.

LOCATION	EVENT
0	Activation then deactivation of the RESET line.
3	Activation of the INT line when the external interrupt is enabled.
7	An overflow of the timer/counter if the T/C interrupt is enabled.

DATA MEMORY

Resident data memory, as shown in Figure 2, consists of up to 256 bytes of RAM. All locations are indirectly addressable by either of two RAM pointer registers at locations 0 and 1. The first eight locations of RAM (0–7) are designated as working registers and are directly addressable by several instructions.

By selecting register bank 1, RAM locations 24–31 become the working registers, replacing those in register bank 0 (0–7).

RAM locations 8–23 are designated as the stack. Two locations (bytes) are used per CALL, allowing nesting of up to eight subroutines.

If additional RAM is required, up to 256 bytes may be added and addressed directly using the MOVX instructions. If more RAM is required an I/O port can be used to select one (256-byte) bank of external memory at a time.

PROGRAM COUNTER AND STACK

The Program Counter (PC) is a 12-bit counter/register that points to the location from which the next instruction is to be fetched. The 8048 and 8049 will automatically address external memory when the boundary of their internal memory is exceeded. All processors access external memory if EA is high.

An interrupt or CALL to a subroutine causes the contents of the program counter to be stored in one of the 8 register pairs of the program counter stack. The pair to be used is determined by a 3-bit stack pointer which is part of the Program Status Word (PSW). Data RAM locations 8 through 23 are available as stack registers and are used to store the program counter and 4 bits of PSW. The stack pointer, when initialized to 000, points to RAM locations 8 and 9. The first subroutine jump or interrupt results in the program counter contents being transferred to locations 8 and 9 of the RAM array. The stack pointer is then incremented by one to point to locations 10 and 11 in anticipation of another CALL. Nesting of subroutines within subroutines can continue up to eight times without overflowing the stack. If overflow does occur the deepest address stored (location 8 and 9) will be overwritten and lost since the stack pointer overflows from 111 to 000. It also underflows from 000 to 111.

The end of a subroutine, which is signalled by a return instruction (RET or RETR), causes the stack pointer to be decremented and the contents of the resulting register pair to be transferred to the program counter.

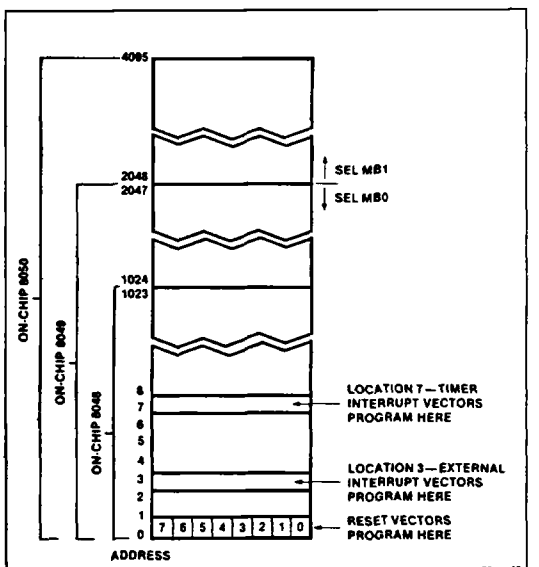


Figure 1. Program Memory Map

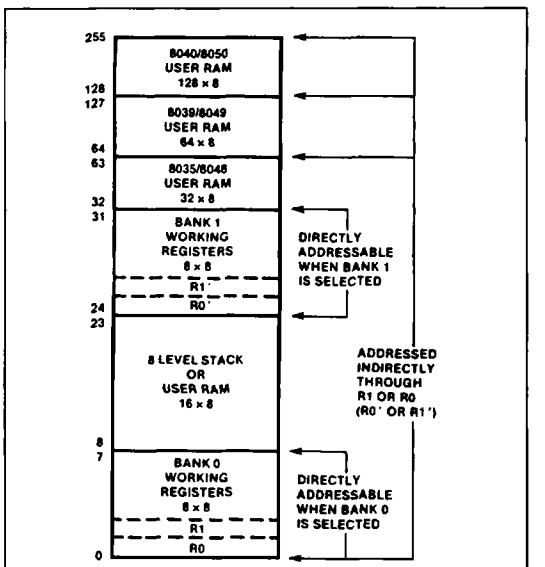


Figure 2. Data Memory Map

OSCILLATOR AND CLOCK

The processor contains its own internal oscillator and clock driver. A crystal, inductor, or external pulse generator may be used to determine the oscillator frequency (see Figure 3). The output of the oscillator is divided by three and can be output on the T0 pin by executing the ENT0 CLK instruction. This CLK signal is divided by 5 to define a machine (instruction) cycle. It is available on Pin 11 as ALE.

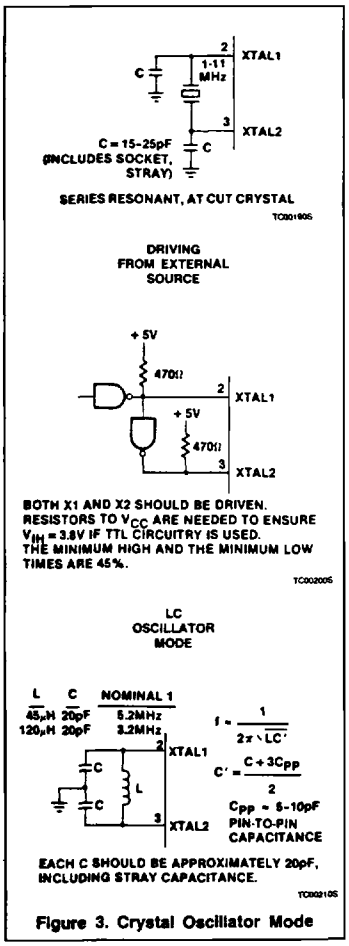


Figure 3. Crystal Oscillator Mode

TIMER/EVENT COUNTER

An internal counter is available which can count either external events or machine cycles (÷ 32). The machine cycles are divided by 32 before they are input to the 8-bit counter. External events are input directly to the counter. The maximum frequency that can be counted is one third of the frequency of the cycle counter. The minimum positive duty cycle that can be detected is 0.2 t_{CY}. The counter is under program control and can be made to generate an interrupt to the processor when it overflows.

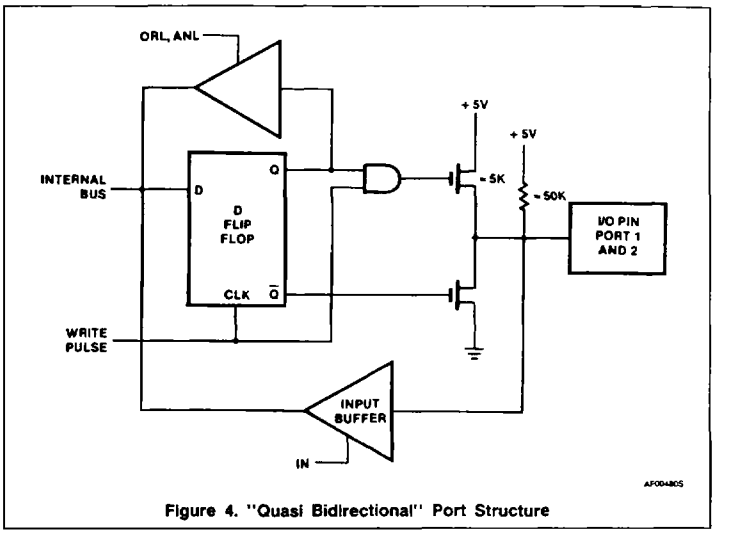


Figure 4. "Quasi Bidirectional" Port Structure

INTERRUPT

An interrupt may be generated by either an external input ($\overline{INT}$, Pin 6) or the overflow of the internal counter, when enabled. In either case, the processor completes execution of the present instruction and then does a CALL to the interrupt service routine. After service, a RETR instruction restores the machine to the state it was prior to the interrupt. The external interrupt has priority over the internal interrupt.

INPUT/OUTPUT

The processor has 27 lines which can be used for input or output functions. These lines are grouped as 3 ports of 8 lines each which serve as either inputs, outputs or bidirectional ports and 3 "test" inputs which can alter program sequences when tested by conditional jump instructions.

Ports 1 and 2

Ports 1 and 2 are each 8 bits wide and have identical characteristics. Data written to these ports is statically latched and remains unchanged until rewritten. As input ports these lines are non-latching; i.e., inputs must be present until read by an input instruction. Inputs are fully TTL compatible and outputs will drive one standard TTL load.

The lines of ports 1 and 2 are called quasi-bidirectional because of a special output circuit structure which allows each line to serve as an input, an output, or both even though outputs are statically latched. Figure 4 shows the circuit configuration. Each line is continuously pulled up to +5V through a resistive device of relatively high impedance (~50K). This pullup is sufficient to provide the source current for a TTL high level yet can be pulled low by a standard TTL gate thus allowing the same pin to be used for both input and output. To provide fast switching times in a "0" to "1" transition a relatively low impedance device (~50KΩ) is switched in momentarily (~500ns) whenever a "1" is written to the line. When a "0" is written to the line, a low impedance (~3000Ω) device overcomes the light pullup and provides TTL current sinking capability.

Since the pulldown transistor is a low impedance device a "1" must first be written to any line which is to be used as an input. Reset initializes all lines to the high impedance "1" state. This structure allows input and output on the same pin and also allows a mix of input lines and output lines on the same port. The quasi-bidirectional port in combination with the ANL and ORL logical instructions provide an efficient means for handling single line inputs and outputs within an 8-bit processor.

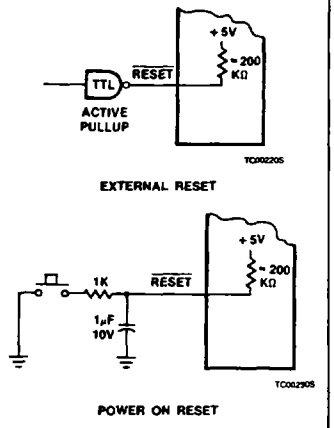


Figure 5. Reset Circuits

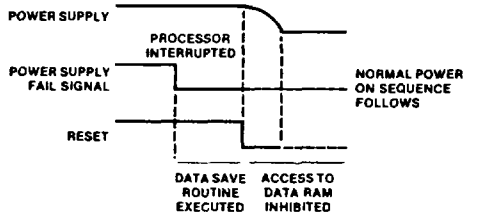


Figure 6. Power Down Sequence

BUS

BUS is also an 8-bit port which is a true bidirectional port with associated input and output strobes. If the bidirectional feature is not needed, BUS can serve as either a statically latched output port or non-latching input port. Input and output lines on this port cannot be mixed.

As a static port, data is written and latched using the OUTL instruction and input using the INS instruction. The INS and OUTL instructions generate pulses on the corresponding $\overline{RD}$ and $\overline{WR}$ output strobe lines; however, in the static port mode they are generally not used. As a bidirectional port, the MOVX instructions are used to read and write to the port. A write to the port generates a pulse on the $\overline{WR}$ output line and output data is valid at the trailing edge of $\overline{WR}$. A read of the port generates a pulse on the $\overline{RD}$ output line and input data must be valid at the trailing edge of $\overline{RD}$. When not being written or read, the BUS lines are in a high impedance state.

Test and INT Inputs

Three pins serve as inputs and are testable with the conditional jump instruction. These are T0, T1, and $\overline{INT}$. These pins allow inputs to cause program branches without the necessity to load an input port into the accumulator. The T0, T1, and $\overline{INT}$ pins have other possible functions as well.

RESET INPUT

The reset input provides a means for initialization for the processor. This Schmitt-trigger input has an internal pullup resistor which in combination with an external 1μF capacitor provides an internal reset pulse of sufficient length to guarantee all circuitry is reset. If the reset pulse is generated externally, the reset pin must be held at ground (0.5V) for at least 10 milliseconds after the power supply is within tolerance. Only five machine cycles (12.5μs @ 6MHz) are required if power is already on and the oscillator has stabilized. Typical circuitry is shown in Figure 5.

SINGLE STEP

By proper control of the $\overline{SS}$ line, the mircrocomputer can be made to execute one instruction and then pause or wait until the single step switch is activated again.

POWER DOWN MODE

The SCN8049 Series devices permit power to be removed from all but the data RAM array for low power standby operation. In the power down mode the contents of data RAM can be maintained while drawing typically 5% of normal operating power.

V_{CC} serves as the 5V supply pin for the bulk of the circuitry while the V_{DD} pin supplies only the RAM array. In normal operation both pins are at + 5V. In standby, V_{CC} is at ground and only V_{DD} is maintained at its specified voltage. Applying $\overline{RESET}$ to the processor through the $\overline{RESET}$ pin inhibits any access to the RAM by the processor and guarantees that RAM cannot be inadvertently altered as power is removed from V_{CC}.

A typical power down sequence occurs as shown in Figure 6.

INSTRUCTION SET

The SCN8049 Series instruction set consists of over 90 one and two byte instructions (see Table 1). Program code efficiency is high because: (1) working registers and program variables are stored in RAM, which require only one byte to address and (2) program memory is divided into pages of 256 bytes each, which means that branch destination addresses require one byte.

The instruction set efficiently manipulates and tests bits in addition to performing logical and arithmetic operations upon and the testing of bytes. A set of move instructions operates indirectly upon either RAM or ROM, which permits efficient access of pointers and data tables. The indirect jump instruction performs a multi (up to 256) way branch upon the content of the accumulator to addresses stored in a lookup table. The "decrement register and jump if not zero" instruction saves a byte every time it is used versus using separate increment and test instructions.

The on-chip counter enables either external events or time to be counted off-line from the main program. The processor can either test the counter (under program control) or cause its overflow to generate an interrupt. These features are highly desirable for real time applications. See Table 2 for instruction timing.

ABSOLUTE MAXIMUM RATINGS[1]

SYMBOL	PARAMETER	RATING	UNIT
T_A	Operating ambient temperature[2] range SCN80xxHC SCN80xxHA	 0 to +70 −40 to +85	 °C °C
T_{STG}	Storage temperature range	−65 to +150	°C
V_{IN}	Input voltages with respect to V_{SS}[3]	−0.5 to +7	V
P_D	Power dissipation	1.5	W

DC ELECTRICAL CHARACTERISTICS T_A = 0°C to 70°C, V_{CC} = V_{DD} = 5V ± 10%, V_{SS} = 0V[4, 5, 6]

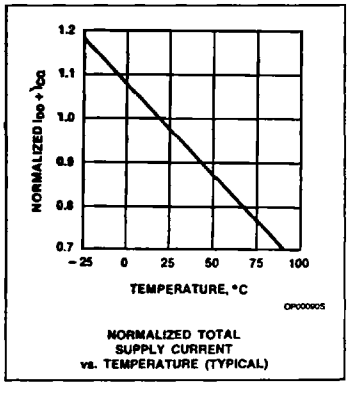
SYMBOL	PARAMETER	TEST CONDITIONS[7]	LIMITS			UNIT
			Min	Typ	Max	
V_{IL}	Input low-voltage All except XTAL1, XTAL2		−0.5		0.8	V
V_{IL1}	XTAL1, XTAL2		−0.5		0.6	V
V_{IH}	Input high voltage All except $\overline{RESET}$, XTAL1, XTAL2		2.0		V_{CC}	V
V_{IH1}	$\overline{RESET}$, XTAL1, XTAL2		3.8		V_{CC}	V
V_{OL}	Output low-voltage	I_{OL} = 2.0mA			0.4	V
V_{OH}	Output high-voltage All except BUS BUS	 I_{OH} = −125μA I_{OH} = −400μA	 2.4 2.4		 	 V V
I_{LI1}	Port1, Port2, EA, $\overline{SS}$	V_{SS} + 0.45 ≤ V_{IN} ≤ V_{CC}			−500	μA
I_{L1}	T1, $\overline{Int}$	V_{SS} + 0.45 ≤ V_{IN} ≤ V_{CC}			±10	μA
I_{LI2}	$\overline{RESET}$	V_{SS} + 0.45 ≤ V_{IN} ≤ V_{CC}	−10		−300	μA
I_{OL}	Output leakage current BUS, T0 (high inpedance state)	 V_{SS} + 0.45 ≤ V_{IN} ≤ V_{CC}			 ±10	 μA
I_{DD}	Standby supply current 8035/8048 8039/8049 8040/8050	$\overline{RESET}$ ≤ V_{IL} All inputs = 0V V_{CC} = 0V			 2.5 4.5 8.5	 mA mA mA
I_{DD} + I_{CC}	Total supply current 8035/8048 8039/8049 8040/8050	$\overline{RESET}$ ≤ V_{IL}		 45 50 60	 80 95 110	 mA mA mA
V_{DD}	Standby power supply		2.5			V
T_A = −40 to 85°C, Automotive temperature range[8]						
V_{IH}	Input high voltage All except XTAL1 and XTAL2		2.2			V
V_{IH1}	$\overline{RESET}$, XTAL1, XTAL2		4.0			V
	Input leakage current					
I_{IL1}	Port1, Port2, EA, $\overline{SS}$	V_{SS} + .45 ≤ V_{IN} ≤ V_{CC}			−750	μA
I_{LI2}	$\overline{RESET}$	V_{SS} + .45 ≤ V_{IN} ≤ V_{CC}	−5		−300	μA
I_{DD}	Standby supply current 8035/8048 8039/8049 8040/8050	 $\overline{RESET}$ ≤ V_{IL} All inputs = 0V V_{CC} = 0V			 3.75 6.75 12.75	 mA mA mA
I_{CC} + I_{DD}	Total supply current 8035/8048 8039/8049 8040/8050	 $\overline{RESET}$ ≤ V_{IL}			 90 105 120	 mA mA mA

AC ELECTRICAL CHARACTERISTICS $T_A = 0°C$ to $70°C$, $V_{CC} = V_{DD} = 5V \pm 10\%$, $V_{SS} = 0V$[4, 5, 6]

SYMBOL	PARAMETER	TEST CONDITIONS[7]	11 MHz VERSIONS		6 MHz VERSIONS		UNIT
			Min	Max	Min	Max	
(Refer to Figures 7, 8 and 9)							
t_{LL}	ALE pulse width		150		400		ns
t_{AL}	Address setup to ALE		70		150		ns
t_{LA}	Address hold from ALE		50		80		ns
t_{CC}	Control pulse width ($\overline{PSEN}$, $\overline{RD}$, $\overline{WR}$)		300		700		ns
t_{DW}	Data setup before $\overline{WR}$		250		500		ns
t_{WD}	Data hold after $\overline{WR}$		40		120		ns
t_{CY}	Cycle time		1.36	3.75	2.5	15.0	µs
t_{DR}	Data hold		0	100	0	200	ns
t_{RD}	$\overline{PSEN}$, $\overline{RD}$ to data in			200		500	ns
t_{AW}	Address setup to $\overline{WR}$		200		230		ns
t_{AD}	Address setup to data in			400		950	ns
t_{AFC}	Address float to $\overline{RD}$, $\overline{PSEN}$		−10		0		ns
t_{CA}	Control pulse to ALE		10		10		ns
(Refer to Figure 10)							
t_{CP}	Port control setup before falling edge of PROG		100		110		ns
t_{PC}	Port control hold after falling edge of PROG		60		130		ns
t_{PR}	PROG to time P2 input must be valid			650		810	ns
t_{DP}	Output data setup time		200		250		ns
t_{PD}	Output data hold time		20		65		ns
t_{PF}	Input data hold time		0	150	0	150	ns
t_{PP}	PROG pulse width		700		1200		ns
t_{PL}	Port 2 I/O data setup		250		350		ns
t_{LP}	Port 2 I/O data hold		20		150		ns

NOTES:

1. Stresses above those listed under absolute maximum ratings may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other condition above those indicated in the operation section of this specification is not implied.
2. For operating at elevated temperatures, the device must be derated based on + 150°C maximum junction temperature.
3. This product includes circuitry specifically designed for the protection of its internal devices from damaging effects of excessive static charge. Nonetheless, it is suggested that conventional precautions be taken to avoid applying any voltages larger than the rated maximum.
4. Parameters are valid over operating temperature range unless otherwise specified.
5. All voltage measurements are referenced to ground (V_{SS}). For testing, all input signals swing between 0.4V and 2.4V with a transition time of 20ns maximum. All time measurements are referenced at input voltages of 0.8V and 2.0V and output voltages 0.8V and 2.0V as appropriate.
6. Typical values are at + 25°C, typical ssupply voltages and typical processing parameters.
7. Control outputs: C_L = 80pF
   Bus outputs: C_L = 150pF
   t_{CY} = 1.36µs for 11 MHz versions
   t_{CY} = 2.5µs for 6 MHz versions
8. Where no specification is shown, the commercial temperature range specification applies.



NORMALIZED TOTAL SUPPLY CURRENT vs. TEMPERATURE (TYPICAL)

TIMING DIAGRAMS

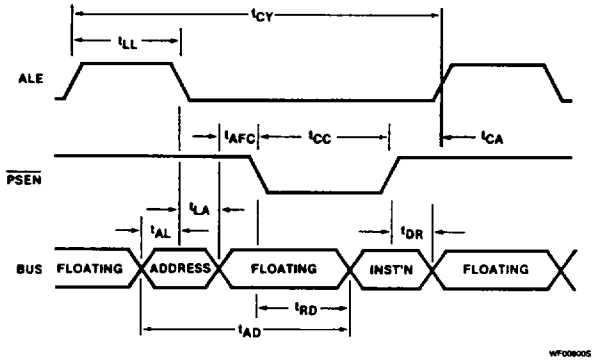


Figure 7. Instruction Fetch From External Program Memory

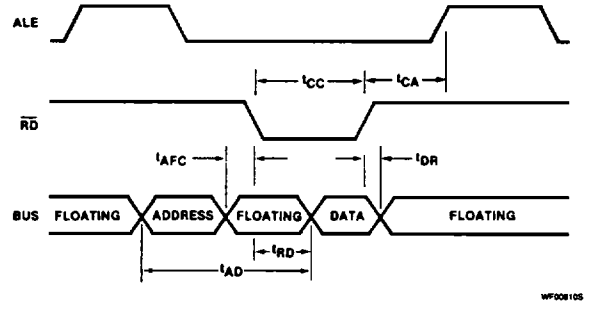


Figure 8. Read From External Data Memory

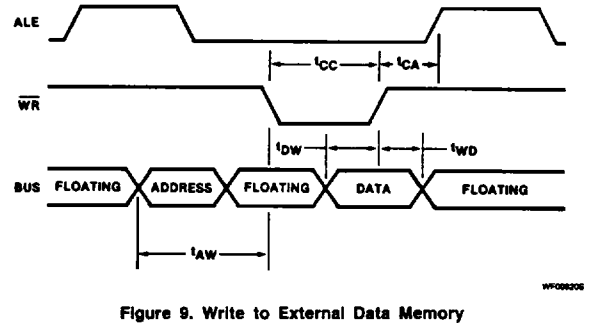


Figure 9. Write to External Data Memory

TIMING DIAGRAMS (Continued)

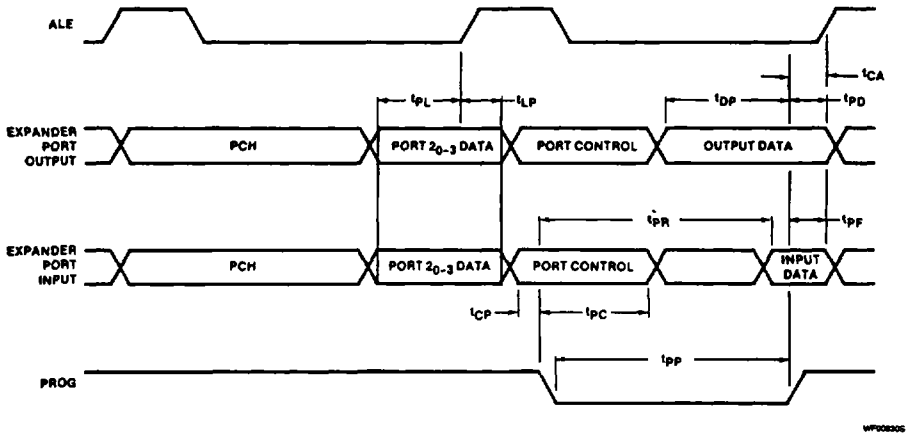


Figure 10. Port 2 Timing

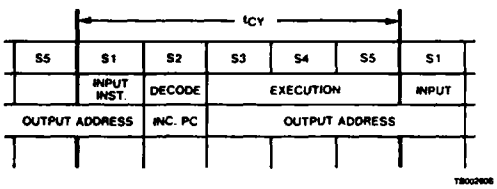


Figure 11. Instruction Cycle

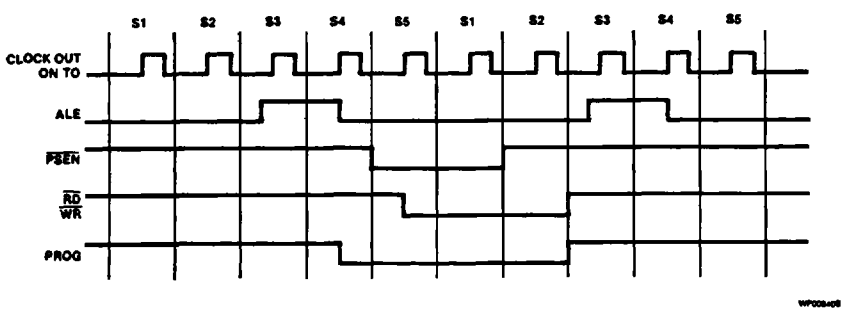


Figure 12. Instruction Cycle Timing

Table 1. Instruction Set

MNEMONIC	FUNCTION	DESCRIPTION	INSTRUCTION CODE D_7 D_6 D_5 D_4 D_3 D_2 D_1 D_0	CYCLES	BYTES	FLAGS C	AC	F0	F1	F2
Accumulator										
ADD A, # data	(A) ← (A) + data	Add immediate the specified data to the accumulator.	0 0 0 0 0 0 1 1 d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0	2	2	•	•			
ADD A, Rr	(A) ← (A) + (Rr) for r = 0 – 7	Add contents of designated register to the accumulator.	0 1 1 0 1 r r r	1	1	•	•			
ADD A, @ Rr	(A) ← (A) + ((Rr)) for r = 0 – 1	Add indirect the contents the data memory location to the accumulator.	0 1 1 0 0 0 0 r	1	1	•	•			
ADDC A, # data	(A) ← (A) + (C) + data	Add immediate with carry the specified data to the accumulator.	0 0 0 1 0 0 1 1 d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0	2	2	•	•			
ADDC A, Rr	(A) ← (A) + (C) + (Rr) for r = 0 – 7	Add with carry the contents of the designated register to the accumulator.	0 1 1 1 1 r r r	1	1	•	•			
ADDC A, @ Rr	(A) ← (A) + (C) + ((Rr)) for r = 0 – 1	Add indirect with carry the contents of data memory location to the accumulator.	0 1 1 1 0 0 0 r	1	1	•	•			
ANL A, # data	(A) ← (A) AND data	Logical AND specified immediate data with accumulator.	0 1 0 1 0 0 1 1 d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0	2	2					
ANL A, Rr	(A) ← (A) AND (Rr) for r = 0 – 7	Logical AND contents of designated register with accumulator.	0 1 0 1 1 r r r	1	1					
ANL A, @ Rr	(A) ← (A) AND ((Rr)) for r = 0 – 1	Logical AND indirect the contents of data memory with accumulator.	0 1 0 1 0 0 0 r	1	1					
CPL A	(A) ← NOT (A)	Complement the contents of the accumulator.	0 0 1 1 0 1 1 1	1	1					
CLR A	(A) ← 0	Clear the contents of the accumulator.	0 0 1 0 0 1 1 1	1	1					
DA A		Decimal adjust the contents of the accumulator.	0 1 0 1 0 1 1 1	1	1	•	•			
DEC A	(A) ← (A) - 1	Decrement the accumulator's contents by 1.	0 0 0 0 0 1 1 1	1	1					
INC A	(A) ← (A) + 1	Increment the accumulator's contents by 1.	0 0 0 1 0 1 1 1	1	1					
ORL A, # data	(A) ← (A) OR data	Logical OR specified immediate data with accumulator.	0 1 0 0 0 0 1 1 d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0	2	2					
ORL A, Rr	(A) ← (A) OR (Rr) for r = 0 – 7	Logical OR contents of designated register with accumulator.	0 1 0 0 1 r r r	1	1					
ORL A, @ Rr	(A) ← (A) OR ((Rr)) for r = 0 – 1	Logical OR indirect the contents of data memory location with accumulator.	0 1 0 0 0 0 0 r	1	1					
RL A	(An + 1) ← (An) (A_0) ← (A_7) for N = 0 ← 6	Rotate accumulator left by 1-bit without carry.	1 1 1 0 0 1 1 1	1	1					
RLC A	(An + 1) ← (An); n = 0 – 6 (A_0) ← (C) (C) ← (A_7)	Rotate accumulator left by 1-bit through carry.	1 1 1 1 0 1 1 1	1	1	•				
RR A	(An) ← (An + 1); n = 0 – 6 (A_7) ← (A_0)	Rotate accumulator right by 1-bit without carry.	0 1 1 1 0 1 1 1	1	1					
RRC A	(An) ← (An + 1); n = 0 – 6 (A_7) ← (C) (C) ← (A_0)	Rotate accumulator right by 1-bit through carry.	0 1 1 0 0 1 1 1	1	1	•				
SWAP A	(A_{4-7}) ← (A_0 – 3)	Swap the 2 4-bit nibbles in the accumulator.	0 1 0 0 0 1 1 1	1	1					
XRL A, # data	(A) ← (A) XOR data	Logical XOR specified immediate data with accumulator.	1 1 0 1 0 0 1 1 d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0	2	2					
XRL A, Rr	(A) ← (A) XOR (Rr) for r = 0 – 7	Logical XOR contents of designated register with accumulator.	1 1 0 1 1 r r r	1	1					
XRL A, @ Rr	(A) ← (A) XOR ((Rr)) for r = 0 – 1	Logical XOR indirect the contents of data memory location with accumulator.	1 1 0 1 0 0 0 r	1	1					

Table 1. Instruction Set (Continued)

MNEMONIC	FUNCTION	DESCRIPTION	INSTRUCTION CODE D_7 D_6 D_5 D_4 D_3 D_2 D_1 D_0	CYCLES	BYTES	FLAGS C	AC	F0	F1	F2
Branch										
DJNZ Rr, addr	(Rr) ← (Rr) – 1; r = 0 – 7 if (Rr) ≠ 0: (PC 0 – 7) ← addr	Decrement the specified register and test contents.	1 1 1 0 1 r r r a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
JBb addr	(PC 0 – 7) ← addr if Bb = 1 (PC) ← (PC) + 2 if Bb = 0	Jump to specified address if accumulator bit is set.	b_2 b_1 b_0 1 0 0 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
JC addr	(PC 0 – 7) ← addr if C = 1 (PC) ← (PC) + 2 if C = 0	Jump to specified address if carry flag is set.	1 1 1 1 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
JF0 addr	(PC 0 – 7) ← addr if F0 = 1 (PC) ← (PC) + 2 if F0 = 0	Jump to specified address if flag F0 is set.	1 0 1 1 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
JF1 addr	(PC 0 – 7) ← addr if F1 = 1 (PC) ← (PC) + 2 if F1 = 0	Jump to specified address if flag F1 is set.	0 1 1 1 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
JMP addr	(PC 8 – 10) ← addr 8 – 10 (PC 0 – 7) ← addr 0 – 7 (PC 11) ← (DBF)	Direct jump to specified address within the 2K address block.	a_{10} a_9 a_8 0 0 1 0 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
JMPP @ A	(PC 0 – 7) ← ((A))	Jump indirect to specified address within address page.	1 0 1 1 0 0 1 1	2	1					
JNC addr	(PC 0 – 7) ← addr if C = 0 (PC) ← (PC) + 2 if C = 1	Jump to specified address if carry flag is low.	1 1 1 0 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
JNI	(PC 0 – 7) ← addr if $\overline{INT}$ = 0 (PC) ← (PC) + 2 if $\overline{INT}$ = 1	Jump to specified address if $\overline{INT}$ input is low.	1 0 0 0 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
JNT0 addr	(PC 0 – 7) ← addr if T0 = 0 (PC) ← (PC) + 2 if T0 = 1	Jump to specified address if test 0 is low.	0 0 1 0 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
JNT1 addr	(PC 0 – 7) ← addr if T1 = 0 (PC) ← (PC) + 2 if T1 = 1	Jump to specified address if test 1 is low.	0 1 0 0 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
JNZ addr	(PC 0 – 7) ← addr if A = 0 (PC) ← (PC) + 2 if A = 0	Jump to specified address if accumulator is non-zero.	1 0 0 1 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
JTF addr	(PC 0 – 7) ← addr if TF = 1 (PC) ← (PC) + 2 if TF = 0	Jump to specified address if timer flag is set to 1.	0 0 0 1 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
JT0 addr	(PC 0 – 7) ← addr if T0 = 1 (PC) ← (PC) + 2 if T0 = 0	Jump to specified address if test 0 is a 1.	0 0 1 1 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
JT1 addr	(PC 0 – 7) ← addr if T1 = 1 (PC) ← (PC) + 2 if T1 = 0	Jump to specified address if test 1 is a 1.	0 1 0 1 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
JZ addr	(PC 0 – 7) ← addr if A = 0 (PC) ← (PC) + 2 if A ≠ 0	Jump to specified address if accumulator is 0.	1 1 0 0 0 1 1 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
Control										
EN I		Enable the external ($\overline{INT}$) interrupt.	0 0 0 0 0 1 0 1	1	1					
DIS I		Disable the external ($\overline{INT}$) interrupt.	0 0 0 1 0 1 0 1	1	1					
SEL RB0	(BS) ← 0	Select bank 0 (locations 0 – 7) of data memory.	1 1 0 0 0 1 0 1	1	1					•

Table 1. Instruction Set (Continued)

MNEMONIC	FUNCTION	DESCRIPTION	INSTRUCTION CODE D_7 D_6 D_5 D_4 D_3 D_2 D_1 D_0	CYCLES	BYTES	FLAGS C	AC	F0	F1	F2
Control (Cont.)										
SEL RB1	(BS) ← 1	Select bank 1 (locations 24 – 31) of data memory.	1 1 0 1 0 1 0 1	1	1					•
SEL MB0	(DBF) ← 0	Select program memory bank 0, addresses 0 – 2047.	1 1 1 0 0 1 0 1	1	1					
SEL MB1	(DBF) ← 1	Select program memory bank 1, addresses 2048 – 4095	1 1 1 1 0 1 0 1	1	1					
ENT0 CLK		Enable clock output on T0 pin.	0 1 1 1 0 1 0 1	1	1					
Data moves										
MOV A, # data	(A) ← data	Move immediate the specified data into the accumulator.	0 0 1 0 0 0 1 1 d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0	2	2					
MOV A, Rr	(A) ← (Rr); r = 0 – 7	Move the contents of the designated register into the accumulator.	1 1 1 1 1 r r r	1	1					
MOV A, @ Rr	(A) ← ((Rr)); r = 0 – 1	Move indirect the contents of data memory location into the accumulator.	1 1 1 1 0 0 0 r	1	1					
MOV A, PSW	(A) ← (PSW)	Move contents of the program status word into the accumulator.	1 1 0 0 0 1 1 1	1	1					
MOV Rr, # data	(Rr) ← data; r = 0 – 7	Move immediate the specified data into the designated register.	1 0 1 1 1 r r r d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0	2	2					
MOV Rr, A	(Rr) ← (A); r = 0 – 7	Move accumulator contents into the designated register.	1 0 1 0 1 r r r	1	1					
MOV @ Rr, A	((Rr)) ← (A); r = 0 – 1	Move indirect accumulator contents into data memory location.	1 0 1 0 0 0 0 r	1	1					
MOV @ Rr, # data	((Rr)) ← data; r = 0 – 1	Move indirect the specified data into data memory.	1 0 1 1 0 0 0 r d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0	2	2					
MOV PSW, A	(PSW) ← (A)	Move contents of accumulator into the program status word.	1 1 0 1 0 1 1 1	1	1	•	•	•		•
MOVP A, @ A	(A) ← ((A))	Move data in the current page into the accumulator.	1 0 1 0 0 0 1 1	2	1					
MOVP3 A, @ A	(A) ← ((A)) in page 3	Move data in page 3 into the accumulator.	1 1 1 0 0 0 1 1	2	1					
MOVX A, @ Rr	(A) ← ((Rr)); r = 0 – 1	Move indirect the contents of external memory location into the accumulator.	1 0 0 0 0 0 0 r	2	1					
MOVX @ Rr, A	((Rr)) ← (A); r = 0 – 1	Move indirect the contents of the accumulator into external memory.	1 0 0 1 0 0 0 r	2	1					
XCH A, Rr	(A)↔(Rr); r = 0 – 7	Exchange the accumulator and designated register's contents.	0 0 1 0 1 r r r	1	1					
XCH A, @ Rr	(A)↔((Rr)); r = 0 – 1	Exchange indirect contents of accumulator and location in data memory.	0 0 1 0 0 0 0 r	1	1					
XCHD A, @ Rr	(A 0 – 3)↔(Rr)(0 – 3) r = 0 – 1	Exchange indirect 4-bit contents of accumulator and data memory.	0 0 1 1 0 0 0 r	1	1					
Flags										
CPL C	(C) ← NOT (C)	Complement content of carry bit.	1 0 1 0 0 1 1 1	1	1	•				
CPL F0	(F0) ← NOT (F0)	Complement content of flag F0.	1 0 0 1 0 1 0 1	1	1			•		
CPL F1	(F1) ← NOT (F1)	Complement content of flag F1.	1 0 1 1 0 1 0 1	1	1				•	
CLR C	(C) ← 0	Clear content of carry bit to 0.	1 0 0 1 0 1 1 1	1	1	•				
CLR F0	(F0) ← 0	Clear content of flag 0 to 0.	1 0 0 0 0 1 0 1	1	1			•		
CLR F1	(F1) ← 0	Clear content of flag 1 to 0.	1 0 1 0 0 1 0 1	1	1					•
Input/output										
ANL BUS, # data	(BUS) ← (BUS) AND data	Logical AND immediate specified data with BUS.	1 0 0 1 1 0 0 0 d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0	2	2					
ANL Pp, # data	(Pp) ← (Pp) AND data p = 1 – 2	Logical AND immediate specified data with designated port (1 or 2).	1 0 0 1 1 0 p p d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0	2	2					
ANLD Pp, A	(Pp) ← (Pp) AND (A 0 – 3) p = 4 – 7	Logical AND contents of accumulator with designated port (4 – 7).	1 0 0 1 1 1 p p	2	1					

Table 1. Instruction Set (Continued)

MNEMONIC	FUNCTION	DESCRIPTION	INSTRUCTION CODE D_7 D_6 D_5 D_4 D_3 D_2 D_1 D_0	CYCLES	BYTES	FLAGS C	AC	F0	F1	F2
Input/output (Cont.)										
IN A, Pp	(A) ← (Pp); p = 1 – 2	Input data from designated port (1 – 2) into accumulator.	0 0 0 0 1 0 p p	2	1					
INS A, BUS	(A) ← (BUS)	Input strobed BUS data into accumulator.	0 0 0 0 1 0 0 0	1	2					
MOVD A, Pp	(A 0 – 3) ← (Pp); p = 4 – 7 (A 4 – 7) ← 0	Move contents of designated port (4 – 7) into accumulator.	0 0 0 0 1 1 p p	2	1					
MOVD Pp, A	(Pp) ← A 0 – 3; p = 4 – 7	Move contents of accumulator to designated port (4 – 7).	0 0 1 1 1 1 p p	1	1					
ORLD Pp, A	(Pp) ← (Pp) OR (A 0 – 3) p = 4 – 7	Logical OR contents of accumulator with designated port (4 – 7).	1 0 0 0 1 1 p p	1	1					
ORL BUS, # data	(BUS) ← (BUS) OR data	Logical OR immediate specified data with BUS.	1 0 0 0 1 0 0 0 d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0	2	2					
ORL Pp, # data	(Pp) ← (Pp) OR data p = 1 – 2	Logical OR immediate specified data with designated port (1 – 2).	1 0 0 0 1 0 p p d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0	2	2					
OUTL BUS, A	(BUS) ← (A)	Output contents of accumulator onto BUS.	0 0 0 0 0 0 1 0	1	2					
OUTL Pp, A	(Pp) ← (A); p = 1 – 2	Output contents of accumulator to designated port (1 – 2).	0 0 1 1 1 0 p p	1	1					
Registers										
DEC Rr	(Rr) ← (Rr) – 1; r = 0 – 7	Decrement contents of designated register by 1.	1 1 0 0 1 r r r	1	1					
INC Rr	(Rr) ← (Rr) + 1; r = 0 – 7	Increment contents of designated register by 1.	0 0 0 1 1 r r r	1	1					
INC @ Rr	((Rr)) ← ((Rr)) + 1; r = 0 – 1	Increment indirect the contents of data memory location by 1.	0 0 0 1 0 0 0 r	1	1					
Subroutine										
CALL addr	((SP)) ← (PC), (PSW 4 – 7) (SP) ← (SP) + 1 (PC 8 – 10) ← addr 8 – 10 (PC 0 – 7) ← addr 0 – 7 (PC 11) ← DBF	Call designated subroutine.	a_{10} a_9 a_8 1 0 1 0 0 a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0	2	2					
RET	(SP) ← (SP) – 1 (PC) ← ((SP))	Return from subroutine without restoring program status word.	1 0 0 0 0 0 1 1	2	1					
RETR	(SP) ← (SP) – 1 (PC) ← ((SP)) (PSW 4 – 7) ← ((SP))	Return from subroutine restoring program status word.	1 0 0 1 0 0 1 1	2	1					
Timer/counter										
EN TCNTI		Enable timer/counter interrupt.	0 0 1 0 0 1 0 1	1	1					
DIS TCNTI		Disable timer/counter interrupt.	0 0 1 1 0 1 0 1	1	1					
MOV A, T	(A) ← (T)	Move contents of timer/counter into accumulator.	0 1 0 0 0 0 1 0	1	1					
MOV T, A	(T) ← (A)	Move contents of accumulator into timer/counter.	0 1 1 0 0 0 1 0	1	1					
STOP TCNT		Stop count for event counter or timer.	0 1 1 0 0 1 0 1	1	1					
STRT CNT		Start count for event counter.	0 1 0 0 0 1 0 1	1	1					
STRT T		Start count for timer.	0 1 0 1 0 1 0 1	1	1					
Miscellaneous										
NOP		No operation performed	0 0 0 0 0 0 0 0	1	1					

NOTES:
1. Instruction code designations r and p form the binary representation of the registers and ports involved.
2. The dot under the appropriate flag bit indicates that its content is subject to change by the instruction in which it appears.
3. Numerical subscripts appearing in the FUNCTION column reference the specific bits affected.

SYMBOL DEFINITIONS

SYMBOL	DESCRIPTION
A	The accumulator
AC	The auxiliary carry flag
addr	Program memory address (11 bits)
Bb	Bit designator (b = 0 – 7)
BS	The bank switch
C	Carry flag
CLK	Clock signal
CNT	Event counter
D	Nibble designator (4 bits)
DBF	Program memory bank flip-flop
data	Number or expression (8 bits)
F_0, F_1	Flags 0, 1
I	Interrupt
$\overline{INT}$	External interrupt

P	"In-Page" operation designator
P_p	Port designator (p = 1, 2 or 4 – 7)
PSW	Program status word
Rr	Register designator (r = 0, 1 or 0 – 7)
SP	Stack pointer
T	Timer
TF	Timer flag
T0, T1	Testable inputs 0, 1
#	Prefix for immediate data
@	Prefix for indirect address
$	Program counter's current value
←	Replaced by
↔	Exchanged with

Table 2. Instruction Timing**

	CYCLE 1					CYCLE 2				
INSTRUCTION	S1	S2	S3	S4	S5	S1	S2	S3	S4	S5
IN A,P	Fetch Instruction	Increment Program Counter	—	Increment Timer	—	—	Read Port	* —	—	—
OUTL P,A	Fetch Instruction	Increment Program Counter	—	Increment Timer	Output To Port	—	—	* —	—	—
ANL P, # data	Fetch Instruction	* Increment Program Counter	—	Increment Timer	Read Port	Fetch Immediate Data	—	* Increment Program Counter	Output To Port	—
ORL P, # data	Fetch Instruction	* Increment Program Counter	—	Increment Timer	Read Port	Fetch Immediate Data	—	* Increment Program Counter	Output To Port	—
INS A, BUS	Fetch Instruction	Increment Program Counter	—	Increment Timer	—	—	Read Port	* —	—	—
OUTL BUS, A	Fetch Instruction	Increment Program Counter	—	Increment Timer	Output To Port	—	—	* —	—	—
ANL BUS, # data	Fetch Instruction	* Increment Program Counter	—	Increment Timer	Read Port	Fetch Immediate Data	—	* Increment Program Counter	Output To Port	—
ORL BUS, # data	Fetch Instruction	* Increment Program Counter	—	Increment Timer	Read Port	Fetch Immediate Data	—	* Increment Program Counter	Output To Port	—
MOVX @R,A	Fetch Instruction	Increment Program Counter	Output RAM Address	Increment Timer	Output Data to RAM	—	—	* —	—	—
MOVX A,@R	Fetch Instruction	Increment Program Counter	Output RAM Address	Increment Timer	—	—	Read Data	* —	—	—
MOVD A, P_i	Fetch Instruction	Increment Program Counter	Output Opcode/Address	Increment Timer	—	—	Read P2 Lower	* —	—	—
MOVD P_i, A	Fetch Instruction	Increment Program Counter	Output Opcode/Address	Increment Timer	Output Data TO P2 Lower	—	—	* —	—	—
ANLD P, A	Fetch Instruction	Increment Program Counter	Output Opcode/Address	Increment Timer	Output Data	—	—	* —	—	—
ORLD P, A	Fetch Instruction	Increment Program Counter	Output Opcode/Address	Increment Timer	Output Data	—	—	* —	—	—
J (CONDITIONAL)	Fetch Instruction	* Increment Program Counter	Sample Condition	Increment Timer	—	Fetch Immediate Data	—	* Update Program Counter	—	—
START CNT/STRT T	Fetch Instruction	* Increment Program Counter	—	—	Start Counter					
STOP TCNT	Fetch Instruction	* Increment Program Counter	—	—	Stop Counter					
EN I	Fetch Instruction	* Increment Program Counter	—	Enable Interrupt	—					
DIS I	Fetch Instruction	* Increment Program Counter	—	Disable Interrupt	—					
ENT0 CLK	Fetch Instruction	* Increment Program Counter	—	Enable Clock	—					

NOTES:
*Valid instruction address are output at this time if external program memory is being accessed.
**See figures 11 and 12 for instruction cycle and cycle timing.