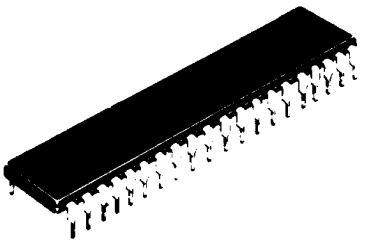
HD6805U1
MCU (Microcomputer Unit)

The HD6805U1 is the 8-bit Microcomputer Unit (MCU) which contains a CPU, on-chip clock, ROM, RAM, I/O and timer. It is designed for the user who needs an economical microcomputer with the proven capabilities of the HD6800-based instruction set.

The following are some of the hardware and software highlights of the MCU.

HD6805U1P



(DP-40)

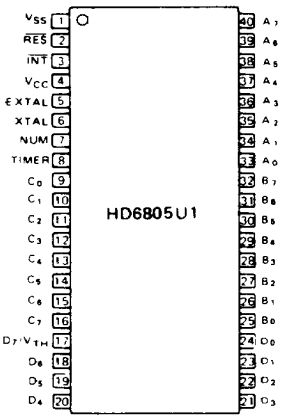
■ HARDWARE FEATURES

- 8-Bit Architecture
- 96 Bytes of RAM
- Memory Mapped I/O
- 2056 Bytes of User ROM
- Internal 8-Bit Timer with 7-Bit Prescaler
- Vectored Interrupts – External and Timer
- 24 I/O Ports + 8 Input Port
  (8 Lines LED Compatible; 7 Bits Comparator Inputs)
- On-Chip Clock Circuit
- Self-Check Mode
- Master Reset
- Low Voltage Inhibit
- Easy for System Development and Debugging
- 5 Vdc Single Supply

■ SOFTWARE FEATURES

- Similar to HD6800
- Byte Efficient Instruction Set
- Easy to Program
- True Bit Manipulation
- Bit Test and Branch Instructions
- Versatile Interrupt Function
- Powerful Indexed Addressing for Tables
- Full Set of Conditional Branches
- Memory Usable as Registers/Flags
- Single Instruction Memory Examine/Change
- 10 Powerful Addressing Modes
- All Addressing Modes Apply to ROM, RAM and I/O
- Compatible Instruction Set with MC6805P2

■ PIN ARRANGEMENT

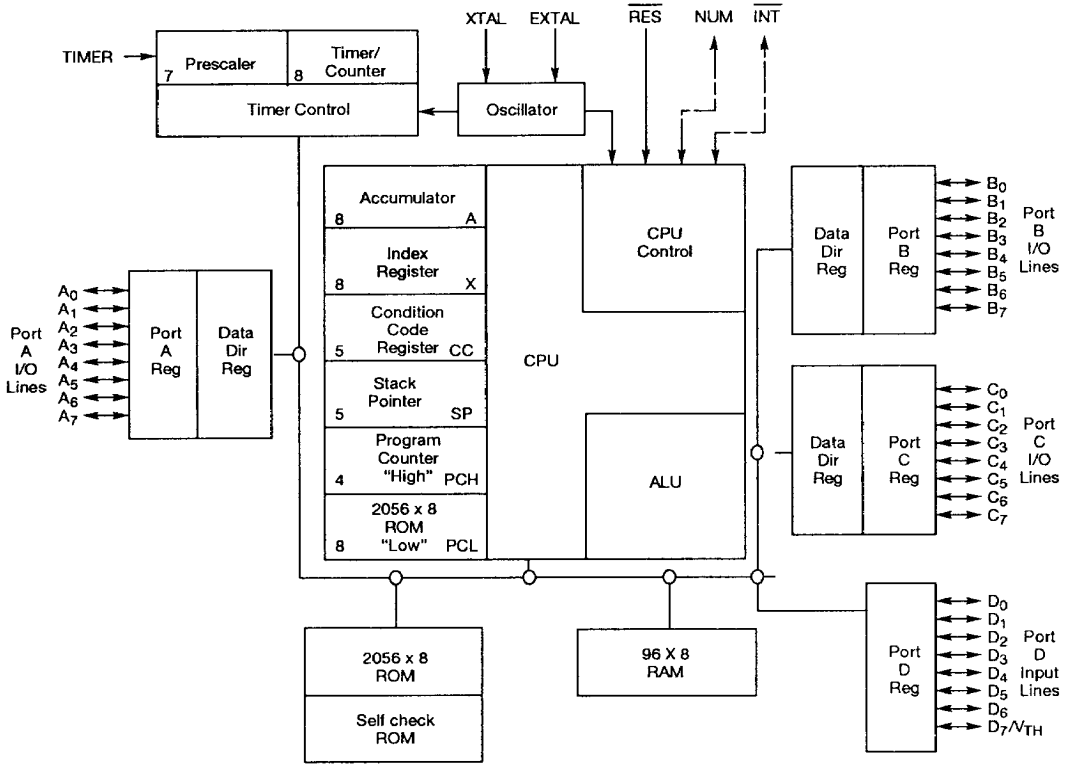


(Top View)

■ PROGRAM DEVELOPMENT SUPPORT TOOLS

- Cross assembler software for use with IBM PCs and compatibles
- In circuit emulator for use with IBM PCs and compatibles

■ **BLOCK DIAGRAM**



■ ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Value	Unit
Supply Voltage	V_{CC} *	-0.3 ~ +7.0	V
Input Voltage (EXCEPT TIMER)	V_{in} *	-0.3 ~ +7.0	V
Input Voltage (TIMER)		-0.3 ~ +12.0	V
Operating Temperature	T_{opr}	0 ~ +70	°C
Storage Temperature	T_{stg}	- 55 ~ +150	°C

* With respect to V_{SS} (SYSTEM GND)

(NOTE) Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating conditions. If these conditions are exceeded, it could affect reliability of LSI.

■ ELECTRICAL CHARACTERISTICS

• DC CHARACTERISTICS (V_{CC}=5.25V ± 0.5V, V_{SS}=GND, Ta=0~+70°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Input "High" Voltage	$\overline{RES}$	V_{IH}		4.0	–	V_{CC}	V
	$\overline{INT}$			3.0	–	V_{CC}	V
	All Other			2.0	–	V_{CC}	V
Input "High" Voltage (Timer)	Timer Mode			2.0	–	V_{CC}	V
	Self-Check Mode			9.0	–	11.0	V
Input "Low" Voltage	$\overline{RES}$	V_{IL}		-0.3	–	0.8	V
	$\overline{INT}$			-0.3	–	0.8	V
	EXTAL(Crystal Mode)			-0.3	–	0.6	V
	All Other			-0.3	–	0.8	V
Power Dissipation		P_D		–	–	700	mW
Low Voltage Recover		LVR		–	–	4.75	V
Low Voltage Inhibit		LVI		–	4.0	–	V
Input Leak Current	TIMER	I_{IL}	V_{in}=0.4V~V_{CC}	-20	–	20	μA
	$\overline{INT}$			-50	–	50	μA
	EXTAL(Crystal Mode)			-1200	–	0	μA

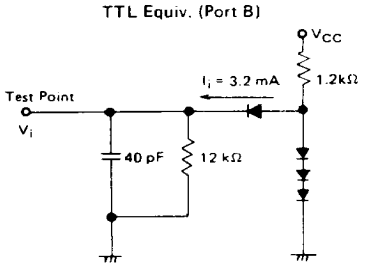
• AC CHARACTERISTICS (V_{CC}=5.25V ± 0.5V, V_{SS}=GND, Ta=0 ~ +70°C, unless otherwise noted.)

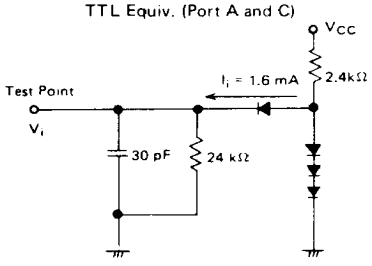
Item		Symbol	Test Condition	min	typ	max	Unit
Clock Frequency		f_{cl}		0.4	–	4.0	MHz
Cycle Time		t_{cyc}		1.0	–	10	μs
Oscillation Frequency (External Resistor Mode)		f_{EXT}	R_{CP}=15.0kΩ±1%	–	3.4	–	MHz
$\overline{INT}$ Pulse Width		t_{IWL}		$\frac{t_{cyc}}{250}$+	–	–	ns
$\overline{RES}$ Pulse Width		t_{RWL}		$\frac{t_{cyc}}{250}$+	–	–	ns
TIMER Pulse Width		t_{TWL}		$\frac{t_{cyc}}{250}$+	–	–	ns
Oscillation Start-up Time (Crystal Mode)		t_{OSC}	C_L=22pF±20%, R_S=60Ω max.	–	–	100	ms
Delay Time Reset		t_{RHL}	External Cap. = 2.2 μF	100	–	–	ms
Input Capacitance	XTAL	C_{in}	V_{in}=0V	–	–	35	pF
	All Other			–	–	10	pF

● PORT ELECTRICAL CHARACTERISTICS (V_{CC} = 5.25V ± 0.5V, V_{SS} = GND, Ta = 0 ~ +70°C, unless otherwise noted.)

Item		Symbol	Test Condition	min	typ	max	Unit
Output "High" Voltage	Port A	V_{OH}	I_{OH} = −10 μA	3.5	–	–	V
			I_{OH} = −100 μA	2.4	–	–	V
	Port B		I_{OH} = −200 μA	2.4	–	–	V
			I_{OH} = −1 mA	1.5	–	–	V
	Port C		I_{OH} = −100 μA	2.4	–	–	V
Output "Low" Voltage	Port A and C	V_{OL}	I_{OL} = 1.6 mA	–	–	0.4	V
	Port B		I_{OL} = 3.2 mA	–	–	0.4	V
			I_{OL} = 10 mA	–	–	1.0	V
Input "High" Voltage	Port A, B, C, and D*	V_{IH}		2.0	–	V_{CC}	V
Input "Low" Voltage		V_{IL}		−0.3	–	0.8	V
Input Leak Current	Port A	I_{IL}	V_{in} = 0.8V	−500	–	–	μA
			V_{in} = 2V	−300	–	–	μA
	Port B, C, and D		V_{in} = 0.4V ~ V_{CC}	− 20	–	20	μA
Input "High" Voltage	Port D** (D_0 ~ D_6)	V_{IH}		–	V_{TH}+0.2	—	V
Input "Low" Voltage	Port D** (D_0 ~ D_6)	V_{IL}		–	V_{TH}-0.2	—	V
Threshold Voltage	Port D**(D_7)	V_{TH}		0	—	0.8×V_{CC}	V

* Port D as digital input
** Port D as analog input





(NOTE) 1. Load capacitance includes the floating capacitance of the probe and the jig etc.
2. All diodes are 1S2074(H) or equivalent.

Figure 1 Bus Timing Test Loads

● **SIGNAL DESCRIPTION**

The input and output signals for the MCU, shown in PIN ARRANGEMENT, are described in the following paragraphs.

● **V_{CC} and V_{SS}**

Power is supplied to the MCU using these two pins. V_{CC} is +5.25V ±0.5V. V_{SS} is the ground connection.

● **$\overline{INT}$**

This pin provides the capability for asynchronously applying an external interrupt to the MCU. Refer to INTERRUPTS for additional information.

● **XTAL and EXTAL**

These pins provide connections for the on-chip clock circuit. A crystal (AT cut, 4 MHz maximum), a resistor or an external signal can be connected to these pins to provide a system clock with various stability/cost tradeoffs. Refer to INTERNAL OSCILLATOR OPTIONS for recommendations about these inputs.

● **TIMER**

This pin allows an external input to be used to decrement the internal timer circuitry. Refer to TIMER for additional information about the timer circuitry.

● **$\overline{RES}$**

This pin allows resetting of the MCU at times other than the automatic resetting capability already in the MCU. Refer to RESETS for additional information.

● **NUM**

This pin is not for user application and should be connected to V_{SS}.

- **Input/Output Lines ($A_0 \sim A_7$, $B_0 \sim B_7$, $C_0 \sim C_7$)**

These 24 lines are arranged into three 8-bit ports (A, B and C). All lines are programmable as either inputs or outputs under software control of the Data Direction Register (DDR). Refer to INPUT/OUTPUT for additional information.

- **Input Lines ($D_0 \sim D_7$)**

These are 8-bit input lines, which has two functions. Firstly, these are TTL compatible inputs, in location $003. The other function is 7 bits comparator, in location $007. Refer to INPUT for more details.

■ MEMORY

The MCU memory is configured as shown in Figure 2. During the processing of an interrupt, the contents of the CPU registers are pushed onto the stack in the order shown in Figure 3. Since the stack pointer decrements during pushes, the low order byte (PCL) of the program counter is stacked first; then the high order four bits (PCH) are stacked. This ensures that the program counter is loaded correctly as the stack pointer increments when it pulls data from the stack. A subroutine call will cause only the program counter (PCH, PCL) contents to be pushed onto the stack.

Caution: – Self Test ROM Address Area
Self test ROM locations can not be used for a user program. If the user's program is in this location, it will be removed when manufacturing mask for production.

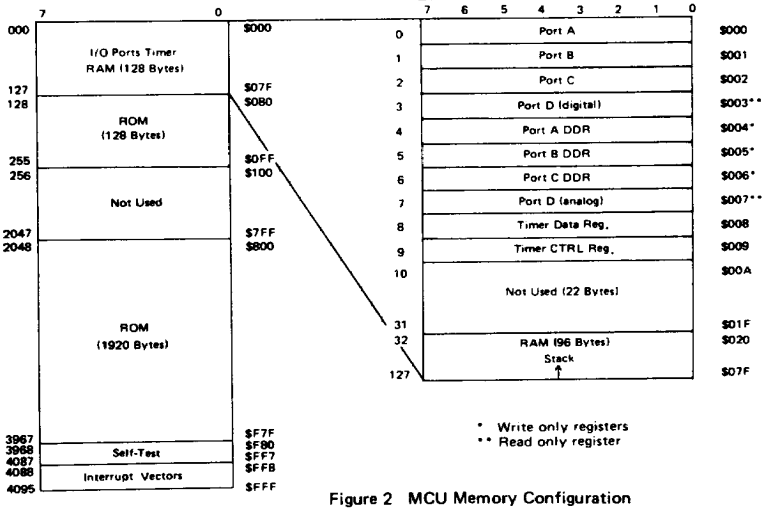


Figure 2 MCU Memory Configuration

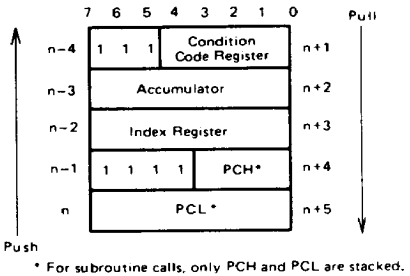


Figure 3 Interrupt Stacking Order

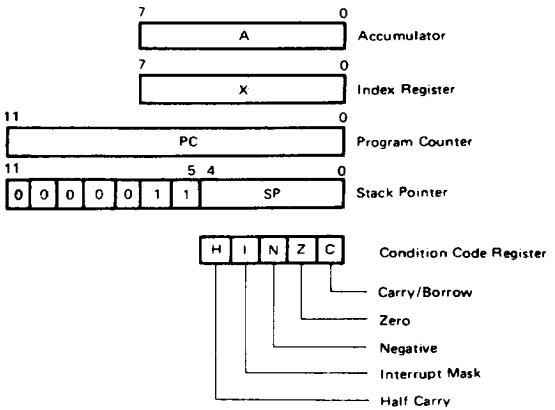


Figure 4 Programming Model

■ REGISTERS

The CPU has five registers available to the programmer. They are shown in Figure 4 and are explained in the following paragraphs.

- **Accumulator (A)**

The accumulator is a general purpose 8-bit register used to hold operands and results of arithmetic calculations or data manipulations.

- **Index Register (X)**

The index register is an 8-bit register used for the indexed addressing mode. It contains an 8-bit address that may be added to an offset value to create an effective address. The index register can also be used for limited calculations and data manipulations when using read/modify/write instructions. When not required by a code sequence being executed, the index register can be used as a temporary storage area.

- **Program Counter (PC)**

The program counter is a 12-bit register that contains the address of the next instruction to be executed.

- **Stack Pointer (SP)**

The stack pointer is a 12-bit register that contains the address of the next free location on the stack. Initially, the stack pointer is set to location $07F and is decremented as data is being pushed onto the stack and incremented as data is being pulled from the stack. The six most significant bits of the stack pointer are permanently set to 0000011. During an MCU reset or the reset stack pointer (RSP) instruction, the stack pointer is set to location $07F. Subroutines and interrupts may be nested down to location $061 which allows the programmer to use up to 15 levels of subroutine calls.

- **Condition Code Register (CC)**

The condition code register is a 5-bit register in which each bit is used to indicate or flag the results of the instruction just executed. These bits can be individually tested by a program and specific action taken as a result of their state. Each individual condition code register bit is explained in the following paragraphs.

Half Carry (H)

Used during arithmetic operations (ADD and ADC) to indicate that a carry occurred between bits 3 and 4.

Interrupt (I)

This bit is set to mask the timer and external interrupt ($\overline{INT}$). If an interrupt occurs while this bit is set it is latched and will be processed as soon as the interrupt bit is reset.

Negative (N)

Used to indicate that the result of the last arithmetic, logical or data manipulation was negative (bit 7 in result equal to a logical one).

Zero (Z)

Used to indicate that the result of the last arithmetic, logical or data manipulation was zero.

Carry/Borrow (C)

Used to indicate that a carry or borrow out of the arithmetic logic unit (ALU) occurred during the last arithmetic operation. This bit is also affected during bit test and branch instructions, shifts, and rotates.

■ TIMER

The MCU timer circuitry is shown in Figure 5. The 8-bit counter, the Timer Data Register (TDR), is loaded under program control and counts down toward zero as soon as the clock input is applied. When the timer reaches zero, the timer interrupt request bit (bit 7) in the Timer Control Register (TCR) is set. The CPU responds to this interrupt by saving the present CPU state on the stack, fetching the timer interrupt vector from locations $FF8 and $FF9 and executing the interrupt routine. The timer interrupt can be masked by setting the timer interrupt mask bit (bit 6) in the TCR. The interrupt bit (I bit) in the Condition Code Register also prevents a time interrupt from being processed.

The clock input to the timer can be from an external source applied to the TIMER input pin or it can be the internal ϕ_2 signal. When the internal ϕ_2 signal is selected as the input source, the node a is connected to b (see Fig. 5). In case of the external source, the node b connects with c. When the ϕ_2 signal is used as the source, the clock signal is input to the prescaler while the TIMER input is "High". The source of the clock input is one of the options that has to be specified before manufac-

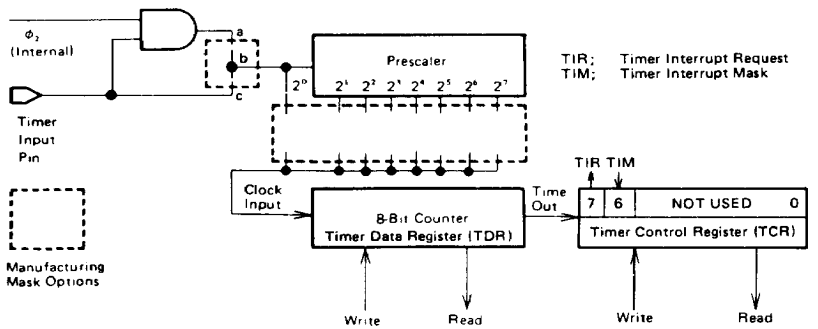


Figure 5 Timer Block Diagram

ture of the MCU. A prescaler option can be applied to the clock input that extends the timing interval up to a maximum of 128 counts before decrementing the counter (TDR). The timer continues to count past zero, falling through to $FF from zero and then continuing the count. Thus, the counter (TDR) can be read at any time by reading the TDR. This allows a program to determine the length of time since a timer interrupt has occurred and not disturb the counting process.

The TDR is 8-bit Read/Write Register in location $008. At power-up or reset, the TDR and the prescaler are initialized with all logical ones.

The Timer Interrupt Request bit (bit 7 of the TCR) is set by hardware when timer count reaches zero, and is cleared by program or by hardware reset. The bit 6 of the TCR is writable by program. Both of those bits can be read by CPU.

(NOTE) If the MCU Timer is not used, the TIMER input pin must be grounded.

■ SELF CHECK

The self-check capability of the MCU provides an internal check to determine if the part is functional. Connect the MCU as shown in Figure 6 and monitor the output of port C bit 3 for an oscillation of approximately 3Hz. ROM, RAM, TIMER, Interrupts, I/O of Port A, B and C are checked by this capability.

■ RESETS

The MCU can be reset three ways; by initial power-up, by the external reset input ($\overline{RES}$) and by an optional internal low voltage inhibit circuit, see Figure 7. All the I/O port are initialized to input mode (DDRs are cleared) during reset.

During power-up, a minimum of 100 milliseconds is needed before allowing the $\overline{RES}$ input to go "High".

This time allows the internal crystal oscillator to stabilize. Connecting a capacitor to the $\overline{RES}$ input, as shown in Figure 8, typically provides sufficient delay.

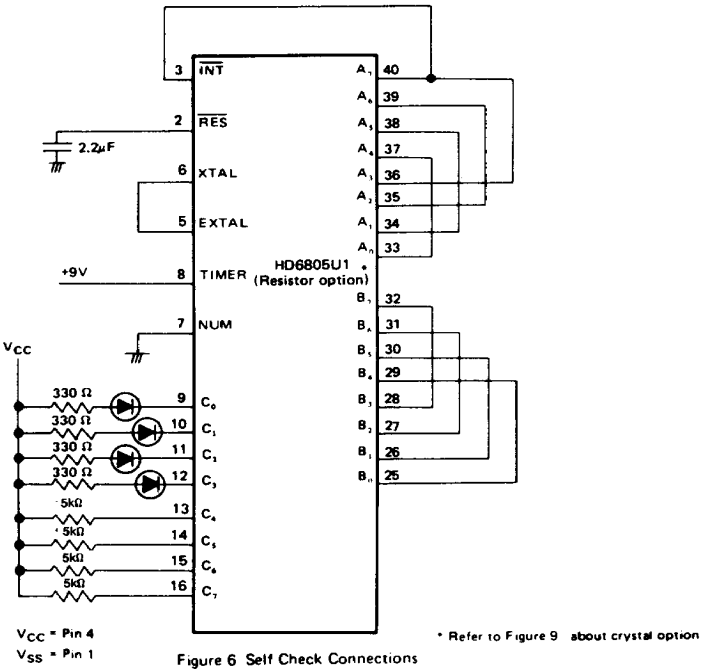


Figure 6 Self Check Connections

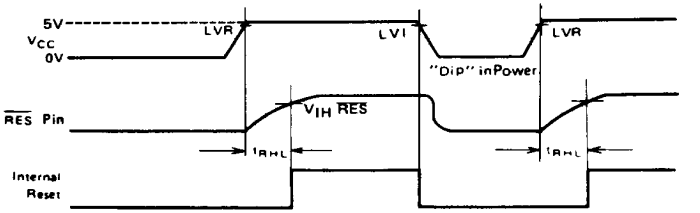


Figure 7 Power Up and $\overline{RES}$ Timing

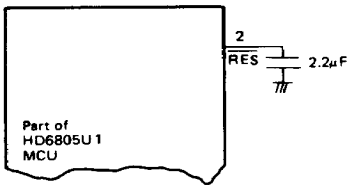


Figure 8 Power Up Reset Delay Circuit

■ INTERNAL OSCILLATOR OPTIONS

The internal oscillator circuit is designed to require a minimum of external components. A crystal, a resistor, a jumper wire, or an external signal may be used to generate a system clock with various stability/cost tradeoff. A manufacturing mask option is required to select either the crystal oscillator or the RC oscillator circuit. The different connection methods are shown in Figure 9. Crystal specifications are given in Figure 10. A resistor selection graph is given in Figure 11.

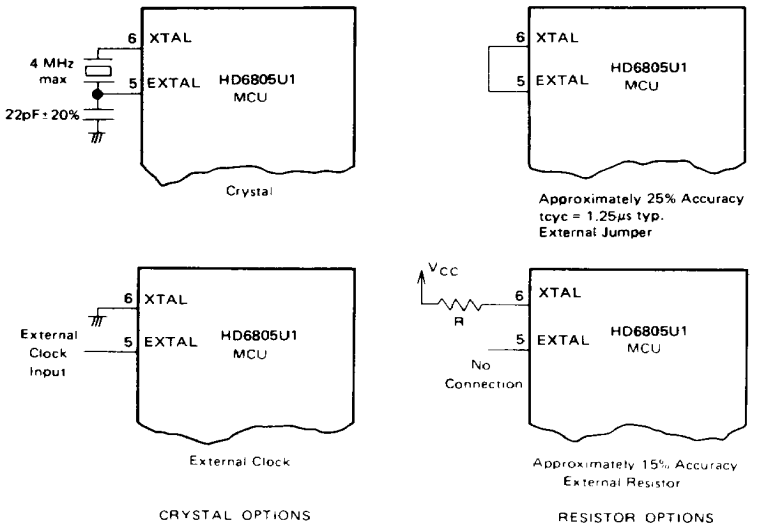
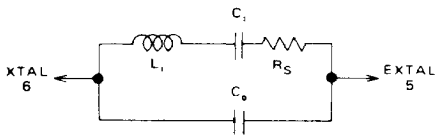


Figure 9 Internal Oscillator Options



AT – Cut Parallel Resonance Crystal
C_0 = 7 pF max.
f = 4 MHz
R_S = 60 Ω max.

Figure 10 Crystal Parameters

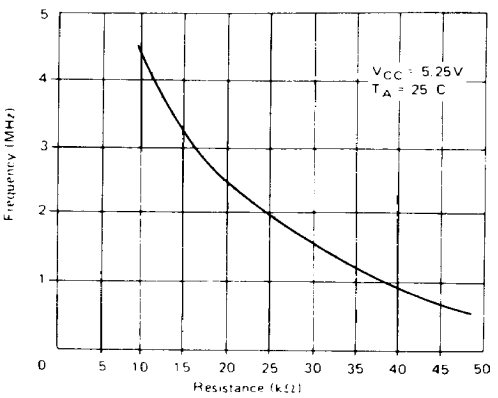


Figure 11 Typical Resistor Selection Graph

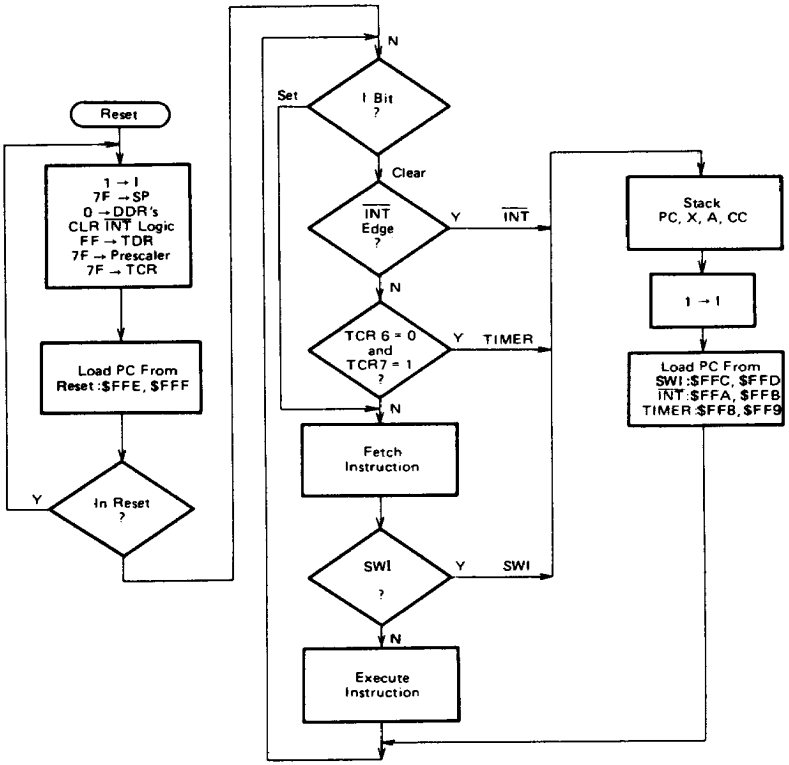
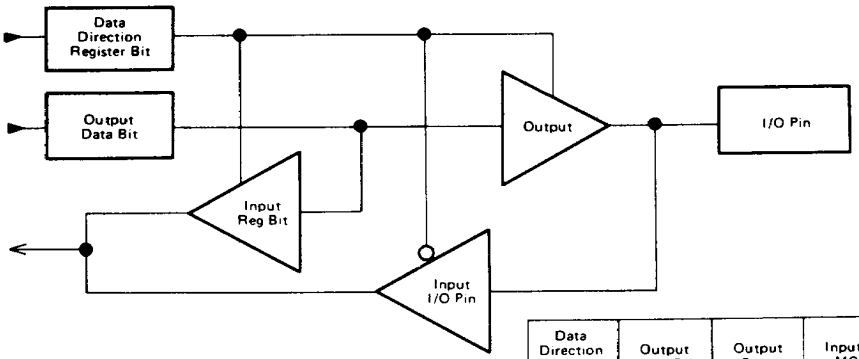


Figure 12 Interrupt Processing Flowchart



Data Direction Register Bit	Output Data Bit	Output State	Input to MCU
1	0	0	0
1	1	1	1
0	x	3-State	Pin

Figure 13 Typical Port I/O Circuitry

■ INTERRUPTS

The CPU can be interrupted three different ways: through the external interrupt ($\overline{INT}$) input pin, the internal timer interrupt request, and a software interrupt instruction (SWI). When any interrupt occurs, processing is suspended, the present CPU state is pushed onto the stack, the interrupt bit (I) in the Condition Code Register is set, the address of the interrupt routine is obtained from the appropriate interrupt vector address, and the interrupt routine is executed. Stacking the CPU registers, setting the I bit, and vector fetching requires 11 cycles. The interrupt service routines normally end with a return from interrupt (RTI) instruction which allows the CPU to resume processing of the program prior to the interrupt. Table 1 provides a listing of the interrupts, their priority, and the vector address that contain the starting address of the appropriate interrupt routine.

A flowchart of the interrupt processing sequence is given in Fig. 12.

Table 1 Interrupt Priorities

Interrupt	Priority	Vector Address
$\overline{RES}$	1	$FFE and $FFF
SWI	2	$FFC and $FFD
$\overline{INT}$	3	$FFA and $FFB
TIMER	4	$FF8 and $FF9

■ INPUT/OUTPUT

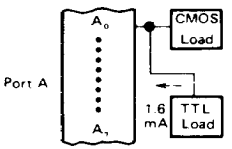
There are 24 input/output pins. All pins are programmable as either inputs or outputs under software control of the corresponding Data Direction Register (DDR). When programmed as outputs, the latched output data is readable as input data, regardless of the logic levels at the output pin due to output loading (see Fig. 13). When Port B is programmed for outputs, it is capable of sinking 10mA on each pin (V_{OL} = 1V max). All input/output lines are TTL compatible as both inputs and outputs. Port A is CMOS compatible as outputs, and Port B and C lines are CMOS compatible as inputs. Figure 14 provides some examples of port connections.

■ INPUT

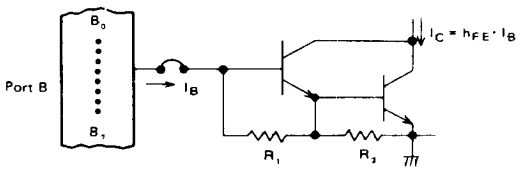
Port D can be used as either 8 TTL compatible inputs or 1 threshold input and 7 analog inputs pins. Fig. 15 (a) shows the construction of port D. The Port D register at location $003 stores TTL compatible inputs, and those in location $007 store the result of comparison D_0 to D_6 inputs with D_7 threshold input. Port D has not only the conventional function as inputs but also voltage-comparison function. Applying the latter, can easily check that 7 analog input electric potential max. exceeds the limit with the construction shown in Fig. 15 (b). Also, using one output pin of MCU, after external capacity is discharged at the preset state, charge the CR circuit of long enough time constant, apply the charging curve to the D_7 pin. The construction described above is shown in Fig. 15 (c). The compared result of D_0 to D_6 is regularly monitored, which gives the analog input electric potential applied to D_0 to D_6 pins from inverted time. This method enables 7 inputs to be converted from analog to digital. Furthermore, combination of two functions gives 3 level voltages from D_0 to D_6. Fig. 15 (d) provides the example when V_{TH} is set to 3.5V.

■ BIT MANIPULATION

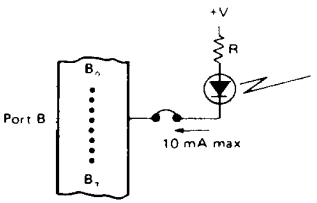
The MCU has the ability to set or clear any single random access memory or input/output bit (except the data direction registers) with a single instruction (BSET, BCLR). Any bit in the page zero read only memory can be tested, using the BRSET and BRCLR instructions, and the program branches as a result of its state. This capability to work with any bit in RAM, ROM or I/O allows the user to have individual flags in RAM or to handle single I/O bits as control lines. The example in Figure 16 illustrates the usefulness of the bit manipulation and test



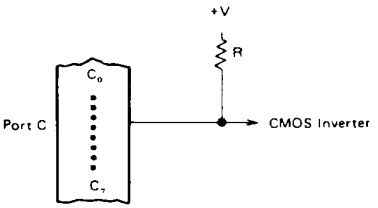
Port A Programmed as output(s), driving CMOS and TTL Load directly.
(a)



Port B Programmed as output(s), driving Darlington base directly.
(b)



Port B Programmed as output(s), driving LED(s) directly.
(c)

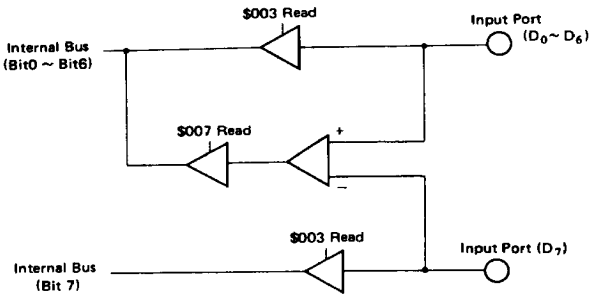


Port C Programmed as output(s), driving CMOS loads, using external pull-up
(d)

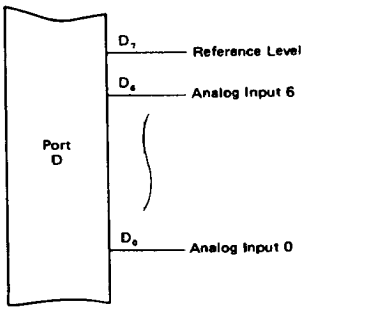
Figure 14 Typical Port Connections

instructions. Assume that bit 0 of port A is connected to a zero crossing detector circuit and that bit 1 of port A is connected to the trigger of a TRIAC which power the controlled hardware.

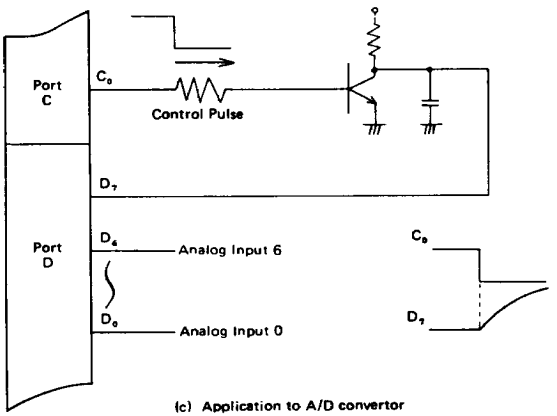
This program, which uses only seven ROM locations, provides turn-on of the TRIAC within 14 microseconds of the zero crossing. The timer could also be incorporated to provide turn-on at some later time which would permit pulse-width modulation of the controlled power.



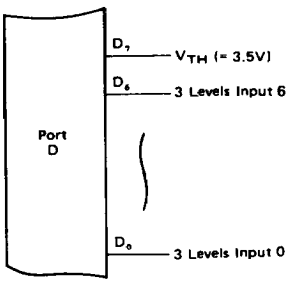
(a) The logic configuration of Port D



(b) Seven analog inputs and a reference level input of Port D



(c) Application to A/D convertor



(d) Application to 3 levels input

Input Voltage	($003)	($007)
0V ~ 0.8V	0	0
2.0V ~ 3.3V	1	0
3.7V ~ V_{CC}	1	1

Figure 15 Configuration and Application of Port D

```
SELF 1    BRCLR 0, PORT A, SELF 1
          BSET 1, PORT A
          BCLR 1, PORT A
```

Figure 16 Bit Manipulation Example

■ ADDRESSING MODES

The CPU has ten addressing modes available for use by the programmer. They are explained and illustrated briefly in the following paragraphs.

- **Immediate**

Refer to Figure 17. The immediate addressing mode accesses constants which do not change during program execution. Such instructions are two bytes long. The effective address (EA) is the PC and the operand is fetched from the byte following the opcode.

- **Direct**

Refer to Figure 18. In direct addressing, the address of the operand is contained in the second byte of the instruction. Direct addressing allows the user to directly address the lowest 256 bytes in memory. All RAM space, I/O registers and 128 bytes of ROM are located in page zero to take advantage of this efficient memory addressing mode.

- **Extended**

Refer to Figure 19. Extended addressing is used to reference any location in memory space. The EA is the contents of the two bytes following the opcode. Extended addressing instructions are three bytes long.

- **Relative**

Refer to Figure 20. The relative addressing mode applies only to the branch instructions. In this mode the contents of the byte following the opcode is added to the program counter when the branch is taken. EA=(PC)+2+Rel. Rel is the contents of the location following the instruction opcode with bit 7 being the sign bit. If the branch is not taken Rel=0, when a branch takes place, the program goes to somewhere within the range of +129 bytes to -127 of the present instruction. These instructions are two bytes long.

- **Indexed (No Offset)**

Refer to Figure 21. This mode of addressing accesses the lowest 256 bytes of memory. These instructions are one byte long and their EA is the contents of the index register.

- **Indexed (8-bit Offset)**

Refer to Figure 22. The EA is calculated by adding the contents of the byte following the opcode to the contents of the index register. In this mode, 511 low memory locations are accessable. These instructions occupy two bytes.

- **Indexed (16-bit Offset)**

Refer to Figure 23. This addressing mode calculates the EA by adding the contents of the two bytes following the opcode to the index register. Thus, the entire memory space may be accessed. Instructions which use this addressing mode are three bytes long.

- **Bit Set/Clear**

Refer to Figure 24. This mode of addressing applies to instructions which can set or clear any bit on page zero. The lower three bits in the opcode specify the bit to be set or cleared while the byte following the opcode specifies the address in page zero.

- **Bit Test and Branch**

Refer to Figure 25. This mode of addressing applies to instructions which can test any bit in the first 256 locations ($00-$FF) and branch to any location relative to the PC. The byte to be tested is addressed by the byte following the opcode. The individual bit within that byte to be tested is addressed by the lower three bits of the opcode. The third byte is the relative address to be added to the program counter if the branch condition is met. These instructions are three bytes long. The value of the bit tested is written to the carry bit in the condition code register.

- **Implied**

Refer to Figure 26. The implied mode of addressing has no EA. All the information necessary to execute an instruction is contained in the opcode. Direct operations on the accumulator and the index register are included in this mode of addressing. In addition, control instructions such as SWI, RTI belong to this group. All implied addressing instructions are one byte long.

■ INSTRUCTION SET

The MCU has a set of 59 basic instructions. They can be divided into five different types: register/memory, read/modify/write, branch, bit manipulation, and control. The following paragraphs briefly explain each type. All the instructions within a given type are presented in individual tables.

- **Register/Memory Instructions**

Most of these instructions use two operands. One operand is either the accumulator or the index register. The other operand is obtained from memory using one of the addressing modes. The jump unconditional (JMP) and jump to subroutine (JSR) instructions have no register operand. Refer to Table 2.

- **Read/Modity/Write Instructions**

These instructions read a memory location or a register, modify or test its contents, and write the modified value back to memory or to the register. The test for negative or zero (TST) instruction is an exception to the read/modify/write instructions since it does not perform the write. Refer to Table 3.

- **Branch Instructions**

The branch instructions cause a branch from the program when a certain condition is met. Refer to Table 4.

- **Bit Manipulation Instructions**

These instructions are used on any bit in the first 256 bytes of the memory. One group either sets or clears. The other group performs the bit test and branch operations. Refer to Table 5.

- **Control Instructions**

The control instructions control the MCU operations during program execution. Refer to Table 6.

- **Alphabetical Listing**

The complete instruction set is given in alphabetical order in Table 7.

- **Opcode Map**

Table 8 is an opcode map for the instructions used on the MCU.

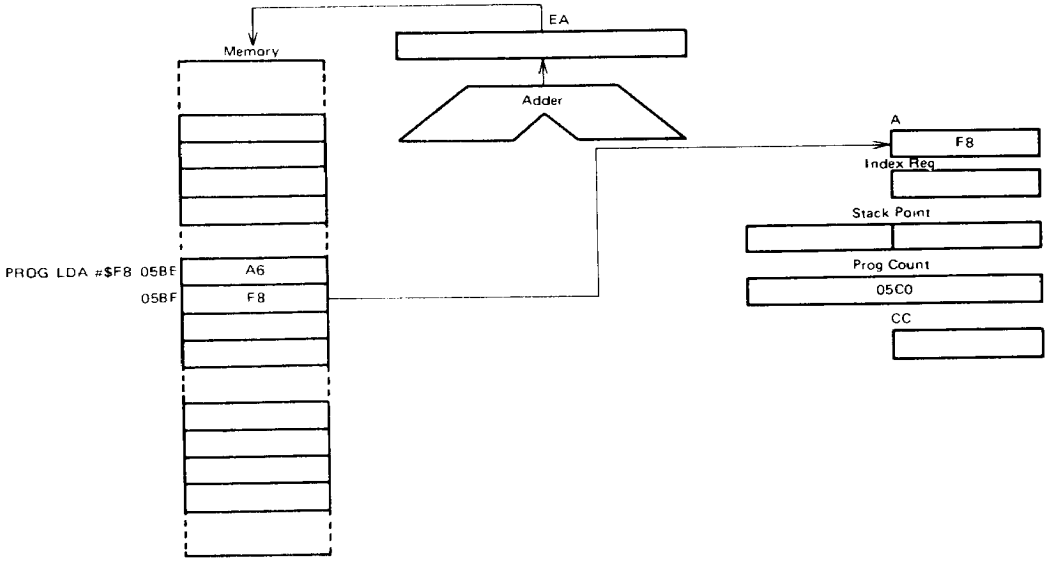


Figure 17 Immediate Addressing Example

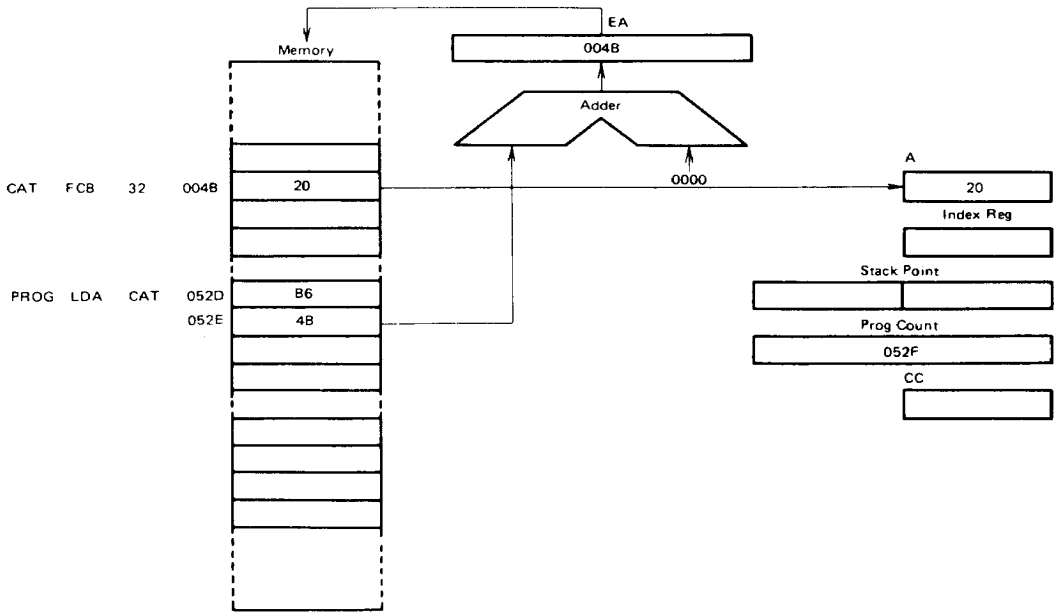


Figure 18 Direct Addressing Example

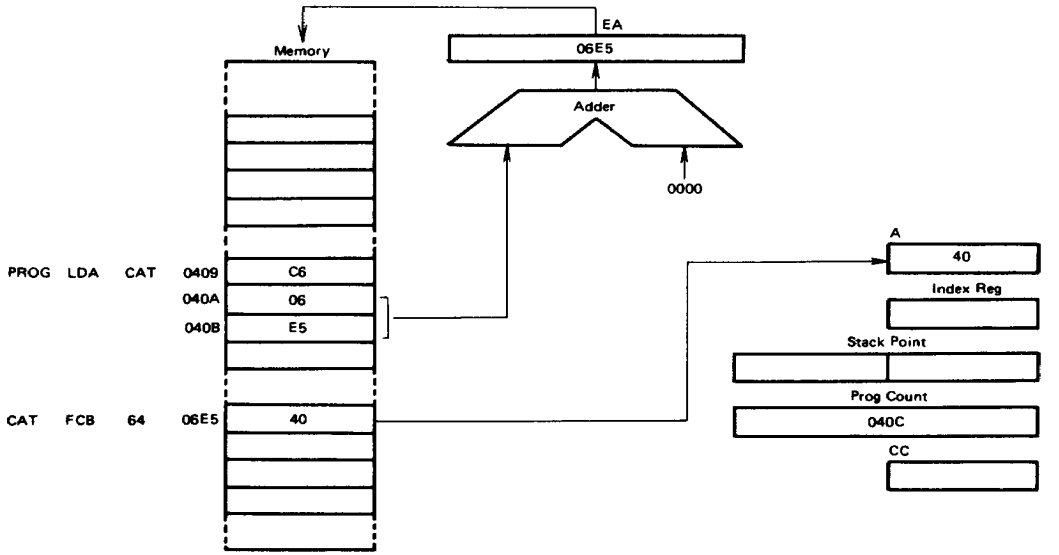


Figure 19 Extended Addressing Example

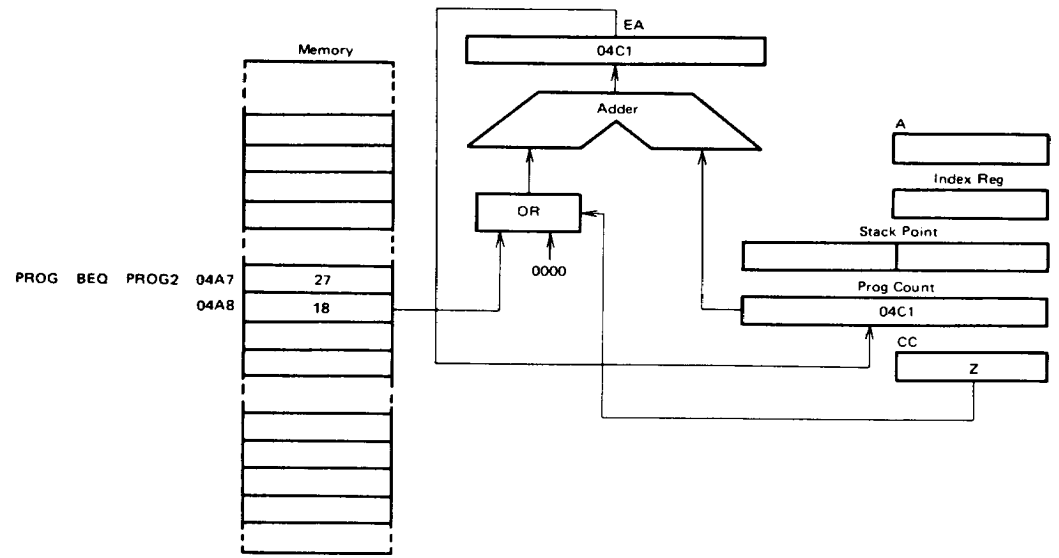


Figure 20 Relative Addressing Example

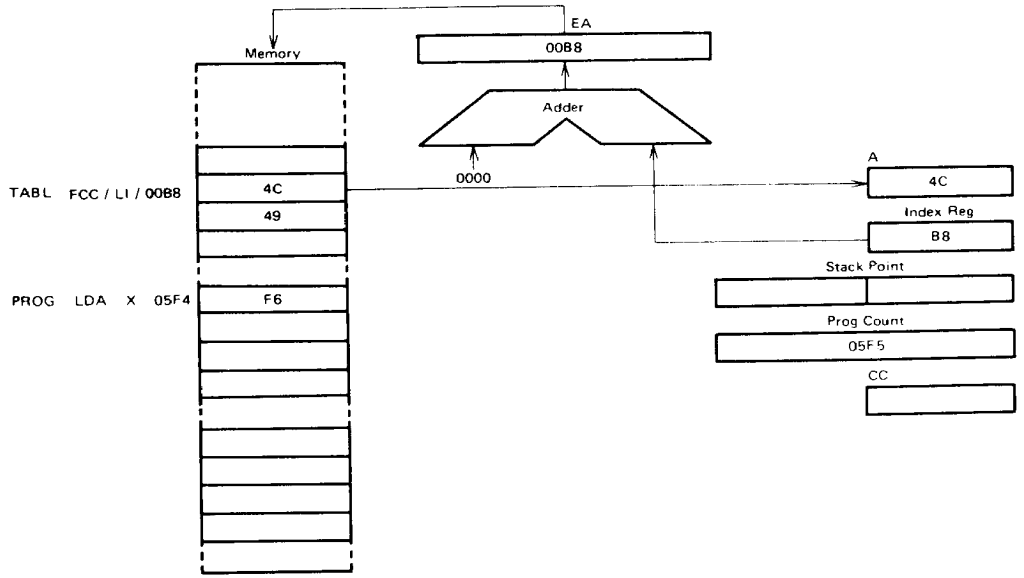


Figure 21 Indexed (No Offset) Addressing Example

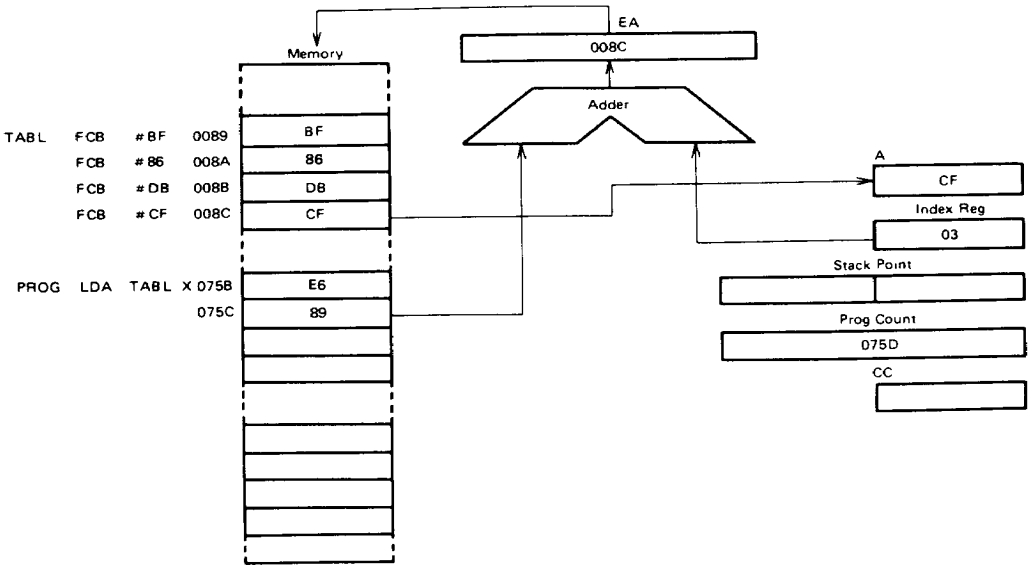


Figure 22 Indexed (8-Bit Offset) Addressing Example

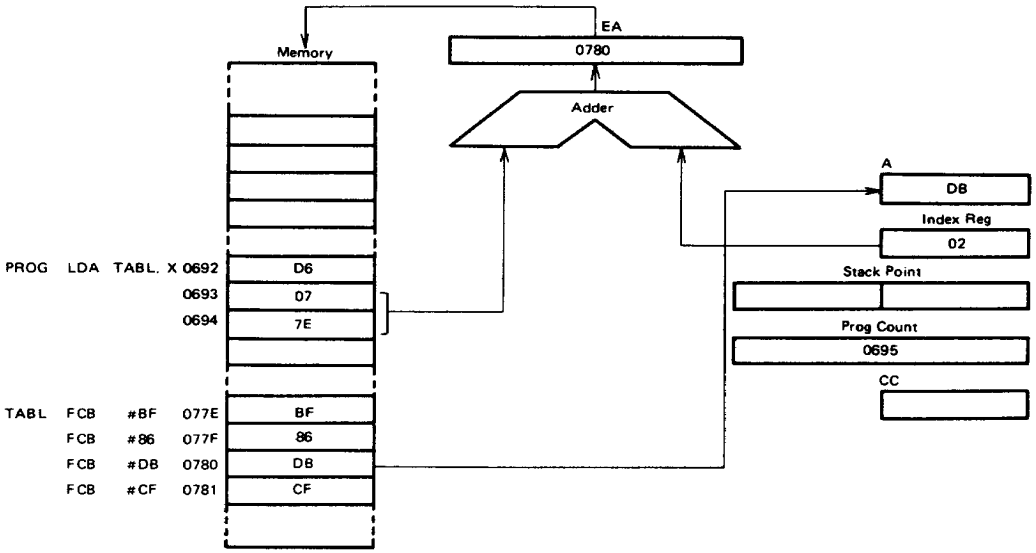


Figure 23 Indexed (16-Bit Offset) Addressing Example

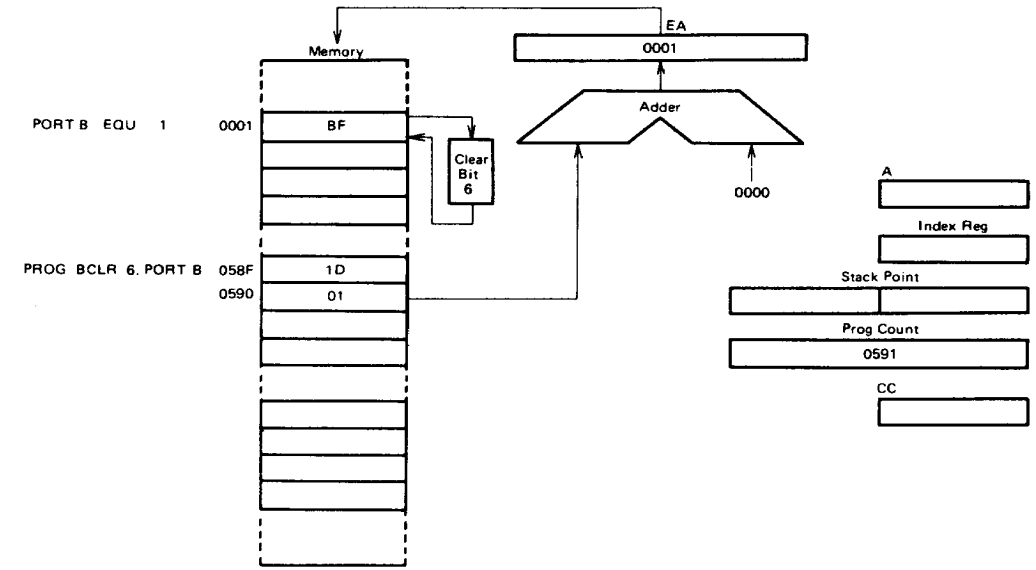


Figure 24 Bit Set/Clear Addressing Example

HITACHI
Hitachi America, Ltd. • Hitachi Plaza • 2000 Sierra Point Pkwy. • Brisbane, CA 94005-1819 • (415) 589-8300

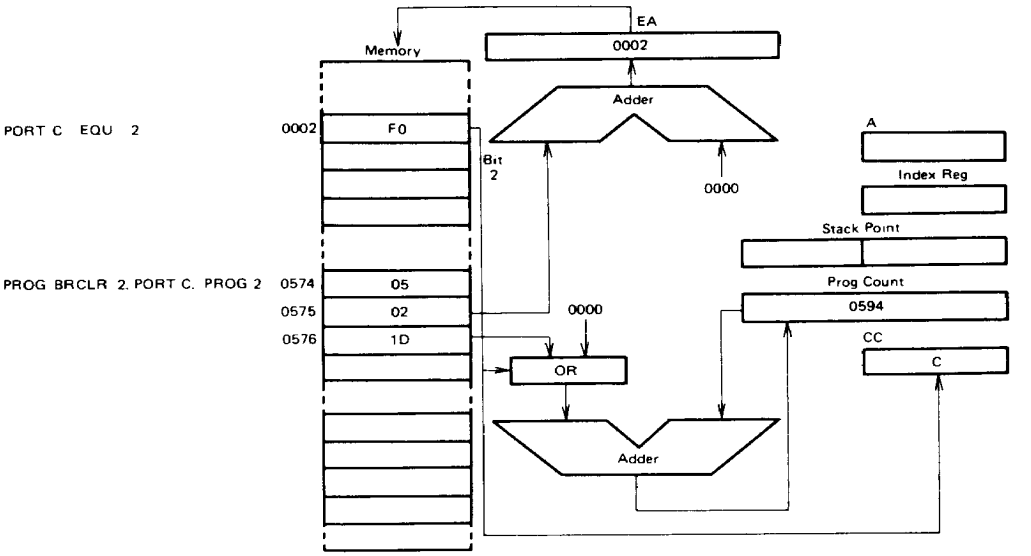


Figure 25 Bit Test and Branch Addressing Example

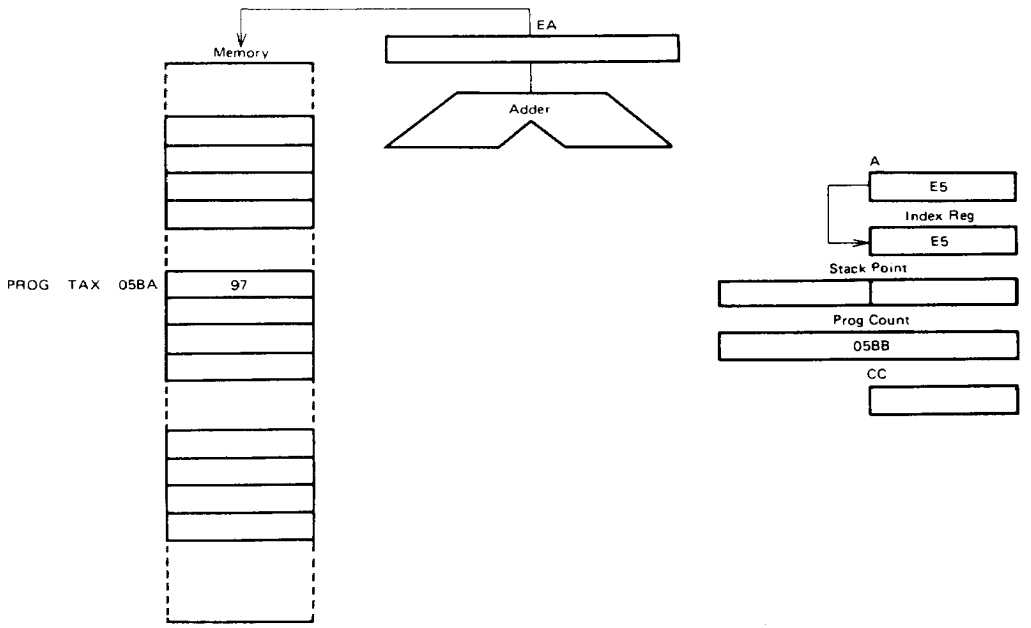


Figure 26 Implied Addressing Example

Table 2 Register/Memory Instructions

Function	Mnemonic	Addressing Modes																	
		Immediate			Direct			Extended			Indexed (No Offset)			Indexed (8-Bit Offset)			Indexed (16-Bit Offset)		
		Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles
Load A from Memory	LDA	A6	2	2	B6	2	4	C6	3	5	F6	1	4	E6	2	5	D6	3	6
Load X from Memory	LDX	AE	2	2	BE	2	4	CE	3	5	FE	1	4	EE	2	5	DE	3	6
Store A in Memory	STA	–	–	–	B7	2	5	C7	3	6	F7	1	5	E7	2	6	D7	3	7
Store X in Memory	STX	–	–	–	BF	2	5	CF	3	6	FF	1	5	EF	2	6	DF	3	7
Add Memory to A	ADD	AB	2	2	BB	2	4	CB	3	5	FB	1	4	EB	2	5	DB	3	6
Add Memory and Carry to A	ADC	A9	2	2	B9	2	4	C9	3	5	F9	1	4	E9	2	5	D9	3	6
Subtract Memory	SUB	A0	2	2	B0	2	4	C0	3	5	F0	1	4	E0	2	5	D0	3	6
Subtract Memory from A with Borrow	SBC	A2	2	2	B2	2	4	C2	3	5	F2	1	4	E2	2	5	D2	3	6
AND Memory to A	AND	A4	2	2	B4	2	4	C4	3	5	F4	1	4	E4	2	5	D4	3	6
OR Memory with A	ORA	AA	2	2	BA	2	4	CA	3	5	FA	1	4	EA	2	5	DA	3	6
Exclusive OR Memory with A	EOR	A8	2	2	B8	2	4	C8	3	5	F8	1	4	E8	2	5	D8	3	6
Arithmetic Compare A with Memory	CMP	A1	2	2	B1	2	4	C1	3	5	F1	1	4	E1	2	5	D1	3	6
Arithmetic Compare X with Memory	CPX	A3	2	2	B3	2	4	C3	3	5	F3	1	4	E3	2	5	D3	3	6
Bit Test Memory with A (Logical Compare)	BIT	A5	2	2	B5	2	4	C5	3	5	F5	1	4	E5	2	5	D5	3	6
Jump Unconditional	JMP	–	–	–	BC	2	3	CC	3	4	FC	1	3	EC	2	4	DC	3	5
Jump to Subroutine	JSR	–	–	–	BD	2	7	CD	3	8	FD	1	7	ED	2	8	DD	3	9

Table 3 Read/Modify/Write Instructions

Function	Mnemonic	Addressing Modes														
		Implied (A)			Implied (X)			Direct			Indexed (No Offset)			Indexed (8-Bit Offset)		
		Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles
Increment	INC	4C	1	4	5C	1	4	3C	2	6	7C	1	6	6C	2	7
Decrement	DEC	4A	1	4	5A	1	4	3A	2	6	7A	1	6	6A	2	7
Clear	CLR	4F	1	4	5F	1	4	3F	2	6	7F	1	6	6F	2	7
Complement	COM	43	1	4	53	1	4	33	2	6	73	1	6	63	2	7
Negate (2's Complement)	NEG	40	1	4	50	1	4	30	2	6	70	1	6	60	2	7
Rotate Left Thru Carry	ROL	49	1	4	59	1	4	39	2	6	79	1	6	69	2	7
Rotate Right Thru Carry	ROR	46	1	4	56	1	4	36	2	6	76	1	6	66	2	7
Logical Shift Left	LSL	48	1	4	58	1	4	38	2	6	78	1	6	68	2	7
Logical Shift Right	LSR	44	1	4	54	1	4	34	2	6	74	1	6	64	2	7
Arithmetic Shift Right	ASR	47	1	4	57	1	4	37	2	6	77	1	6	67	2	7
Arithmetic Shift Left	ASL	48	1	4	58	1	4	38	2	6	78	1	6	68	2	7
Test for Negative or Zero	TST	4D	1	4	5D	1	4	3D	2	6	7D	1	6	6D	2	7

Table 4 Branch Instructions

Function	Mnemonic	Relative Addressing Mode		
		Op Code	# Bytes	# Cycles
Branch Always	BRA	20	2	4
Branch Never	BRN	21	2	4
Branch IF Higher	BHI	22	2	4
Branch IF Lower or Same	BLS	23	2	4
Branch IF Carry Clear	BCC	24	2	4
(Branch IF Higher or Same)	(BHS)	24	2	4
Branch IF Carry Set	BCS	25	2	4
(Branch IF Lower)	(BLO)	25	2	4
Branch IF Not Equal	BNE	26	2	4
Branch IF Equal	BEQ	27	2	4
Branch IF Half Carry Clear	BHCC	28	2	4
Branch IF Half Carry Set	BHCS	29	2	4
Branch IF Plus	BPL	2A	2	4
Branch IF Minus	BMI	2B	2	4
Branch IF Interrupt Mask Bit is Clear	BMC	2C	2	4
Branch IF Interrupt Mask Bit is Set	BMS	2D	2	4
Branch IF Interrupt Line is Low	BIL	2E	2	4
Branch IF Interrupt Line is High	BIH	2F	2	4
Branch to Subroutine	BSR	AD	2	8

Table 5 Bit Manipulation Instructions

Function	Mnemonic	Addressing Modes					
		Bit Set/Clear			Bit Test and Branch		
		Op Code	# Bytes	# Cycles	Op Code	# Bytes	# Cycles
Branch IF Bit n is set	BRSET n (n=0 ..... 7)	–	–	–	2•n	3	10
Branch IF Bit n is clear	BRCLR n (n=0 .....7)	–	–	–	01+2•n	3	10
Set Bit n	BSET n (n=0 ..... 7)	10+2•n	2	7	–	–	–
Clear bit n	BCLR n (n=0 ..... 7)	11+2•n	2	7	–	–	–

Table 6 Control Instructions

Function	Mnemonic	Implied		
		Op Code	# Bytes	# Cycles
Transfer A to X	TAX	97	1	2
Transfer X to A	TXA	9F	1	2
Set Carry Bit	SEC	99	1	2
Clear Carry Bit	CLC	98	1	2
Set Interrupt Mask Bit	SEI	9B	1	2
Clear Interrupt Mask Bit	CLI	9A	1	2
Software Interrupt	SWI	83	1	11
Return from Subroutine	RTS	81	1	6
Return from Interrupt	RTI	80	1	9
Reset Stack Pointer	RSP	9C	1	2
No-Operation	NOP	9D	1	2

Table 7 Instruction Set

Mnemonic	Addressing Modes										Condition Code				
	Implied	Immediate	Direct	Extended	Relative	Indexed (No Offset)	Indexed (8 Bits)	Indexed (16 Bits)	Bit Set/ Clear	Bit Test & Branch	H	I	N	Z	C
ADC		x	x	x		x	x	x			^	•	^	^	^
ADD		x	x	x		x	x	x			^	•	^	^	^
AND		x	x	x		x	x	x			•	•	^	^	•
ASL	x		x			x	x				•	•	^	^	^
ASR	x		x			x	x				•	•	^	^	^
BCC					x						•	•	•	•	•
BCLR									x		•	•	•	•	•
BCS					x						•	•	•	•	•
BEQ					x						•	•	•	•	•
BHCC					x						•	•	•	•	•
BHCS					x						•	•	•	•	•
BHI					x						•	•	•	•	•
BHS					x						•	•	•	•	•
BIH					x						•	•	•	•	•
BIL					x						•	•	•	•	•
BIT		x	x	x		x	x	x			•	•	^	^	•
BLO					x						•	•	•	•	•
BLS					x						•	•	•	•	•
BMC					x						•	•	•	•	•
BMI					x						•	•	•	•	•
BMS					x						•	•	•	•	•
BNE					x						•	•	•	•	•
BPL					x						•	•	•	•	•
BRA					x						•	•	•	•	•
BRN					x						•	•	•	•	•
BRCLR										x	•	•	•	•	^
BRSET										x	•	•	•	•	^
BSET									x		•	•	•	•	•
BSR					x						•	•	•	•	•
CLC	x										•	•	•	•	0
CLI	x										•	0	•	•	•
CLR	x		x			x	x				•	•	0	1	•
CMP		x	x	x		x	x	x			•	•	^	^	^
COM	x		x			x	x				•	•	^	^	1
CPX		x	x	x		x	x	x			•	•	^	^	^
DEC	x		x			x	x				•	•	^	^	•
EOR		x	x	x		x	x	x			•	•	^	^	•
INC	x		x			x	x				•	•	^	^	•
JMP			x	x		x	x	x			•	•	•	•	•
JSR			x	x		x	x	x			•	•	•	•	•
LDA		x	x	x		x	x	x			•	•	^	^	•
LDX		x	x	x		x	x	x			•	•	^	^	•

(to be continued)

Condition Code Symbols
- H Half Carry (From Bit 3)
- I Interrupt Mask
- N Negative (Sign Bit)
- Z Zero
- C Carry Borrow
- ^ Test and Set if True, Cleared Otherwise
- • Not Affected

Table 7 Instruction Set

Mnemonic	Addressing Modes										Condition Code				
	Implied	Immediate	Direct	Extended	Relative	Indexed (No Offset)	Indexed (8 Bits)	Indexed (16 Bits)	Bit Set/ Clear	Bit Test & Branch	H	I	N	Z	C
LSL	x		x			x	x				•	•	∧	∧	∧
LSR	x		x			x	x				•	•	0	∧	∧
NEG	x		x			x	x				•	•	∧	∧	∧
NOP	x										•	•	•	•	•
ORA		x	x	x		x	x	x			•	•	∧	∧	•
ROL	x		x			x	x				•	•	∧	∧	∧
ROR	x		x			x	x				•	•	∧	∧	∧
RSP	x										•	•	•	•	•
RTI	x										?	?	?	?	?
RTS	x										•	•	•	•	•
SBC		x	x	x		x	x	x			•	•	∧	∧	∧
SEC	x										•	•	•	•	1
SEI	x										•	1	•	•	•
STA			x	x		x	x	x			•	•	∧	∧	•
STX			x	x		x	x	x			•	•	∧	∧	•
SUB		x	x	x		x	x	x			•	•	∧	∧	∧
SWI	x										•	1	•	•	•
TAX	x										•	•	•	•	•
TST	x		x			x	x				•	•	∧	∧	•
TXA	x										•	•	•	•	•

Condition Code Symbols:
- H Half Carry (From Bit 3)
- I Interrupt Mask
- N Negative (Sign Bit)
- Z Zero
- C Carry/Borrow
- ∧ Test and Set if True, Cleared Otherwise
- • Not Affected
- ? Load CC Register From Stack

Table 8 Opcode Map

	Bit Manipulation		Branch	Read/Modify/Write					Control		Register/Memory						
	Test & Branch	Set/ Clear	Rel	DIR	A	X	,X1	,X0	IMP	IMP	IMM	DIR	EXT	,X2	,X1	,X0	
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	← HIGH
0	BRSET0	BSET0	BRA	NEG					RTI*	–	SUB						0
1	BRCLR0	BCLR0	BRN	–					RTS*	–	CMP						1
2	BRSET1	BSET1	BHI	–					–	–	SBC						2
3	BRCLR1	BCLR1	BLS	COM					SWI*	–	CPX						3 L
4	BRSET2	BSET2	BCC	LSR					–	–	AND						4 O
5	BRCLR2	BCLR2	BCS	–					–	–	BIT						5 W
6	BRSET3	BSET3	BNE	ROR					–	–	LDA						6
7	BRCLR3	BCLR3	BEQ	ASR					–	TAX	–	STA(+1)					7
8	BRSET4	BSET4	BHCC	LSL/ASL					–	CLC	EOR						8
9	BRCLR4	BCLR4	BHCS	ROL					–	SEC	ADC						9
A	BRSET5	BSET5	BPL	DEC					–	CLI	ORA						A
B	BRCLR5	BCLR5	BMI	–					–	SEI	ADD						B
C	BRSET6	BSET6	BMC	INC					–	RSP	–	JMP(-1)					C
D	BRCLR6	BCLR6	BMS	TST					–	NOP	BSR*	JSR(+3)					D
E	BRSET7	BSET7	BIL	–					–	–	LDX						E
F	BRCLR7	BCLR7	BIH	CLR					–	TXA	–	STX(+1)					F
	3/10	2/7	2/4	2/6	1/4	1/4	2/7	1/6	1/*	1/2	2/2	2/4	3/5	3/6	2/5	1/4	

(NOTE) 1. Undefined opcodes are marked with "–".
2. The number at the bottom of each column denote the number of bytes and the number of cycles required (Bytes/Cycles).
Mnemonics followed by a "*" require a different number of cycles as follows:

RTI	9
RTS	6
SWI	11
BSR	8

3. ( ) indicate that the number in parenthesis must be added to the cycle count for that instruction.