## Description

The µPD80C35, µPD80C48, and µPD48 are true stand-alone 8-bit microcomputers fabricated using CMOS technology. All of the functional blocks necessary for an integrated microcomputer are incorporated, including a 1K-byte ROM (µPD80C48 only), a 64-byte RAM, 27 I/O lines, an 8-bit timer/event counter, and a clock generator. This integrated capability permits use in stand-alone applications. For designs requiring extra capability, the µPD80C35/µPD80C48 can be expanded using peripherals and is memory compatible with industry-standard 8080A/8085A processors.

Providing compatibility with industry-standard 8048, 8748, and 8035 processors, the µPD80C35/µPD80C48 features significant savings in power consumption. In addition to the power savings gained through CMOS technology, the µPD80C35/µPD80C48 offers two standby modes (Halt and Stop modes) to further minimize power drain.

## Features

- [ ] 8-Bit CPU with memory and I/O on a single-chip
- [ ] Hardware/software-compatible with industry-standard 8048, 8748, and 8035 processors
- [ ] 1K x 8 ROM (µPD80C48 only)
- [ ] 64 x 8 RAM
- [ ] 27 I/O lines
- [ ] 2.5-µs cycle time (6-MHz crystal)
- [ ] All instructions executable in 1 or 2 cycles
- [ ] 97 instructions: 70 percent are single-byte instructions
- [ ] Internal timer/event counter
- [ ] Two interrupts (external and timer)
- [ ] Easily expandable memory and I/O
- [ ] Bus compatible with 8080A/8085A peripherals
- [ ] Power-efficient CMOS technology requiring a single +2.5 to +6.0 V power supply
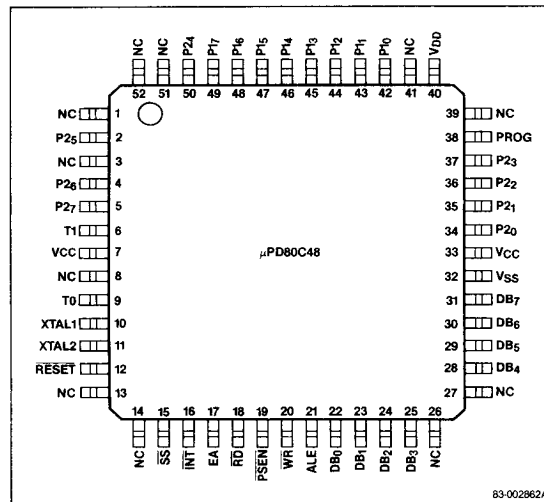- [ ] Halt mode
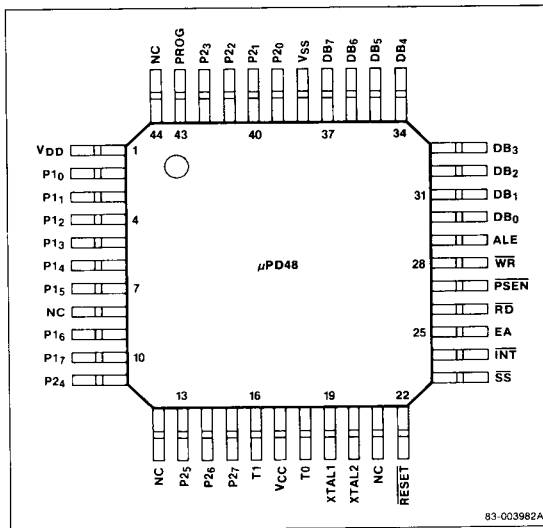- [ ] Stop mode

## Pin Configurations

### 40-Pin Plastic DIP
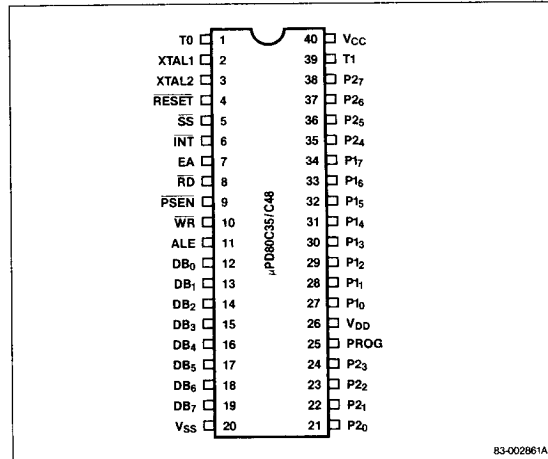


83-002861A

### 52-Pin Plastic Miniflat



83-002862A

**4**

## Pin Configurations (cont)

### 44-Pin Plastic Miniflat



83-003982A

## Ordering Information

| Part Number | Package Type | Max Frequency of Operation | ROM |
|---|---|---|---|
| μPD80C35C | 40-pin plastic DIP | 6 MHz | None |
| μPD80C48C | 40-pin plastic DIP | 6 MHz | 1K x 8 |
| μPD80C48G-00 | 52-pin plastic miniflat | 6 MHz | 1K x 8 |
| μPD48G-22 | 44-pin plastic miniflat | 6 MHz | 1K x 8 |

**Note:**

μPD80C48C, μPD80C48G-00, and μPD48G-22 have two optional port types: type 0, $I_{OH} = -5 \mu A$; type 1, $I_{OH} = -50 \mu A$. Type 0 or 1 can be selected independently for $P1_0-P1_7$, $P2_0-P2_3$, and $P2_4-P2_7$.

## Pin Identification

| Symbol | Function |
|---|---|
| T0 | Test 0 input/clock output |
| XTAL1 | Crystal 1 input |
| XTAL2 | Crystal 2 input |
| $\overline{RESET}$ | Reset input |
| $\overline{SS}$ | Software stop input |
| $\overline{INT}$ | Interrupt input |
| EA | External access input |
| $\overline{RD}$ | Read output |
| $\overline{PSEN}$ | Program store enable output |
| $\overline{WR}$ | Write output |
| ALE | Address latch enable output |
| $DB_0-DB_7$ | Bidirectional data bus |
| $V_{SS}$ | Ground |
| $P2_0-P2_7$ | Quasi-bidirectional port 2 |
| PROG | Program output |
| $V_{DD}$ | Oscillator control voltage |
| $P1_0-P1_7$ | Quasi-bidirectional port 1 |
| T1 | Test 1 input |
| $V_{CC}$ | Primary power supply |
| NC | No connection |

## Pin Functions

### XTAL1, XTAL2 [Crystals 1, 2]

XTAL1 and XTAL2 are the crystal inputs for the internal clock oscillator. XTAL1 is also used as an input for external clock signals.

### T0 [Test 0]

The JT0 and JNT0 instructions test the level of T0 and, if it is high, the program address jumps to the specified address. T0 becomes a clock output when the ENT0 CLK instruction is executed.

### T1 [Test 1]

The JT1 and JNT1 instructions test the level of T1 and, if it is high, the program address jumps to the specified address. T1 becomes an internal counter input when the STRT CNT instruction is executed.

### RESET [Reset]

$\overline{RESET}$ initializes the processor and is also used to verify the internal ROM. $\overline{RESET}$ determines the oscillation stabilizing time during the release of STOP mode. The $\overline{RESET}$ pulse width requires at least 5 machine cycles when the supply voltage is within specifications and the oscillation frequency is stable.

### $\overline{SS}$ [Single Step]

$\overline{SS}$ causes the processor to execute the program one step at a time.

### $\overline{INT}$ [Interrupt]

$\overline{INT}$ starts an interrupt if interrupts are enabled. A reset disables an interrupt. INT can be tested with the JNI instruction and, depending on the results, a jump to the specified address can occur.

### EA [External Access]

EA disables internal program memory and fetches and accesses external program memory. EA is used for system testing and debugging.

### $\overline{RD}$ [Read]

$\overline{RD}$ enables a data read from external memory.

### $\overline{WR}$ [Write]

$\overline{WR}$ enables a data write to external memory.

### $\overline{PSEN}$ [Program Store Enable]

$\overline{PSEN}$ fetches instructions only from external program memory.

### ALE [Address Latch Enable]

ALE occurs at each cycle. The falling edge of ALE addresses external data memory or external program memory. ALE can also be used as a clock output.

### $DB_0$-$DB_7$ [Data Bus]

$DB_0$-$DB_7$ is a bidirectional port, which reads and writes data using $\overline{RD}$ and $\overline{WR}$ for latching. During an external program memory fetch, $DB_0$-$DB_7$ output the low-order eight bits of the memory address. $\overline{PSEN}$ fetches the instruction. $DB_0$-$DB_7$ also output the address of an external data memory fetch. The addressed data is read and written by $\overline{RD}$ and $\overline{WR}$.

### $P1_0$-$P1_7$ [Port 1]

$P1_0$-$P1_7$ is an 8-bit quasi-bidirectional port.

### $P2_0$-$P2_7$ [Port 2]

$P2_0$-$P2_7$ is an 8-bit quasi-bidirectional port. $P2_0$-$P2_3$ output the high-order four bits of the address during an external program memory fetch. $P2_0$-$P2_3$ also function as a 4-bit I/O bus for the μPD82C43 I/O port expander.

### PROG [Program Pulse]

PROG is used as an output pulse during a fetch when interfacing with the μPD82C43 I/O port expander.

### $V_{DD}$ [Oscillator Control Voltage]

$V_{DD}$ stops and starts the oscillator in STOP mode. STOP mode is enabled by forcing $V_{DD}$ low during a rest.

### $V_{CC}$ [Primary Power Supply]

$V_{CC}$ is the primary power supply. $V_{CC}$ must be between +2.5 V and +6.0 V for normal operation. In STOP mode, $V_{CC}$ must be at least +2.0 V to ensure data retention.
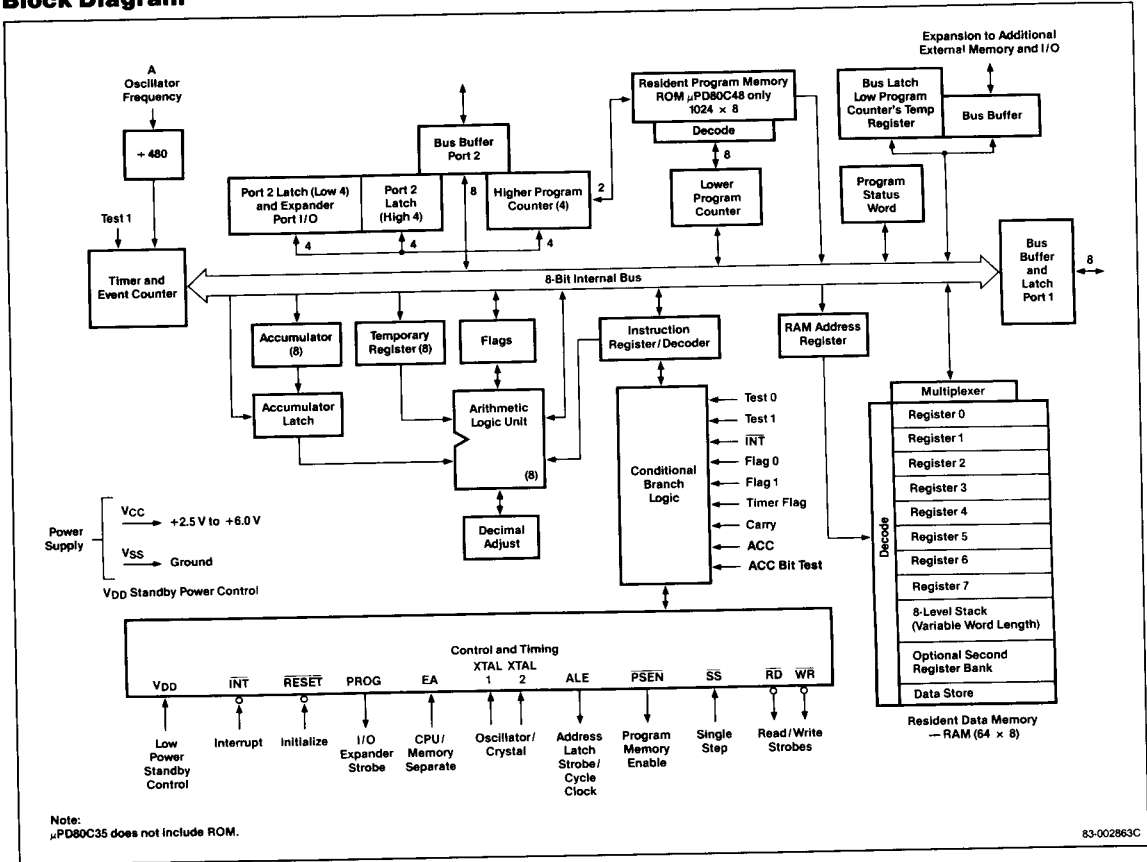
### $V_{SS}$ [Ground]

$V_{SS}$ is ground potential.

### NC [No Connection]

NC is no connection.

**4**

## Block Diagram



Note:
μPD80C35 does not include ROM.

83-002863C

## Absolute Maximum Ratings

$T_A = 25°C$

| | |
|---|---|
| Power supply voltage, $V_{CC}$ | $V_{SS} - 0.3$ to $+10$ V |
| Input voltage, $V_{IN}$ | $V_{SS} - 0.3$ to $V_{CC} + 0.3$ V |
| Output voltage, $V_O$ | $V_{SS} - 0.3$ to $V_{CC} + 0.3$ V |
| Operating temperature, $T_{OPT}$ | $-40°C$ to $+85°C$ |
| Storage temperature, $T_{STG}$ | $-65°C$ to $+150°C$ |

**Comment:** Exposing the device to stresses above those listed in Absolute Maximum Ratings could cause permanent damage. The device is not meant to be operated under conditions outside the limits described in the operational sections of the specification. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

## DC Characteristics

### Standard Voltage Range

$T_A = -40°C$ to $+85°C$, $V_{CC} = +5 V \pm 10\%$, $V_{SS} = 0 V$

| Parameter | Symbol | Min | Typ | Max | Unit | Test Conditions |
|-----------|--------|-----|-----|-----|------|-----------------|
| Input voltage low | $V_{IL}$ | $-0.3$ | | $+0.8$ | V | |
| Input voltage high | $V_{IH}$ | $V_{CC}-2$ | | $V_{CC}$ | V | Except XTAL1, XTAL2, $\overline{RESET}$ |
| | $V_{IH1}$ | $V_{CC}-1$ | | $V_{CC}$ | V | $\overline{RESET}$, XTAL1, XTAL2 |
| Output voltage low | $V_{OL}$ | | | $+0.45$ | V | $I_{OL} = 2.0 \, mA$ |
| Output voltage high | $V_{OH}$ | 2.4 | | | V | Bus, $\overline{RD}$, $\overline{WR}$, $\overline{PSEN}$, ALE, PROG, T0; $I_{OH} = -100 \, \mu A$ |
| | $V_{OH1}(1)$ | 2.4 | | | V | $I_{OH} = -5 \, \mu A$ (type 0) port 1, port 2 |
| | | 2.4 | | | V | $I_{OH} = -50 \, \mu A$ (type 1) port 1, port 2 |
| | $V_{OH2}$ | $V_{CC}-0.5$ | | | V | All outputs, $I_{OH} = -0.2 \, \mu A$ |
| Input current | $I_{ILP}(1)$ | | $-15$ | $-40$ | $\mu A$ | Port 1, port 2; $V_{IN} \leqslant V_{IL}$ (type 0) |
| | | | | $-500$ | $\mu A$ | Port 1, port 2; $V_{IN} \leqslant V_{IL}$ (type 1) |
| | $I_{ILC}$ | | | $-40$ | $\mu A$ | $\overline{SS}$, $\overline{RESET}$; $V_{IN} \leqslant V_{IL}$ |
| Input leakage current | $I_{LI1}$ | | | $\pm 1$ | $\mu A$ | T1, $\overline{INT}$, $V_{CC}$; $V_{SS} \leqslant V_{IN} \leqslant V_{CC}$ |
| | $I_{LI2}$ | | | $\pm 3$ | $\mu A$ | EA; $V_{SS} \leqslant V_{IN} \leqslant V_{CC}$ |
| Output leakage current | $I_{LO}$ | | | $\pm 1$ | $\mu A$ | $V_{SS} \leqslant V_0 \leqslant V_{CC}$ High impedance, bus, T0 |
| Standby current | $I_{CC1}$ | | 0.4 | 0.8 | mA | Halt mode $t_{CY} = 2.5 \, \mu s$ |
| | $I_{CC2}$ | | 1 | 20 | $\mu A$ | Stop mode (Note 2) |
| Supply current | $I_{CC}$ | | 4 | 8 | mA | $t_{CY} = 2.5 \, \mu s$ |
| Data retention voltage | $V_{CCDR}$ | 2.0 | | | V | Stop mode ($V_{DD}$, $\overline{RESET} \leqslant 0.4 V$) |

### Extended Voltage Range

$T_A = -40°C$ to $+85°C$, $V_{CC} = +2.5 V$ to $+6.0 V$, $V_{SS} = 0 V$

| Parameter | Symbol | Min | Typ | Max | Unit | Test Conditions |
|-----------|--------|-----|-----|-----|------|-----------------|
| Input voltage low | $V_{IL}$ | $-0.3$ | | $+0.18 V_{CC}$ | V | |
| Input voltage high | $V_{IH}$ | $0.7 V_{CC}$ | | $V_{CC}$ | V | Except XTAL1, XTAL2 |
| | $V_{IH1}$ | $0.8 V_{CC}$ | | $V_{CC}$ | V | XTAL1, XTAL2 |
| Output voltage low | $V_{OL}$ | | | $+0.45$ | V | $I_{OL} = 1.0 \, mA$ |
| Output voltage high | $V_{OH}$ | $0.75 V_{CC}$ | | | V | Bus, $\overline{RD}$, $\overline{WR}$, $\overline{PSEN}$, ALE, PROG, T0; $I_{OH} = -100 \, \mu A$ |
| | $V_{OH1}$ | $0.7 V_{CC}$ | | | V | All other outputs; $I_{OH} = -1 \, \mu A$ (type 0) port 1, port 2 |
| | | $0.7 V_{CC}$ | | | V | All other outputs; $I_{OH} = -10 \, \mu A$ (type 1) port 1, port 2 |
| Input current | $I_{ILP}$ | | $-15$ | $-40$ | $\mu A$ | Port 1, port 2; $V_{IN} \leqslant V_{IL}$ (type 0) |
| | | | | $-500$ | $\mu A$ | Port 1, port 2; $V_{IN} \leqslant V_{IL}$ (type 1) |
| Input leakage current | $I_{IL}$ | | | $-40$ | $\mu A$ | $\overline{SS}$, $\overline{RESET}$; $V_{IN} \leqslant V_{IL}$ |
| | $I_{LI1}$ | | | $\pm 1$ | $\mu A$ | T1, $\overline{INT}$, $V_{SS}$ $<V_{IN}<V_{CC}$ |
| | $I_{LI2}$ | | | $\pm 5$ | $\mu A$ | EA; $V_{SS}$ $<V_{IN}<V_{CC}$ |
| Output leakage current | $I_{LO}$ | | | $\pm 1$ | $\mu A$ | $V_{SS}<V_0<V_{CC}$, Bus, T0 — high impedance state |
| Supply current | $I_{CC}$ | | 0.8 | 1.6 | mA | $V_{CC} = 3 V$, $t_{CY} = 10 \, \mu s$ |
| | | | 6 | 12 | $\mu A$ | $V_{CC} = 6 V$, $t_{CY} = 25 \, \mu s$ |
| Standby current | $I_{CC1}$ | | 100 | 200 | $\mu A$ | Halt mode; $V_{CC} = 3 V$, $t_{CY} = 10 \, \mu s$ |
| | | | 0.6 | 1.2 | mA | $V_{CC} = 6 V$, $t_{CY} = 2.5 \, \mu s$ |
| | $I_{CC2}$ | | 1 | 20 | $\mu A$ | Stop mode, $V_{CC} = 3 V$ |
| | | | 1 | 50 | $\mu A$ | $V_{CC} = 6 V$ |

**Note:**

(1) Types 0, 1 for $\mu$PD80C48 only.
Type 0 for $\mu$PD80C35 only.

(2) Input pin voltage is $V_{IN} \leqslant V_{IL}$, or $V_{IN} \geqslant V_{IH}$.

**4**

## AC Characteristics

### Read, Write and Instruction Fetch: External Data and Program Memory

$T_A = -40°C$ to $+85°C$, $V_{CC} = V_{DD} = +5V \pm 10\%$; $V_{SS} = 0V$

| Parameter | Symbol | $V_{CC} = +5V \pm 10\%$ Min | Max | $V_{CC} = 2.5V$ to $6.0V$ Min | Max | Unit | Test Conditions |
|---|---|---|---|---|---|---|---|
| ALE pulse width | $t_{LL}$ | 400 | | 2160 | | ns | |
| Address setup before ALE | $t_{AL}$ | 120 | | 1620 | | ns | |
| Address hold from ALE | $t_{LA}$ | 80 | | 330 | | ns | (Note 1) |
| Control pulse width ($\overline{RD}$, $\overline{WR}$, $\overline{PSEN}$) | $t_{CC}$ | 700 | | 3700 | | ns | |
| Data setup before $\overline{WR}$ | $t_{DW}$ | 500 | | 3500 | | ns | |
| Data hold after $\overline{WR}$ | $t_{WD}$ | 120 | | 370 | | ns | (Note 2) |
| Cycle time | $t_{CY}$ | 2.5 | 150 | 10 | 150 | µs | 6 MHz XTAL |
| Data hold | $t_{DR}$ | 0 | 200 | 0 | 950 | ns | |
| $\overline{PSEN}$, $\overline{RD}$ to data in | $t_{RD}$ | | 500 | | 2750 | ns | |
| Address setup before $\overline{WR}$ | $t_{AW}$ | 230 | | 3230 | | ns | (Note 1) |
| Address setup before data in | $t_{AD}$ | | 950 | | 5450 | | |
| Address float to $\overline{RD}$, $\overline{PSEN}$ | $t_{AFC}$ | 0 | | 500 | | ns | |
| Control pulse to ALE | $t_{CA}$ | 10 | | 10 | | ns | |

### Port 2 Timing

$T_A = -40°C$ to $+85°C$, $V_{CC} = V_{DD} = +5V \pm 10\%$; $V_{SS} = 0V$

| Parameter | Symbol | $V_{CC} = +5V \pm 10\%$ Min | Max | $V_{CC} = 2.5V$ to $6.0V$ Min | Max | Unit | Test Conditions |
|---|---|---|---|---|---|---|---|
| Port control setup before falling edge of PROG | $t_{CP}$ | 110 | | 860 | | ns | |
| Port control hold after falling edge of PROG | $t_{PC}$ | 0 | 80 | 0 | 200 | ns | (Note 4) |
| PROG to time P2 input must be valid | $t_{PR}$ | | 810 | | 5310 | ns | |
| Output data setup time | $t_{DP}$ | 250 | | 3250 | | ns | (Note 3) |
| Output data hold time | $t_{PD}$ | 65 | | 820 | | ns | |
| Input data hold time | $t_{PF}$ | 0 | 150 | 0 | 900 | ns | |
| PROG pulse width | $t_{PP}$ | 1200 | | 6450 | | ns | |
| Port 2 I/O data setup time | $t_{PL}$ | 350 | | 2100 | | ns | |
| Port 2 I/O data hold time | $t_{LP}$ | 150 | | 1400 | | ns | |

**Note:**
(1) Control outputs: $C_L = 80 pF$, bus outputs: $C_L = 150 pF$
(2) $C_L = 20 pF$
(3) Control outputs: $C_L = 80 pF$
(4) Refer to the operating characteristics curves for supply voltage and port control hold.

### Bus Timing Requirements (Note 1)

| Symbol | Timing Formula | Min/Max | Unit |
|---|---|---|---|
| $t_{LL}$ | $(7/30) t_{CY} - 170$ | Min | ns |
| $t_{AL}$ | $(1/5) t_{CY} - 380$ | Min | ns |
| $t_{LA}$ | $(1/30) t_{CY}$ | Min | ns |
| $t_{CC}$ | $(2/5) t_{CY} - 300$ | Min | ns |
| $t_{DW}$ | $(2/5) t_{CY} - 500$ | Min | ns |
| $t_{WD}$ | $(1/30) t_{CY} + 40$ | Min | ns |
| $t_{DR}$ | $(1/10) t_{CY} - 50$ | Max | ns |
| $t_{RD}$ | $(3/10) t_{CY} - 250$ | Max | ns |
| $t_{AW}$ | $(2/5) t_{CY} - 770$ | Min | ns |
| $t_{AD}$ | $(3/5) t_{CY} - 550$ | Max | ns |
| $t_{AFC}$ | $(1/15) t_{CY} - 165$ | Min | ns |

| Symbol | Timing Formula | Min/Max | Unit |
|---|---|---|---|
| $t_{CP}$ | $(1/10) t_{CY} - 140$ | Min | ns |
| $t_{PC2}$ | $(4/15) t_{CY} - 200$ | Min | ns |
| $t_{PR}$ | $(3/5) t_{CY} - 690$ | Max | ns |
| $t_{PF}$ | $(1/10) t_{CY} - 100$ | Max | ns |
| $t_{DP}$ | $(2/5) t_{CY} - 750$ | Min | ns |
| $t_{PD}$ | $(1/10) t_{CY} - 180$ | Min | ns |
| $t_{PP}$ | $(7/10) t_{CY} - 550$ | Min | ns |
| $t_{PL}$ | $(7/30) t_{CY} - 230$ | Min | ns |
| $t_{LP}$ | $(1/6) t_{CY} - 265$ | Min | ns |

**Note:**
(1) Unlisted parameters are not affected by cycle time.

## Timing Waveforms

### Instruction Fetch From External Memory



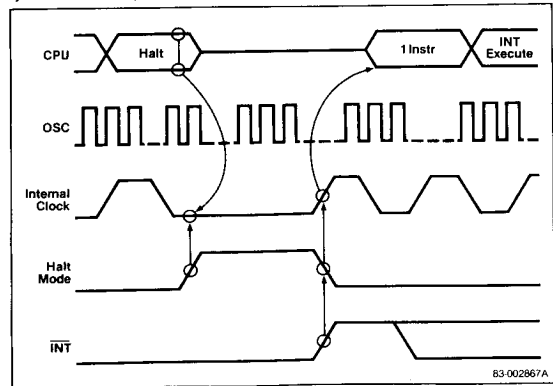83-002864A
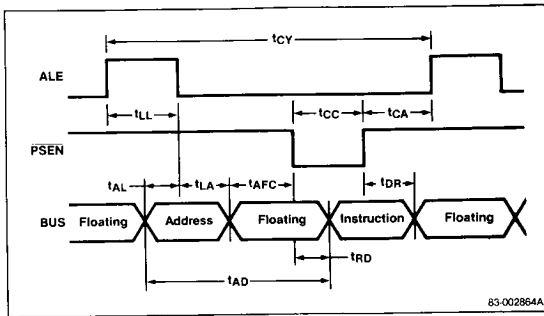
### Read From External Data Memory



83-002865A

### Write to External Memory



83-002866A
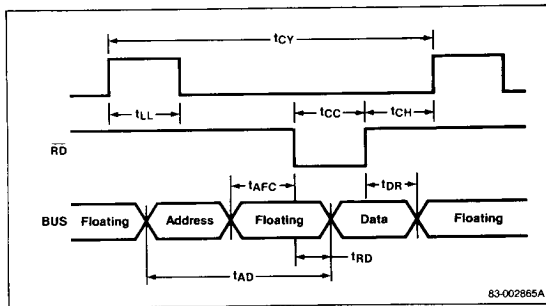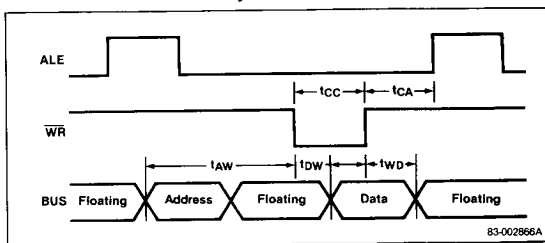
### Low Power Standby Operation
### 1) Halt Mode (When EI)



83-002867A

### 2) Stop Mode



83-002868A

### Port 2 Timing



83-002869A
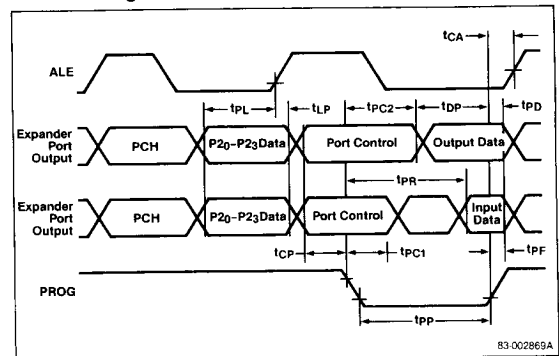
4-219
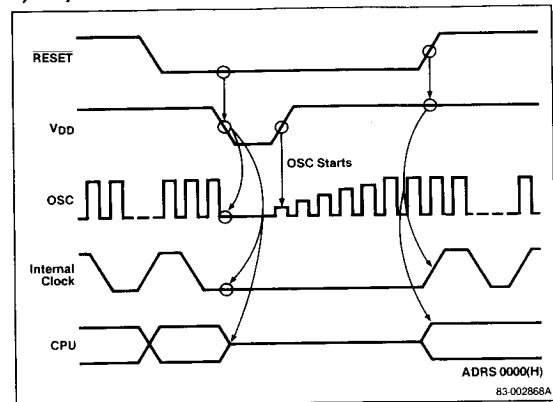
## Functional Description

### Standby Function

#### Halt Mode

In Halt mode, the oscillator continues to operate, but the internal clock is disabled. The status of all internal logic just prior to execution of the HALT instruction is maintained by the CPU. In Halt mode, power consumption is less than 10 percent of normal μPD80C48 operation and less than 1 percent of normal 8048 operation.

The Halt mode is initiated by execution of the HALT instruction, and is released by either INT or RESET input.

INT Input. When the INT pin receives a low-level input, if interrupts are enabled, the internal clock is restarted and the interrupt is executed after the first or second instruction following the HALT instruction. However, if interrupts are disabled, program operation is resumed from the next address following the HALT instruction. The first instruction following a HALT instruction should be a NOP instruction to ensure proper program execution.

If the Halt mode is released when interrupts are enabled, the interrupt service routine is usually executed after the first or second instruction following the release of Halt mode. However, if either a timer or external interrupt is accepted within one machine cycle prior to a HALT instruction, the corresponding timer or external interrupt service routine is executed immediately following the release of Halt mode. It is important to note this sequence of execution when considering interrupt service routine execution following a HALT instruction.
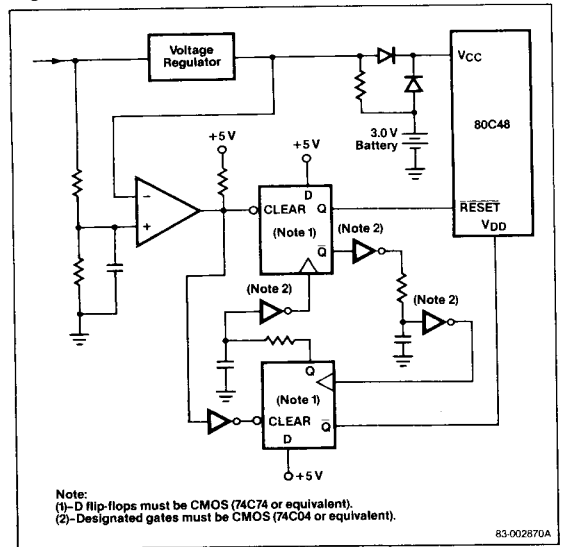
RESET Input. When a low-level input is received by the RESET pin, Halt mode is released and the normal reset operation is activated, restarting program operation from address 0.

#### Stop Mode

In Stop mode, the oscillator is deactivated and only the contents of RAM are maintained. The operation status of the μPD80C35/μPD80C48 resembles that of a reset condition. Because only the contents of RAM are maintained, Stop mode provides even lower power consumption than Halt mode, only requiring a minimum $V_{CC}$ as low as +2 V.

Stop mode is initiated by setting $V_{DD}$ to low when RESET is low, to protect the contents of RAM. Stop mode is released by first raising the supply voltage at the $V_{CC}$ pin from standby level to correct operating level and setting $V_{DD}$ to high when RESET is low. After the oscillator has been restarted and the oscillation has stabilized, RESET must be set to high, whereby program operation is started from address 0. Figure 1 shows the Stop mode circuit.

### Figure 1. Stop Mode Circuit



Note:
(1)–D flip-flops must be CMOS (74C74 or equivalent).
(2)–Designated gates must be CMOS (74C04 or equivalent).

83-002870A

Stop Mode Circuit. Since $V_{DD}$ controls the restarting of the oscillator, it is important that $V_{DD}$ be protected from noise interference. The time required to reset the CPU is represented by $t_1$ (see figure 2), which is a minimum of 5 machine cycles. The reset operation will not be completed in less than 5 machine cycles. In Stop mode, it is important to note that if $V_{DD}$ goes low before 5 machine cycles have elapsed, the CPU will be deactivated and the output of ALE, RD, WR, PSEN, and PROG will not have been stabilized.

### Figure 2. Stop Mode Timing



83-002871A
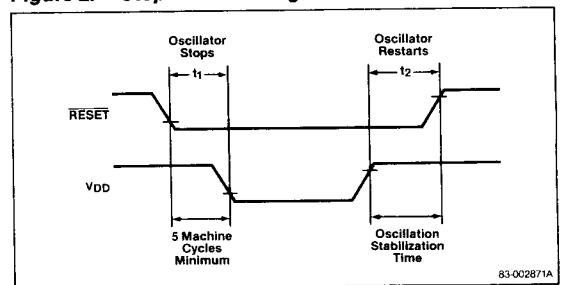
Oscillation stabilization time is represented by $t_2$ (see figure 2). When $V_{DD}$ goes high, oscillator operation is re-activated, but it takes time before oscillation can be stabilized. In particular, such high Q resonators as crystals require longer periods to stabilize. Because there is a delay between restarting of the oscillator and oscillator stabilization, $t_2$ should be long enough to ensure that the oscillator has been fully stabilized.

To facilitate Stop mode control, an external capacitor can be connected to the $\overline{RESET}$ pin (see figure 3), affecting only $t_2$, allowing control of the oscillator stabilization time. When $V_{DD}$ is asserted in Stop mode, the capacitor begins charging, pulling up $\overline{RESET}$. When $\overline{RESET}$ reaches a threshold level equivalent to a logic 1, Stop mode is released. The time it takes $\overline{RESET}$ to reach the threshold level of logic 1 determines the oscillator stabilization time, which is a function of the capacitance and pull-up resistance values.

**Figure 3. Stop Mode Control Circuit**



Note: (1)–Polarized electrolytic capacitor.

83-002872A
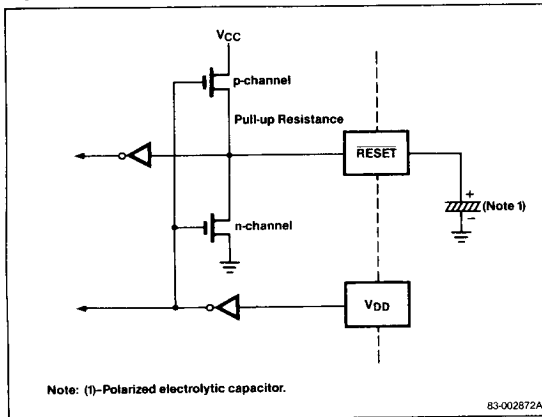
**Port Operation**

A port-loading option is offered at the time of ordering the mask. Individual source current requirements for Port 1 and the upper and lower halves of Port 2 may be factory set at either $-5\,\mu A$ or $-50\,\mu A$ (see Port-Loading Options table). The $-50\,\mu A$ option is required for interfacing with TTL/NMOS devices. The $-5\,\mu A$ option is recommended for interfacing to other CMOS devices. The CMOS option results in lower power consumption and greater noise immunity.

Port lines $P1_0$-$P1_7$ and $P2_4$-$P2_7$ include a protective circuit "E" to prevent a signal conflict at the port. The circuit prevents a logic 1 from being written to a line that is being pulled down externally (see figure 4, Port Protection Circuit E diagram). When a logic 0 is detected at the port line and a logic 1 is written from the bus, the NOR gate sends a logic 1 to the D input of the flip-flop. The output is inverted, forcing the NAND gate to send a high-level output. This turns off transistor A, preventing the output of a logic 1 from the port.

**Figure 4. Port Protection Circuit E**



83-002873A
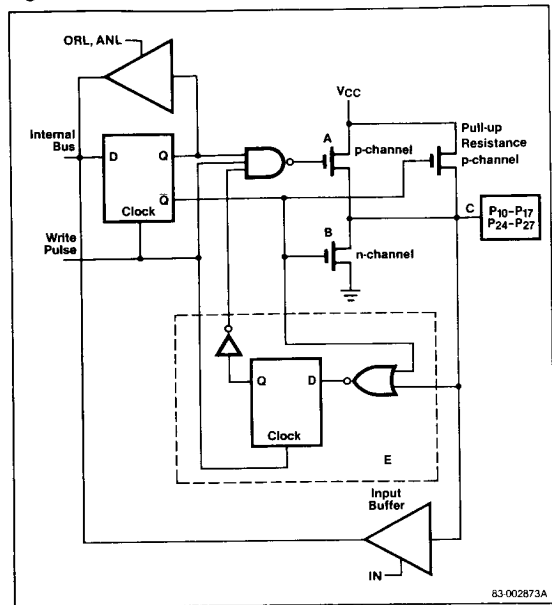
## Port-Loading Options

$I_{OH}$ (min) $V_{CC} = V_{DD} = 5 V \pm 10\%$; $V_{OH} = 2.4 V$ (min)

| Option Selected | $P1_0$-$P1_7$ | $P2_0$-$P2_3$ | $P2_4$-$P2_7$ | Unit |
|---|---|---|---|---|
| A | −5 | −5 | −5 | μA |
| B | −50 | −5 | −5 | μA |
| C | −5 | −50 | −5 | μA |
| D | −50 | −50 | −5 | μA |
| E | −5 | −5 | −50 | μA |
| F | −50 | −5 | −50 | μA |
| G | −5 | −50 | −50 | μA |
| H | −50 | −50 | −50 | μA |

**Note:**

(1) The selection of $I_{OH} = -5\,\mu A$ will result in a port source current of $I_{ILP} = -40\,\mu A$ max when used as input port.

(2) The selection of $I_{OH} = -50\,\mu A$ will result in a port source current of $I_{ILP} = -500\,\mu A$ max when used as input port.

## Oscillator Operation

The oscillator maintains an internal frequency for clock generation and controls all system timing cycles. The oscillation is initiated by either a self-generating external resonator or external clock input. The oscillator acts as a high-gain amplifier which produces square-wave pulses at the frequency determined by the resonator or clock source to which it is connected.

To obtain the oscillation frequency, an external LC network (figure 5) may be connected to the oscillator, or, a ceramic or crystal external resonator (figure 6) may be connected.

**Figure 5. LC Frequency Reference Circuit**

| L | C | Nominal f |
|---|---|---|
| 45 μH | 20 pF | 5.2 MHz |
| 120 μH | 20 pF | 3.2 MHz |

$$f \approx \frac{1}{2\pi\sqrt{LC'}} \qquad C' = \frac{C + 3\,C_{PP}}{2}$$

**Note:**
$C_{PP} = 5 - 10$ pF. Pin to pin capacitance should be approximately 20pF, including stray capacitance.

83-002875A
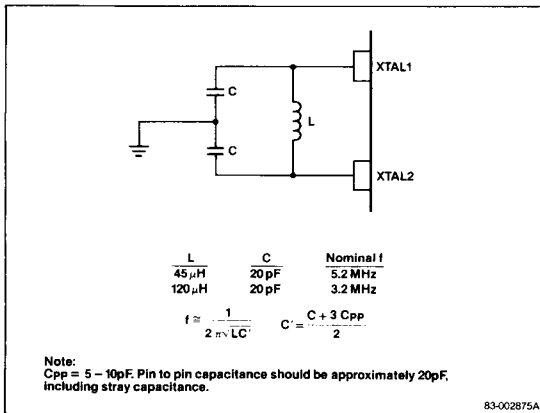
**Figure 6. Crystal Frequency Reference Circuit**

**Notes:**
(1) Crystal oscillator constants of
$f_{OSC} = 6$ MHz
$R_{max} = 50\,\Omega$
$C_L = 16 \pm 0.2$ pF
$P = 1 \pm 0.2$ mW
(2) Operating frequency less than 4 MHz
$0 < C_1 \leqslant 20$ pF
$0 < C_2 \leqslant 20$ pF
$|C_2 - C_1| \leqslant 10$ pF
(3) Operating frequency more than 4 MHz
$0 < C_1 \leqslant 10$ pF
$0 < C_2 \leqslant 10$ pF
$|C_2 - C_1| \leqslant 5$ pF

83-002874A
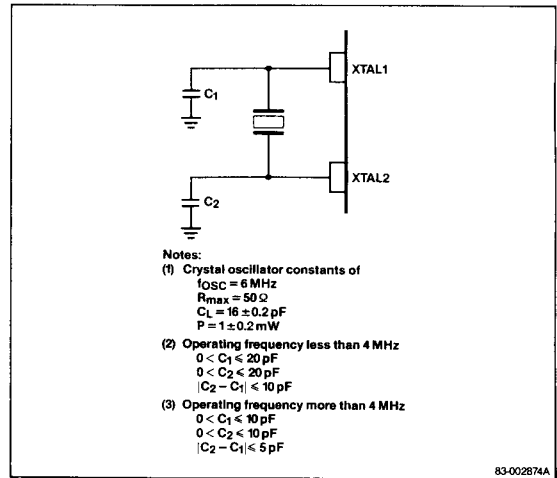
As the crystal frequency is lowered, there is an equivalent reduction in series resistance (R). As the temperature of the crystal is lowered, R is increased. Due to this relationship, it becomes difficult to stabilize oscillation when there is low power supply voltage. When $V_{CC}$ is less than 2.7 V and the oscillator frequency is 3 MHz or less, $T_A$ (ambient temperature) should not be less than $-10°C$.

Figures 7 and 8 show the ceramic resonator and external clock frequency reference circuits. Figure 9 shows the μPD80C35/μPD80C48 major I/O signals.

**Figure 7. Ceramic Resonator Frequency Reference Circuit**

**Note:**
$C_1 > C_2$
$|C_1 - C_2| \cong 20$ pF
For example, $C_1 = 30$ pF, and $C_2 = 10$ pF.
Values of $C_1$ and $C_2$ do not include stray capacitance.

83-002876A
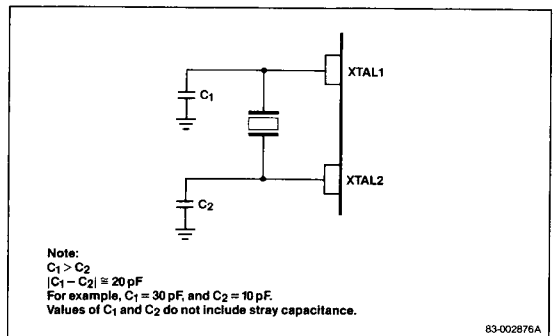
**Figure 8.   External Clock Frequency Reference Circuit**



Note:
A minimum voltage of $V_{CC} - 1$ is required for XTAL1 to go HIGH.

83-002877A
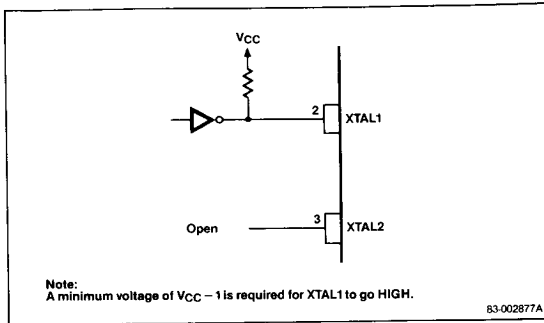
**Figure 9.   Major Input and Output Signals**



83-002878A
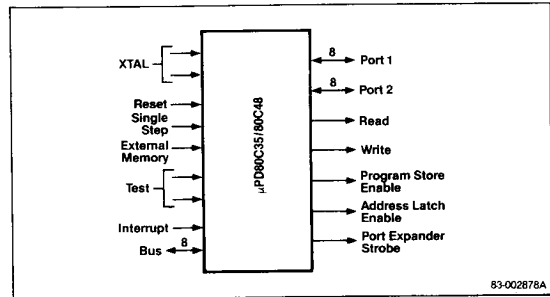
## Instruction Set

### Instruction Set Symbol Definitions

| Symbol | Description |
|--------|-------------|
| A | Accumulator |
| AC | Auxiliary carry flag |
| addr | Program or data memory address ($a_0$–$a_7$) or ($a_0$–$a_{10}$) |
| b | Accumulator bit (b = 0–7) |
| BS | Bank switch |
| BUS | Bus |
| C | Carry flag |
| CLK | Clock |
| CNT | Counter |
| data | 8-bit binary data ($d_0$–$d_7$) |
| DBF | Memory bank flip-flop |
| F0, F1 | Flag 0, flag 1 |
| INT | Interrupt pin |
| n | Indicates the hex number of the specified register or port |
| PC | Program counter |
| Pp | Port 1, port 2, or ports 4–7 (p = 1, 2 or 4–7) |
| PSW | Program status word |
| Rr | Register (r = 0–7) |

| Symbol | Description |
|--------|-------------|
| SP | Stack pointer |
| T | Timer |
| TF | Timer flag |
| T0, T1 | Test 0, test 1 pin |
| # | Prefix for immediate data |
| @ | Prefix for indirect address |
| x | Indicates the hex number corresponding to the accumulator bit or page number specified in the operand |
| (x) | Contents of RAM |
| ((x)) | Contents of memory location addressed by (x) |
| ← | Transfer direction, result |
| AND | Logical product (logical AND) |
| OR | Logical sum (logical OR) |
| EXOR | Exclusive-OR |
| ‾‾‾ | Complement |

**4**

# Instruction Set (cont)

| Mnemonic | Function | Description | Hex Code | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | Cycles | Bytes |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Accumulator** | | | | | | | | | | | | | |
| ADD A, # data | $(A) \leftarrow (A) + data$ | Adds immediate data $d_0$–$d_7$ to the accumulator. Sets or clears both carry flags. (Note 2) | 03 | 0 / $d_7$ | 0 / $d_6$ | 0 / $d_5$ | 0 / $d_4$ | 0 / $d_3$ | 0 / $d_2$ | 1 / $d_1$ | 1 / $d_0$ | 2 | 2 |
| ADD A, Rr | $(A) \leftarrow (A) + (Rr)$ $r = 0$–$7$ | Adds the contents of register Rr to the accumulator. Sets or clears both carry flags. (Note 2) | 6n(4) | 0 | 1 | 1 | 0 | 1 | r | r | r | 1 | 1 |
| ADD A, @ Rr | $(A) \leftarrow (A) + ((Rr))$ $r = 0$–$1$ | Adds the contents of the internal data memory location specified by bits 0–5 of register Rr to the accumulator. Sets or clears both carry flags. (Note 2) | 6n(4) | 0 | 1 | 1 | 0 | 0 | 0 | 0 | r | 1 | 1 |
| ADDC A, # data | $(A) \leftarrow (A) + (C) + data$ | Adds, with carry, immediate data $d_0$–$d_7$ to the accumulator. Sets or clears both carry flags. (Note 2) | 13 | 0 / $d_7$ | 0 / $d_6$ | 0 / $d_5$ | 1 / $d_4$ | 0 / $d_3$ | 0 / $d_2$ | 1 / $d_1$ | 1 / $d_0$ | 2 | 2 |
| ADDC A, Rr | $(A) \leftarrow (A) + (C) + (Rr)$ $r = 0$–$7$ | Adds, with carry, the contents of register Rr to the accumulator. Sets or clears both carry flags. (Note 2) | 7n(4) | 0 | 1 | 1 | 1 | 1 | r | r | r | 1 | 1 |
| ADDC A, @ Rr | $(A) \leftarrow (A) + (C) + ((Rr))$ $r = 0$–$1$ | Adds, with carry, the contents of the internal data memory location specified by bits 0–5 of register Rr, to the accumulator. Sets or clears both carry flags. (Note 2) | 7n(4) | 0 | 1 | 1 | 1 | 0 | 0 | 0 | r | 1 | 1 |
| ANL A, # data | $(A) \leftarrow (A)$ AND data | Takes the logical product (logical AND) of immediate data $d_0$–$d_7$ and the contents of the accumulator, and stores the result in the accumulator. | 53 | 0 / $d_7$ | 1 / $d_6$ | 0 / $d_5$ | 1 / $d_4$ | 0 / $d_3$ | 0 / $d_2$ | 1 / $d_1$ | 1 / $d_0$ | 2 | 2 |
| ANL A, Rr | $(A) \leftarrow (A)$ AND (Rr) $r = 0$–$7$ | Takes the logical product (logical AND) of the contents of register Rr and the accumulator, and stores the result in the accumulator. | 5n(4) | 0 | 1 | 0 | 1 | 1 | r | r | r | 1 | 1 |
| ANL A, @ Rr | $(A) \leftarrow (A)$ AND ((Rr)) $r = 0$–$1$ | Takes the logical product (logical AND) of the contents of the internal data memory location specified by bits 0–5 of register Rr, and the accumulator, and stores the result in the accumulator. | 5n(4) | 0 | 1 | 0 | 1 | 0 | 0 | 0 | r | 1 | 1 |
| CPL A | $(A) \leftarrow (\bar{A})$ | Takes the complement of the contents of the accumulator. | 37 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| CLR A | $(A) \leftarrow 0$ | Clears the contents of the accumulator. | 27 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| DA A | | Converts the contents of the accumulator to BCD. Sets or clears the carry flags. When the lower 4 bits ($A_0$–$A_3$) are greater than 9, or if the auxiliary carry flag has been set, adds 6 to ($A_0$–$A_3$). When the upper 4 bits ($A_4$–$A_7$) are greater than 9 or if the carry flag (C) has been set, adds 6 to ($A_4$–$A_7$). If an overflow occurs at this point, C is set. (Note 2) | 57 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| DEC A | $(A) \leftarrow (A) - 1$ | Decrements the contents of the accumulator by 1. | 07 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| INC A | $(A) \leftarrow (A) + 1$ | Increments the contents of the accumulator by 1. | 17 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| ORL A, # data | $(A) \leftarrow (A)$ OR data | Takes the logical sum (logical OR) of immediate data $d_0$–$d_7$ and the contents of the accumulator, and stores the result in the accumulator. | 43 | 0 / $d_7$ | 1 / $d_6$ | 0 / $d_5$ | 0 / $d_4$ | 0 / $d_3$ | 0 / $d_2$ | 1 / $d_1$ | 1 / $d_0$ | 2 | 2 |
| ORL A, Rr | $(A) \leftarrow (A)$ OR (Rr) $r = 0$–$7$ | Takes the logical sum (logical OR) of register Rr and the contents of the accumulator, and stores the result in the accumulator. | 4n(4) | 0 | 1 | 0 | 0 | 1 | r | r | r | 1 | 1 |

4-224

**4**

## Instruction Set (cont)

### Accumulator (cont)

| Mnemonic | Function | Description | Hex Code | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | Cycles | Bytes |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ORL A, @ Rr | $(A) \leftarrow (A)$ OR $((Rr))$ $r = 0\text{-}1$ | Takes the logical sum (logical OR) of the contents of the internal data memory location specified by bits 0-5 in register Rr, and the contents of the accumulator, and stores the result in the accumulator. | 4n(4) | 0 | 1 | 0 | 0 | 0 | 0 | 0 | r | 1 | 1 |
| RL A | $(AN + 1) \leftarrow (AN)$ $(A_0) \leftarrow (A_7)$ $N = 0\text{-}6$ | Rotates the contents of the accumulator one bit to the left. The MSB is rotated into the LSB. | E7 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| RLC A | $(AN + 1) \leftarrow (AN)$; $N = 0\text{-}6$ $(A_0) \leftarrow (C)$ $(C) \leftarrow (A_7)$ | Rotates the contents of the accumulator one bit to the left through carry. | F7 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| RR A | $(AN) \leftarrow (AN + 1)$; $N = 0\text{-}6$ $(A_7) \leftarrow (A_0)$ | Rotates the contents of the accumulator one bit to the right. The LSB is rotated into the MSB. | 77 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| RRC A | $(AN) \leftarrow (AN + 1)$; $N = 0\text{-}6$ $(A_7) \leftarrow (C)$ $(C) \leftarrow (A_0)$ | Rotates the contents of the accumulator one bit to the right through carry. | 67 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| SWAP A | $(A_4\text{-}A_7) \leftrightarrow (A_0\text{-}A_3)$ | Exchanges the contents of the lower 4 bits of the accumulator with the upper 4 bits of the accumulator. | 47 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| XRL A, # data | $(A) \leftarrow (A)$ XOR data | Takes the exclusive OR of immediate data $d_0\text{-}d_7$ and the contents of the accumulator, and stores the result in the accumulator. | D3 | 1 d7 | 1 d6 | 0 d5 | 1 d4 | 0 d3 | 0 d2 | 1 d1 | 1 d0 | 2 | 2 |
| XRL A, Rr | $(A) \leftarrow (A)$ XOR $(Rr)$ $r = 0\text{-}7$ | Takes the exclusive OR of the contents of register Rr and the accumulator, and stores the result in the accumulator. | Dn(4) | 1 | 1 | 0 | 1 | 1 | r | r | r | 1 | 1 |
| XRL A, @ Rr | $(A) \leftarrow (A)$ XOR $((Rr))$ $r = 0\text{-}1$ | Takes the exclusive OR of the contents of the location in data memory specified by bits 0-5 in register Rr, and the accumulator, and stores the result in the accumulator. | Dn(4) | 1 | 1 | 0 | 1 | 0 | 0 | 0 | r | 1 | 1 |

### Branch

| Mnemonic | Function | Description | Hex Code | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | Cycles | Bytes |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DJNZ Rr, addr | $(Rr) \leftarrow (Rr) - 1$; $r = 0\text{-}7$ If $(Rr) \neq 0$; $(PC_0\text{-}PC_7) \leftarrow$ addr | Decrements the contents of register Rr by 1, and if the result is not equal to 0, jumps to the address indicated by $a_0\text{-}a_7$. | En | 1 a7 | 1 a6 | 1 a5 | 0 a4 | 1 a3 | r a2 | r a1 | r a0 | 2 | 2 |
| JBb addr | $(PC_0\text{-}PC_7) \leftarrow$ addr if b = 1 $(PC) = (PC) + 2$ if b = 0 | Jumps to the address specified by $a_0\text{-}a_7$ if the bit in the accumulator specified by $b_0\text{-}b_2$ is set. | x2(6) | b2 a7 | b1 a6 | b0 a5 | 1 a4 | 0 a3 | 0 a2 | 1 a1 | 0 a0 | 2 | 2 |

4-225

## Instruction Set (cont)

### Branch (cont)

| Mnemonic | Function | Description | Hex Code | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | Cycles | Bytes |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| JC addr | $(PC_0-PC_7) \leftarrow$ addr if $C = 1$ / $(PC) \leftarrow (PC) + 2$ if $C = 0$ | Jumps to the address specified by $a_0-a_7$ if the carry flag is set. | F6 | 1 / $a_7$ | 1 / $a_6$ | 1 / $a_5$ | 1 / $a_4$ | 0 / $a_3$ | 1 / $a_2$ | 1 / $a_1$ | 0 / $a_0$ | 2 | 2 |
| JF0 addr | $(PC_0-PC_7) \leftarrow$ addr if $F0 = 1$ / $(PC) \leftarrow (PC) + 2$ if $F0 = 0$ | Jumps to the address specified by $a_0-a_7$ if F0 is set. | B6 | 1 / $a_7$ | 0 / $a_6$ | 1 / $a_5$ | 1 / $a_4$ | 0 / $a_3$ | 1 / $a_2$ | 1 / $a_1$ | 0 / $a_0$ | 2 | 2 |
| JF1 addr | $(PC_0-PC_7) \leftarrow$ addr if $F1 = 1$ / $(PC) \leftarrow (PC) + 2$ if $F1 = 0$ | Jumps to the address specified by $a_0-a_7$ if F1 is set. | 76 | 0 / $a_7$ | 1 / $a_6$ | 1 / $a_5$ | 1 / $a_4$ | 0 / $a_3$ | 1 / $a_2$ | 1 / $a_1$ | 0 / $a_0$ | 2 | 2 |
| JMP addr | $(PC_8-PC_{10}) \leftarrow (addr_8-addr_{10})$ / $(PC_0-PC_7) \leftarrow (addr_0-addr_7)$ / $(PC_{11}) \leftarrow$ DBF | Jumps directly to the address specified by $a_0-a_{10}$ and the DBF. | x4(6) | $a_{10}$ / $a_7$ | $a_9$ / $a_6$ | $a_8$ / $a_5$ | 0 / $a_4$ | 0 / $a_3$ | 1 / $a_2$ | 0 / $a_1$ | 0 / $a_0$ | 2 | 2 |
| JMPP @ A | $(PC_0-PC_7) \leftarrow ((A))$ | Replaces the lower 8 bits of the program counter with the contents of program memory specified by the contents of the accumulator, producing a jump to the specified address within the current page. | B3 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 2 | 1 |
| JNC addr | $(PC_0-PC_7) \leftarrow$ addr if $C = 0$ / $(PC) \leftarrow (PC) + 2$ if $C = 1$ | Jumps to the address specified by $a_0-a_7$ if the carry flag is not set. | E6 | 1 / $a_7$ | 1 / $a_6$ | 1 / $a_5$ | 0 / $a_4$ | 0 / $a_3$ | 1 / $a_2$ | 1 / $a_1$ | 0 / $a_0$ | 2 | 2 |
| JNI addr | $(PC_0-PC_7) \leftarrow$ addr if $I = 0$ / $(PC) \leftarrow (PC) + 2$ if $I = 1$ | Jumps to the address specified by $a_0-a_7$ if the interrupt flag is not set. | 86 | 1 / $a_7$ | 0 / $a_6$ | 0 / $a_5$ | 0 / $a_4$ | 0 / $a_3$ | 1 / $a_2$ | 1 / $a_1$ | 0 / $a_0$ | 2 | 2 |
| JNT0 addr | $(PC_0-PC_7) \leftarrow$ addr if $T0 = 0$ / $(PC) \leftarrow (PC) + 2$ if $T0 = 1$ | Jumps to the address specified by $a_0-a_7$ if test 0 is low. | 26 | 0 / $a_7$ | 0 / $a_6$ | 1 / $a_5$ | 0 / $a_4$ | 0 / $a_3$ | 1 / $a_2$ | 1 / $a_1$ | 0 / $a_0$ | 2 | 2 |
| JNT1 addr | $(PC_0-PC_7) \leftarrow$ addr if $T1 = 0$ / $(PC) \leftarrow (PC) + 2$ if $T1 = 1$ | Jumps to the address specified by $a_0-a_7$ if test 1 is low. | 46 | 0 / $a_7$ | 1 / $a_6$ | 0 / $a_5$ | 0 / $a_4$ | 0 / $a_3$ | 1 / $a_2$ | 1 / $a_1$ | 0 / $a_0$ | 2 | 2 |
| JNZ addr | $(PC_0-PC_7) \leftarrow$ addr if $A \neq 0$ / $(PC) \leftarrow (PC) + 2$ if $A = 0$ | Jumps to the address specified by $a_0-a_7$ if the contents of the accumulator are not equal to 0. | 96 | 1 / $a_7$ | 0 / $a_6$ | 0 / $a_5$ | 1 / $a_4$ | 0 / $a_3$ | 1 / $a_2$ | 1 / $a_1$ | 0 / $a_0$ | 2 | 2 |
| JTF addr | $(PC_0-PC_7) \leftarrow$ addr if $TF = 1$ / $(PC) \leftarrow (PC) + 2$ if $TF = 0$ | Jumps to the address specified by $a_0-a_7$ if the timer flag is set. The timer flag is cleared after the instruction is executed. | 16 | 0 / $a_7$ | 0 / $a_6$ | 0 / $a_5$ | 1 / $a_4$ | 0 / $a_3$ | 1 / $a_2$ | 1 / $a_1$ | 0 / $a_0$ | 2 | 2 |
| JT0 addr | $(PC_0-PC_7) \leftarrow$ addr if $T0 = 1$ / $(PC) \leftarrow (PC) + 2$ if $T0 = 0$ | Jumps to the address specified by $a_0-a_7$ if test 0 is high. | 36 | 0 / $a_7$ | 0 / $a_6$ | 1 / $a_5$ | 1 / $a_4$ | 0 / $a_3$ | 1 / $a_2$ | 1 / $a_1$ | 0 / $a_0$ | 2 | 2 |
| JT1 addr | $(PC_0-PC_7) \leftarrow$ addr if $T1 = 1$ / $(PC) \leftarrow (PC) + 2$ if $T1 = 0$ | Jumps to the address specified by $a_0-a_7$ if test 1 is high. | 56 | 0 / $a_7$ | 1 / $a_6$ | 0 / $a_5$ | 1 / $a_4$ | 0 / $a_3$ | 1 / $a_2$ | 1 / $a_1$ | 0 / $a_0$ | 2 | 2 |
| JZ | $(PC_0-PC_7) \leftarrow$ addr if $A = 0$ / $(PC) \leftarrow (PC) + 2$ if $A = 1$ | Jump to the address specified by $a_0-a_7$ if the contents of the accumulator are equal to 0. | C6 | 1 / $a_7$ | 1 / $a_6$ | 0 / $a_5$ | 0 / $a_4$ | 0 / $a_3$ | 1 / $a_2$ | 1 / $a_1$ | 0 / $a_0$ | 2 | 2 |

## Instruction Set (cont)

| Mnemonic | Function | Description | Hex Code | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | Cycles | Bytes |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Control** | | | | | | | | | | | | | |
| EN I | | Enables external interrupts. When external interrupts are enabled, a low-level input to the INT pin causes the processor to vector to the interrupt service routine. | 05 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 |
| DIS I | | Disables external interrupts. When external interrupts are disabled, low-level inputs to the INT pin have no effect on program execution. | 15 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 |
| ENT0 CLK | | Enables clock output to pin T0. | 75 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 |
| SEL MB0 | (DBF) ← 0 | Clears the memory bank flip-flop, selecting program memory bank 0 (program memory addresses 0–2047$_{(10)}$). Clears PC$_{11}$ after the next JMP or CALL instruction. | E5 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 |
| SEL MB1 | (DBF) ← 1 | Sets the memory bank flip-flop, selecting program memory bank 1 (program memory addresses 2048–4095$_{(10)}$). Sets PC$_{11}$ after the next JMP or CALL instruction. | F5 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 |
| SEL RB0 | (BS) ← 0 | Selects data memory bank 0 by clearing bit 4 (bank switch) of the PSW. Specifies data memory addresses 0–7$_{(10)}$ as registers 0–7 of data memory bank 0. | C5 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 |
| SEL RB1 | (BS) ← 1 | Selects data memory bank 1 by setting bit 4 (bank switch) of the PSW. Specifies data memory 24–31$_{(10)}$ as registers 0–7 of data memory bank 1. | D5 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 |
| HALT | | Initiates halt mode. | 01 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| **Data Moves** | | | | | | | | | | | | | |
| MOV A, # data | (A) ← data | Moves immediate data d$_0$–d$_7$ into the accumulator. | 23 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 2 | 2 |
| | | | | d$_7$ | d$_6$ | d$_5$ | d$_4$ | d$_3$ | d$_2$ | d$_1$ | d$_0$ | | |
| MOV A, Rr | (A) ← (Rr); r = 0–7 | Moves the contents of register Rr into the accumulator. | Fn(4) | 1 | 1 | 1 | 1 | 1 | r | r | r | 1 | 1 |
| MOV A, @ Rr | (A) ← ((Rr)); r = 0–1 | Moves the contents of internal data memory specified by bits 0–5 in register Rr, into the accumulator. | Fn(4) | 1 | 1 | 1 | 1 | 0 | 0 | 0 | r | 1 | 1 |
| MOV A, PSW | (A) ← (PSW) | Moves the contents of the program status word into the accumulator. | C7 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| MOV Rr, # data | (Rr) ← data; r = 0–7 | Moves immediate data d$_0$–d$_7$ into register Rr. | Bn(4) | 1 | 0 | 1 | 1 | r | r | r | r | 2 | 2 |
| | | | | d$_7$ | d$_6$ | d$_5$ | d$_4$ | d$_3$ | d$_2$ | d$_1$ | d$_0$ | | |
| MOV Rr, A | (Rr) ← (A); r = 0–7 | Moves the contents of the accumulator into register Rr. | An(4) | 1 | 0 | 1 | 0 | 1 | r | r | r | 1 | 1 |
| MOV @ Rr, A | ((Rr)) ← (A); r = 0–1 | Moves the contents of the accumulator into the data memory location specified by bits 0–5 in register Rr. | An(4) | 1 | 0 | 1 | 0 | 0 | 0 | 0 | r | 1 | 1 |
| MOV @ Rr, # data | ((Rr)) ← data; r = 0–1 | Moves immediate data d$_0$–d$_7$ into the data memory location specified by bits 0–5 in register Rr. | Bn(4) | 1 | 0 | 1 | 1 | 0 | 0 | 0 | r | 2 | 2 |
| | | | | d$_7$ | d$_6$ | d$_5$ | d$_4$ | d$_3$ | d$_2$ | d$_1$ | d$_0$ | | |

## Instruction Set (cont)

### Data Moves (cont)

| Mnemonic | Function | Description | Hex Code | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | Cycles | Bytes |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MOV PSW, A | $(PSW) \leftarrow (A)$ | Moves the contents of the accumulator into the program status word. | D7 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| MOVP A, @ A | $(PC_0-PC_7) \leftarrow (A)$; $(A) \leftarrow ((PC))$ | Moves the contents of the program memory location specified by $PC_8-PC_{11}$ concatenated with the contents of the accumulator, into the accumulator. | A3 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 2 | 1 |
| MOVP3 A, @ A | $(PC_0-PC_7) \leftarrow (A)$; $(PC_8-PC_{11}) \leftarrow 001$; $(A) \leftarrow ((PC))$ | Moves the contents of the program memory location specified by 0011 ($PC_8-PC_{11}$, page 3 of program memory bank 0) and the contents of the accumulator, into the accumulator. | E3 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 2 | 1 |
| MOVX A, @ R | $(A) \leftarrow ((Rr))$; $r = 0-1$ | Moves the contents of the external data memory location specified by register Rr, into the accumulator. | 8n(4) | 1 | 0 | 0 | 0 | 0 | 0 | 0 | r | 2 | 1 |
| MOVX @ R, A | $((Rr)) \leftarrow (A)$; $r = 0-1$ | Moves the contents of the accumulator into the external data memory location specified by register Rr. | 9n(4) | 1 | 0 | 0 | 1 | 0 | 0 | 0 | r | 2 | 1 |
| XCH A, Rr | $(A) \leftrightarrow (Rr)$; $r = 0-7$ | Exchanges the contents of the accumulator and register Rr. | 2n(4) | 0 | 0 | 1 | 0 | 1 | r | r | r | 1 | 1 |
| XCH A, @ Rr | $(A) \leftrightarrow ((Rr))$; $r = 0-1$ | Exchanges the contents of the accumulator and the contents of the data memory location specified by bits 0–5 in register Rr. | 2n(4) | 0 | 0 | 1 | 0 | 0 | 0 | 0 | r | 1 | 1 |
| XCHD A, @ Rr | $(A_0-A_3) \leftrightarrow ((Rr))_0-((Rr))_3$; $r = 0-1$ | Exchanges the contents of the lower 4 bits of the accumulator with the contents of the lower 4 bits of the internal data memory location specified by bits 0–5 in register Rr. | 3n(4) | 0 | 0 | 1 | 1 | 0 | 0 | 0 | r | 1 | 1 |

### Flags

| Mnemonic | Function | Description | Hex Code | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | Cycles | Bytes |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CPL C | $(C) \leftarrow (\overline{C})$ | Takes the complement of the carry bit. | A7 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| CPL F0 | $(F0) \leftarrow (\overline{F0})$ | Takes the complement of flag 0. | 95 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 |
| CPL F1 | $(F1) \leftarrow (\overline{F1})$ | Takes the complement of flag 1. | B5 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 |
| CLR C | $(C) \leftarrow 0$ | Clears the carry bit. | 97 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| CLR F0 | $(F0) \leftarrow 0$ | Clears flag 0. | 85 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 |
| CLR F1 | $(F1) \leftarrow 0$ | Clears flag 1. | A5 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 |

## Instruction Set (cont)

### Input / Output

| Mnemonic | Function | Description | Hex Code | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | Cycles | Bytes |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ANL BUS, # data | (bus) ← (bus) AND data | Takes the logical AND of the contents of the bus and immediate data d0–d7, and sends the result to the bus. | 98 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 2 | 2 |
|  |  |  |  | d7 | d6 | d5 | d4 | d3 | d2 | d1 | d0 |  |  |
| ANL Pp, # data | (Pp) ← (Pp) AND data; p = 1–2 | Takes the logical AND of the contents of designated port Pp and immediate data d0–d7, and sends the result to port Pp for output. | 9n(5) | 1 | 0 | 0 | 1 | 1 | 0 | p | p | 2 | 2 |
|  |  |  |  | d7 | d6 | d5 | d4 | d3 | d2 | d1 | d0 |  |  |
| ANLD Pp, A | (Pp) ← (Pp) AND (A0–A3); p = 4–7 | Takes the logical AND of the contents of designated port Pp and the lower 4 bits of the accumulator, and sends the result to port Pp for output. | 9n(5) | 1 | 0 | 0 | 1 | 1 | 1 | p | p | 2 | 1 |
| IN A, Pp | (A) ← (Pp); p = 1–2 | Loads the accumulator with the contents of designated port Pp. | 0n(5) | 0 | 0 | 0 | 0 | 1 | 0 | p | p | 2 | 1 |
| INS A, BUS | (A) ← (bus) | Loads the contents of the bus into the accumulator on the rising edge of RD. | 08 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 2 | 1 |
| MOVD A, Pp | (A0–A3) ← (Pp); p = 4–7 (A4–A7) ← 0 | Moves the contents of designated port Pp to the lower 4 bits of the accumulator, and clears the upper 4 bits. | 0n(5) | 0 | 0 | 0 | 0 | 1 | 1 | p | p | 2 | 1 |
| MOVD Pp, A | (Pp) ← (A0–A3); p = 4–7 | Moves the lower 4 bits of the accumulator to designated port Pp. The upper 4 bits of the accumulator are not changed. | 3n(5) | 0 | 0 | 1 | 1 | 1 | 1 | p | p | 2 | 1 |
| ORL BUS, # data | (bus) ← (bus) OR data | Takes the logical OR of the contents of the bus and immediate data d0–d7, and sends the result to the bus. | 88 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 2 | 2 |
|  |  |  |  | d7 | d6 | d5 | d4 | d3 | d2 | d1 | d0 |  |  |
| ORLD Pp, A | (Pp) ← (Pp) OR (A0–A3); p = 4–7 | Takes the logical OR of the contents of designated port Pp and the lower 4 bits of the accumulator, and sends the result to port Pp for output. | 8n(5) | 1 | 0 | 0 | 0 | 1 | 1 | p | p | 2 | 1 |
| ORL Pp, # data | (Pp) ← (Pp) OR data; p = 1–2 | Takes the logical OR of the contents of designated port Pp and immediate data d0–d7, and sends the result to port Pp for output. | 8n(5) | 1 | 0 | 0 | 0 | 1 | 0 | p | p | 2 | 2 |
|  |  |  |  | d7 | d6 | d5 | d4 | d3 | d2 | d1 | d0 |  |  |
| OUTL BUS, A | (bus) ← (A) | Latches the contents of the accumulator onto the bus on the rising edge of WR. Note: Never use the OUTL BUS instruction when using external program memory, as this will permanently latch the bus. | 02 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 2 | 1 |
| OUTL Pp, A | (Pp) ← (A); p = 1–2 | Latches the contents of the accumulator into designated port Pp for output. | 3n(5) | 0 | 0 | 1 | 1 | 1 | 0 | p | p | 2 | 1 |

### Registers

| Mnemonic | Function | Description | Hex Code | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | Cycles | Bytes |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DEC Rr | (Rr) ← (Rr) − 1; r = 0–7 | Decrements the contents of register Rr by 1. | Cn(4) | 1 | 1 | 0 | 0 | 1 | r | r | r | 1 | 1 |
| INC Rr | (Rr) ← (Rr) + 1; r = 0–7 | Increments the contents of register Rr by 1. | 1n(4) | 0 | 0 | 0 | 1 | 1 | r | r | r | 1 | 1 |
| INC @Rr | ((Rr)) ← ((Rr)) + 1; r = 0–1 | Increments by 1 the contents of the data memory location specified by bits 0–5 in register Rr. | 1n(4) | 0 | 0 | 0 | 1 | 0 | 0 | 0 | r | 1 | 1 |

4

## Instruction Set (cont)

| Mnemonic | Function | Description | Hex Code | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | Cycles | Bytes |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Subroutine** | | | | | | | | | | | | | |
| CALL addr | $((SP)) \leftarrow (PC), (PSW_4-PSW_7)$ $(SP) \leftarrow (SP) + 1$ $(PC_8-PC_{10}) \leftarrow (addr_8-addr_{10})$ $(PC_0-PC_7) \leftarrow (addr_0-addr_7)$ $(PC_{11}) \leftarrow DBF$ | Stores the contents of the program counter and the upper 4 bits of the PSW in the address indicated by the stack pointer, and increments the contents of the stack pointer, calling the subroutine specified by address $a_0-a_{10}$ and the DBF. | x4(6) | $a_{10}$ $a_7$ | $a_9$ $a_6$ | $a_8$ $a_5$ | 1 $a_4$ | 0 $a_3$ | 1 $a_2$ | 0 $a_1$ | 0 $a_0$ | 2 | 2 |
| RET | $(SP) \leftarrow (SP) - 1$ $(PC) \leftarrow ((SP))$ | Decrements the contents of the stack pointer by 1 and stores, in the program counter, the contents of the location specified by the stack pointer, executing a return from subroutine without restoring the PSW. | 83 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 2 | 1 |
| RETR | $(SP) \leftarrow (SP) - 1$ $(PC) \leftarrow ((SP))$ $(PSW_4-PSW_7) \leftarrow ((SP))$ | Decrements the contents of the stack pointer by 1 and stores, in the program counter, the contents of the upper 4 bits of the PSW and the contents of the location specified by the stack pointer, executing a return from subroutine with restoration of the PSW. | 93 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 2 | 1 |
| **Timer / Counter** | | | | | | | | | | | | | |
| EN TCNTI | | Enables internal interrupt of timer / event counter. If an overflow condition occurs, then an interrupt will be generated. | 25 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 |
| DIS TCNTI | | Disables internal interrupt of timer / event counter. | 35 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 |
| MOV A, T | $(A) \leftarrow (T)$ | Moves the contents of the timer / counter into the accumulator. | 42 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 |
| MOV T, A | $(T) \leftarrow (A)$ | Moves the contents of the accumulator into the timer / counter. | 62 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 |
| STOP TCNT | | Stops the operation of the timer / event counter. | 65 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 |
| STRT CNT | | Starts the event counter operation of the timer / counter when T1 changes from a low-level input to a high-level input. | 45 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 |
| STRT T | | Starts the timer operation of the timer / counter. The timer is incremented every 32 machine cycles. | 55 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 |
| **Miscellaneous** | | | | | | | | | | | | | |
| NOP | | Uses one machine cycle without performing any operation. | 00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |

**4**

## Instruction Set (cont)

**Note:**

(1) Binary operation code designations r and p represent encoded values or the lowest-order bit value of specified registers and ports, respectively.

(2) Execution of the ADD, ADDC, and DA instructions affect the carry flags, which are not shown in the respective function equations. These instructions set the carry flags when there is an overflow in the accumulator (the auxiliary carry flag is set when there is an overflow of bit 3 of the accumulator) and clear the carry flags when there is no overflow. Flags that are specifically addressed by flag instructions are shown in the function equations for those instructions.

(3) References to addresses and data are specified in byte 1 and/or 2 in the opcode of the corresponding instruction.

(4) The hex value of n for specific registers is as follows:

a) Direct addressing

| | | | |
|---|---|---|---|
| R0: n = 8 | R2: n = A | R4: n = C | R6: n = E |
| R1: n = 9 | R3: n = B | R5: n = D | R7: n = F |

b) Indirect addressing

@ R0: n = 0    @ R1: n = 1

(5) The hex value of n for specific ports is as follows:

| | | |
|---|---|---|
| P1: n = 9 | P4: n = C | P6: n = E |
| P2: n = A | P5: n = D | P7: n = F |

(6) The hex value of x for specific accumulator or address bits is as follows:

a) JBb instruction

| | | | |
|---|---|---|---|
| B0: x = 1 | B2: x = 5 | B4: x = 9 | B6: x = D |
| B1: x = 3 | B3: x = 7 | B5: x = B | B7: x = F |

b) JMP instruction

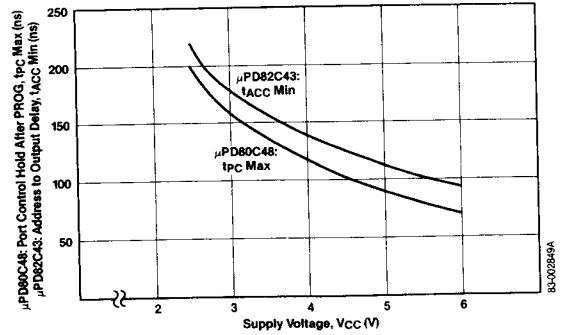| | | | |
|---|---|---|---|
| Page 0: x = 0 | Page 2: x = 4 | Page 4: x = 8 | Page 6: x = C |
| Page 1: x = 2 | Page 3: x = 6 | Page 5: x = A | Page 7: x = E |

c) CALL instruction

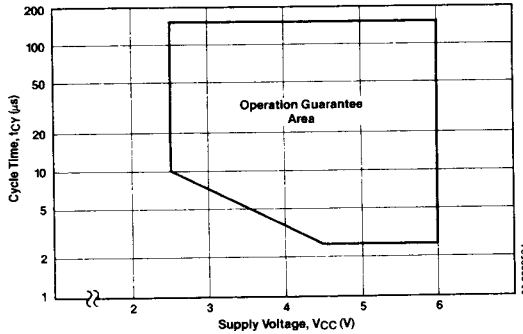| | | | |
|---|---|---|---|
| Page 0: x = 1 | Page 2: x = 5 | Page 4: x = 9 | Page 6: x = D |
| Page 1: x = 3 | Page 3: x = 7 | Page 5: x = B | Page 7: x = F |

## Operating Characteristics
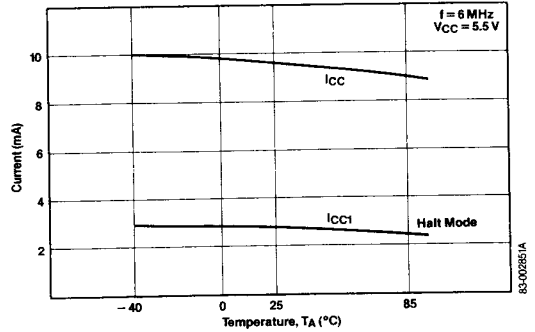
### Supply Current vs. Oscillation Frequency
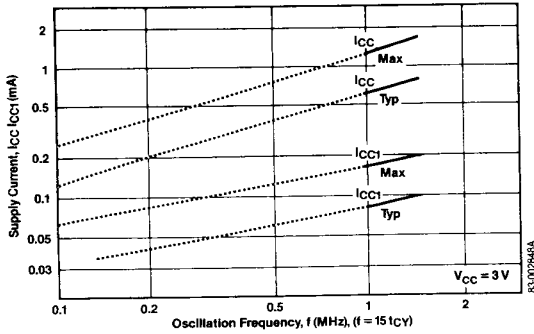


### Port Control Hold After PROG, tPC Max (μPD80C48), and Address to Output Delay, tACC Min (μPD82C43), vs. Supply Voltage
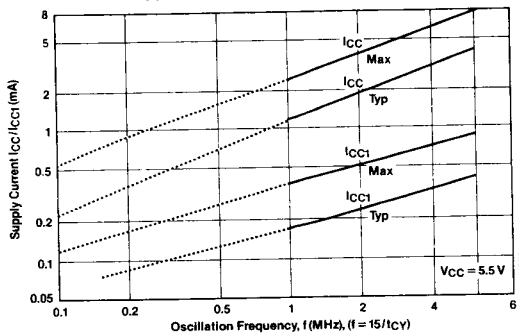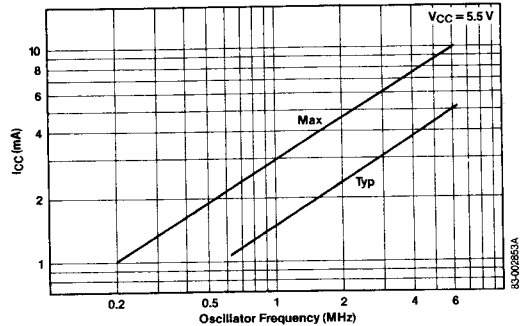


### Cycle Time vs. Supply Voltage



### Current Consumption as a Function of Temperature — Normal Operating Mode
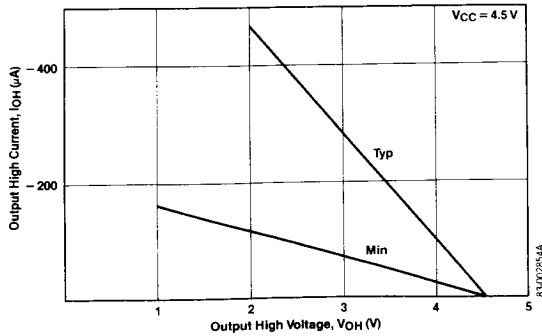


### Supply Current vs. Oscillation Frequency



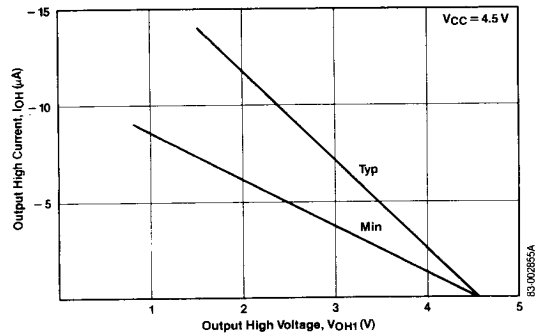### Current Consumption as a Function of Operating Frequency — Normal Operating Mode



Note: External oscillation is for frequency less than 1 MHz.
Internal oscillation requires more power.

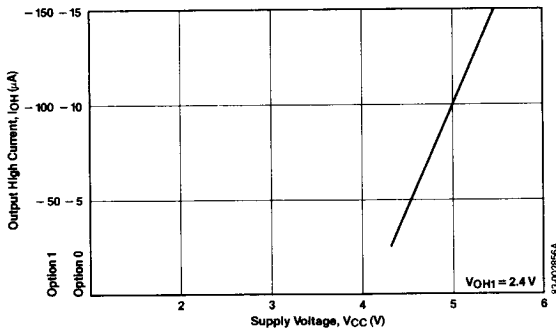## Operating Characteristics (cont)
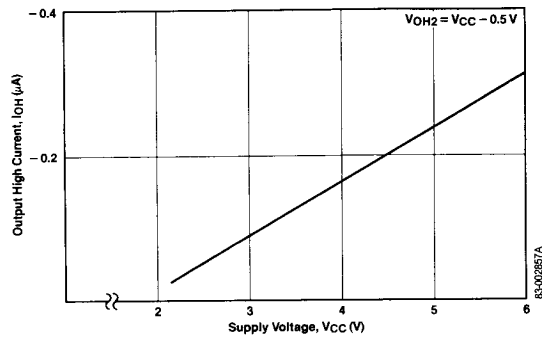
Output High Current vs. Output High Voltage

$V_{CC} = 4.5$ V

Output High Current, $I_{OH}$ (µA)

Typ

Min

Output High Voltage, $V_{OH}$ (V)

83-002854A

Output High Current vs. Output High Voltage

$V_{CC} = 4.5$ V

Output High Current, $I_{OH}$ (µA)

Typ

Min

Output High Voltage, $V_{OH1}$ (V)

83-002855A

Output High Current vs. Supply Voltage

Output High Current, $I_{OH}$ (µA)

Option 1  Option 0

$V_{OH1} = 2.4$ V

Supply Voltage, $V_{CC}$ (V)

83-002856A

Output High Current vs. Supply Voltage

$V_{OH2} = V_{CC} - 0.5$ V

Output High Current, $I_{OH}$ (µA)

Supply Voltage, $V_{CC}$ (V)

83-002857A

Output Low Current vs. Supply Voltage

Output Low Current, $I_{OL}$ (mA)

$V_{OL} = 0.45$ V

Supply Voltage, $V_{CC}$ (V)

83-002858A
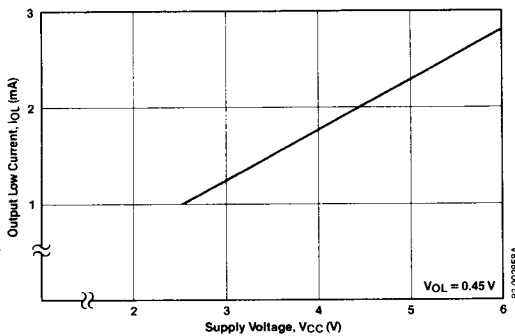
Output Low Current vs. Output Low Voltage

Output Low Current, $I_{OL}$ (mA)

$V_{CC} = 4.5$ V

Output Low Voltage, $V_{OL}$ (V)

83-002859A

4-233

## Operating Characteristics (cont)

**Current Consumption as a Function
of Temperature — Stop Mode**