

## COP8ACC7

### 8-Bit CMOS OTP Microcontroller with 16k Memory and High Resolution A/D

#### General Description

The COP8ACC7 OTP (One Time Programmable) microcontrollers are highly integrated COP8™ Feature core devices with 16k memory and advanced features including a High-Resolution A/D. This multi-chip CMOS device is suited for applications requiring a full featured controller with a high resolution A/D (only one external capacitor required), and for pre-production devices for a ROM design. Pin and software compatible (different  $V_{CC}$  range) 4k ROM versions are available (COPACC5). Erasable windowed versions are available for use with a range of COP8 software and hardware development tools.

Family features include an 8-bit memory mapped architecture, 4 MHz CKI with 2.5  $\mu$ s instruction cycle, two external clock options (–XE = Crystal; –RE = RC), 6 channel A/D with 12-bit resolution, analog capture timer, analog current source and  $V_{CC}/2$  reference, one multi-function 16-bit timer/counter, MICROWIRE/PLUS™ serial I/O, two power saving HALT/IDLE modes, MIWU, high current outputs, software selectable I/O options, WATCHDOG™ timer and Clock Monitor, 2.7V to 5.5V operation, program code security, and 20/28 pin packages.

Device included in this datasheet is:

| Device      | Memory (bytes) | RAM (bytes) | I/O Pins | Packages             | Temperature  |
|-------------|----------------|-------------|----------|----------------------|--------------|
| COP8ACC7xx9 | 16k OTP EPROM  | 128         | 15/23    | 20 SOIC, 28 DIP/SOIC | 0 to +70°C   |
| COP8ACC7xx8 | 16k OTP EPROM  | 128         | 15/23    | 20 SOIC, 28 DIP/SOIC | -40 to +85°C |

#### Key Features

- Analog Function Block with 12-bit A/D including:
  - Analog comparator with seven input muxes
  - Constant Current Source and  $V_{CC}/2$  Reference
  - 16-bit capture timer (upcounter) clocked from CKI with auto reset on timer startup
- Quiet design (reduced radiated emissions)
- 4096 bytes on-board OTP EPROM with security feature
- 128 bytes on-board RAM

#### Additional Peripheral Features

- Idle Timer
- One 16-bit timer with two 16-bit registers supporting:
  - Processor Independent PWM mode
  - External Event counter mode
  - Input Capture mode
- Multi-Input Wake-Up (MIWU) with optional interrupts
- WATCHDOG and clock monitor logic
- MICROWIRE/PLUS serial I/O with programmable shift clock-polarity

#### I/O Features

- Software selectable I/O options (Push-Pull Output, Weak Pull-Up Input, High Impedance Input)
- High current outputs
- Schmitt Trigger inputs on ports G and L
- Packages:
  - 28 DIP/SO with 23 I/O pins
  - 20 SO with 15 I/O pins

#### CPU/Instruction Set Features

- 2.5  $\mu$ s instruction cycle time
- Eight multi-source vectored interrupt servicing:
  - External Interrupt
  - Idle Timer T0
  - Timer T1 associated Interrupts
  - MICROWIRE/PLUS
  - Multi-Input Wake Up
  - Software Trap
  - Default VIS
  - A/D (Capture Timer)
- 8-bit Stack Pointer (SP)—stack in RAM
- Two 8-bit Registers Indirect Data Memory Pointers (B and X)

#### Fully Static CMOS

- Two power saving modes: HALT and IDLE
- Temperature ranges: 0°C to +70°C, -40°C to +85°C
- Available with Crystal (–XE) or R/C (–RE) oscillator

#### Development System

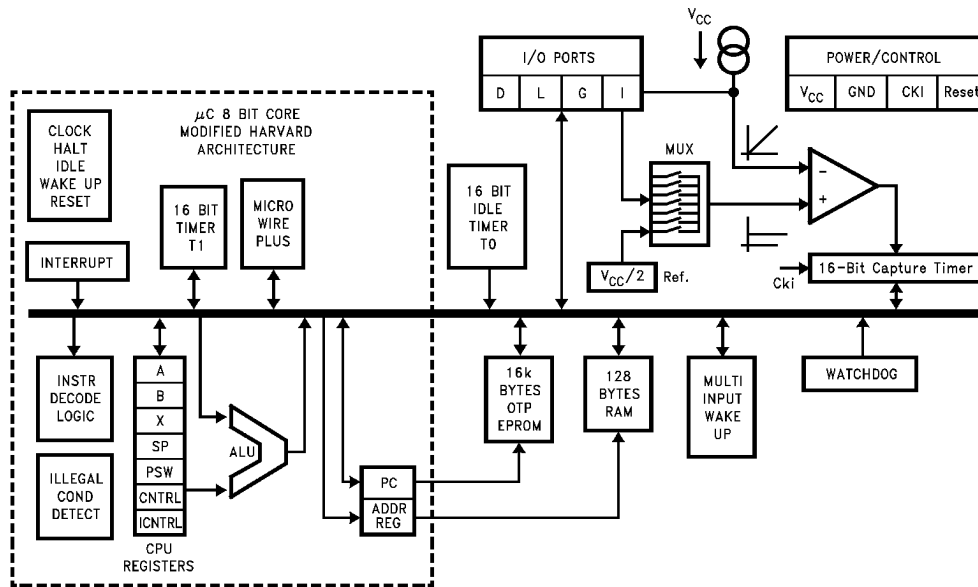
- Emulation device for COP8ACC5
- Real time emulation and full program debug offered by MetaLink® development system

#### Applications

- Battery Chargers
- Appliances
- Data Acquisition systems

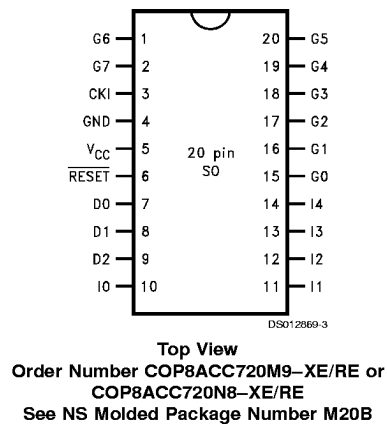
Driveway™ is a trademark of Aisys Intelligent Systems.  
 COP8™, MICROWIRE™, MICROWIRE/PLUS™, and WATCHDOG™ are trademarks of National Semiconductor Corporation.  
 TRI-STATE® is a registered trademark of National Semiconductor Corporation.  
 iceMASTER® is a registered trademark of MetaLink Corporation.

### Block Diagram

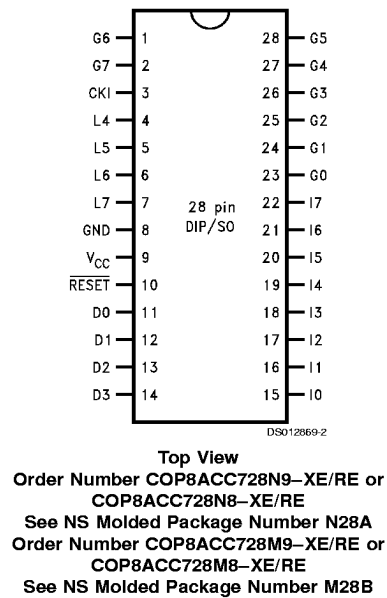


**FIGURE 1. Block Diagram**

## Connection Diagrams



**Note: -X Crystal Oscillator  
-R R/C Oscillator  
-E Halt Enable**



### FIGURE 2. Connection Diagrams

## Connection Diagrams (Continued)

### Pinouts for 28-Pin, 20-Pin Packages

| Port            | Type  | Alt. Fun         | Alt. Fun  | 28-Pin<br>DIP/SO | 20-Pin<br>SO |
|-----------------|-------|------------------|-----------|------------------|--------------|
| L4              | I/O   | MIWU             | Ext. Int. | 4                |              |
| L5              | I/O   | MIWU             | Ext. Int. | 5                |              |
| L6              | I/O   | MIWU             | Ext. Int. | 6                |              |
| L7              | I/O   | MIWU             | Ext. Int. | 7                |              |
| G0              | I/O   | INT              |           | 23               | 15           |
| G1              | WDOUT |                  |           | 24               | 16           |
| G2              | I/O   | T1B              |           | 25               | 17           |
| G3              | I/O   | T1A              |           | 26               | 18           |
| G4              | I/O   | SO               |           | 27               | 19           |
| G5              | I/O   | SK               |           | 28               | 20           |
| G6              | I     | SI               |           | 1                | 1            |
| G7              | I/CKO | HALT Restart     |           | 2                | 2            |
| D0              | O     |                  |           | 11               | 7            |
| D1              | O     |                  |           | 12               | 8            |
| D2              | O     |                  |           | 13               | 9            |
| D3              | O     |                  |           | 14               |              |
| I0              | I     | Analog CH1       |           | 15               | 10           |
| I1              | I     | I <sub>src</sub> |           | 16               | 11           |
| I2              | I     | Analog CH2       |           | 17               | 12           |
| I3              | I     | Analog CH3       |           | 18               | 13           |
| I4              | I     | Analog CH4       |           | 19               | 14           |
| I5              | I     | Analog CH5       |           | 20               |              |
| I6              | I     | Analog CH6       |           | 21               |              |
| I7              | I     | C <sub>OUT</sub> |           | 22               |              |
| V <sub>CC</sub> |       |                  |           | 9                | 5            |
| GND             |       |                  |           | 8                | 4            |
| CKI             |       |                  |           | 3                | 3            |
| RESET           |       |                  |           | 10               | 6            |

## Ordering Information

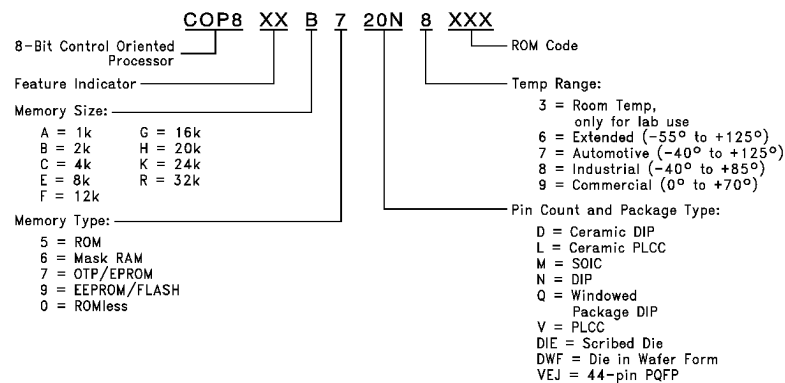


FIGURE 3. Part Numbering Scheme

DS012869-39

## Absolute Maximum Ratings (Note 1)

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

Supply Voltage ( $V_{CC}$ ) 7V  
Voltage at Any Pin  $-0.3V$  to  $V_{CC} + 0.3V$

Total Current into  $V_{CC}$  Pin (Source) 100 mA  
Total Current out of GND Pin (Sink) 110 mA  
Storage Temperature Range  $-65^{\circ}C$  to  $+140^{\circ}C$

**Note 1:** Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.

## DC Electrical Characteristics

$0^{\circ}C \leq T_A \leq +70^{\circ}C$  unless otherwise specified

| Parameter                            | Conditions                          | Min          | Typ | Max           | Units   |
|--------------------------------------|-------------------------------------|--------------|-----|---------------|---------|
| Operating Voltage                    | Peak-to-Peak                        | 2.7          |     | 5.5           | V       |
| Power Supply Ripple (Note 2)         |                                     |              |     | $0.1 V_{CC}$  | V       |
| Supply Current (Note 3)              |                                     |              |     |               |         |
| CKI = 4 MHz                          | $V_{CC} = 5.5V$ , $t_C = 2.5 \mu s$ |              |     | 9.5           | mA      |
| CKI = 4 MHz                          | $V_{CC} = 4V$ , $t_C = 2.5 \mu s$   |              |     | 6.5           | mA      |
| CKI = 1 MHz                          | $V_{CC} = 4V$ , $t_C = 10 \mu s$    |              |     | 5.4           | mA      |
| HALT Current (Note 4)                | $V_{CC} = 5.5V$ , CKI = 0 MHz       |              | < 5 | 10            | $\mu A$ |
|                                      | $V_{CC} = 4V$ , CKI = 0 MHz         |              | < 3 | 6             | $\mu A$ |
| IDLE Current                         |                                     |              |     |               |         |
| CKI = 4 MHz                          | $V_{CC} = 5.5V$ , $t_C = 2.5 \mu s$ |              |     | 1.5           | mA      |
| CKI = 1 MHz                          | $V_{CC} = 4V$ , $t_C = 10 \mu s$    |              |     | 0.5           | mA      |
| Input Levels ( $V_{IH}$ , $V_{IL}$ ) |                                     |              |     |               |         |
| RESET                                |                                     |              |     |               |         |
| Logic High                           |                                     | $0.8 V_{CC}$ |     |               | V       |
| Logic Low                            |                                     |              |     | $0.2 V_{CC}$  | V       |
| CKI, All Other Inputs                |                                     |              |     |               |         |
| Logic High                           |                                     | $0.7 V_{CC}$ |     |               | V       |
| Logic Low                            |                                     |              |     | $0.2 V_{CC}$  | V       |
| Hi-Z Input Leakage                   | $V_{CC} = 5.5V$                     | 1            |     | 1             | $\mu A$ |
| Input Pullup Current                 | $V_{CC} = 5.5V$ , $V_{IN} = 0V$     | -40          |     | -250          | $\mu A$ |
| G and L Port Input Hysteresis        | (Note 6)                            |              |     | $0.35 V_{CC}$ | V       |
| Output Current Levels                |                                     |              |     |               |         |
| D Outputs                            |                                     |              |     |               |         |
| Source                               | $V_{CC} = 4V$ , $V_{OH} = 3.3V$     | -0.4         |     |               | mA      |
|                                      | $V_{CC} = 2.7V$ , $V_{OH} = 1.8V$   | -0.2         |     |               | mA      |
| Sink                                 | $V_{CC} = 4V$ , $V_{OL} = 1V$       | 10           |     |               | mA      |
|                                      | $V_{CC} = 2.7V$ , $V_{OL} = 0.4V$   | 2.0          |     |               | mA      |
| All Others                           |                                     |              |     |               |         |
| Source (Weak Pull-Up Mode)           | $V_{CC} = 4V$ , $V_{OH} = 2.7V$     | -10          |     | -110          | $\mu A$ |
|                                      | $V_{CC} = 2.7V$ , $V_{OH} = 1.8V$   | -2.5         |     | -33           | $\mu A$ |
| Source (Push-Pull Mode)              | $V_{CC} = 4V$ , $V_{OH} = 3.3V$     | -0.4         |     |               | mA      |
|                                      | $V_{CC} = 2.7V$ , $V_{OH} = 1.8V$   | -0.2         |     |               | mA      |
| Sink (Push-Pull Mode)                | $V_{CC} = 4V$ , $V_{OL} = 0.4V$     | 1.6          |     |               | mA      |
|                                      | $V_{CC} = 2.7V$ , $V_{OL} = 0.4V$   | 0.7          |     |               | mA      |
| TRI-STATE® Leakage                   | $V_{CC} = 5.5V$                     | 1            |     | 1             | $\mu A$ |
| Allowable Sink/Source                |                                     |              |     |               |         |
| Current per Pin                      |                                     |              |     |               |         |
| D Outputs (Sink)                     |                                     |              |     | 15            | mA      |
| All others                           |                                     |              |     | 3             | mA      |
| Maximum Input Current                | Room Temp                           |              |     | $\pm 200$     | mA      |
| without Latchup (Note 5)             |                                     |              |     |               |         |
| RAM Retention Voltage, $V_r$         | 500 ns Rise and Fall Time (min)     | 2            |     |               | V       |

## DC Electrical Characteristics (Continued)

0°C ≤ T<sub>A</sub> ≤ +70°C unless otherwise specified

| Parameter              | Conditions | Min | Typ | Max  | Units |
|------------------------|------------|-----|-----|------|-------|
| Input Capacitance      | (Note 6)   |     |     | 7    | pF    |
| Load Capacitance on D2 | (Note 6)   |     |     | 1000 | pF    |

## AC Electrical Characteristics

0°C ≤ T<sub>A</sub> ≤ +70°C unless otherwise specified

| Parameter                                              | Conditions                                     | Min | Typ | Max  | Units          |
|--------------------------------------------------------|------------------------------------------------|-----|-----|------|----------------|
| Instruction Cycle Time (Note 8)                        |                                                |     |     |      |                |
| Crystal, Resonator                                     | 2.7V ≤ V <sub>CC</sub> ≤ 4V                    | 2.5 |     | DC   | μs             |
|                                                        | 4V ≤ V <sub>CC</sub> ≤ 5.5V                    | 1.0 |     | DC   | μs             |
| R/C Oscillator                                         | 2.7V ≤ V <sub>CC</sub> ≤ 4V                    | 7.5 |     | DC   | μs             |
|                                                        | 4V ≤ V <sub>CC</sub> ≤ 5.5V                    | 3.0 |     | DC   | μs             |
| Inputs                                                 |                                                |     |     |      |                |
| t <sub>SETUP</sub>                                     | 4V ≤ V <sub>CC</sub> ≤ 5.5V                    | 200 |     |      | ns             |
|                                                        | 2.7V ≤ V <sub>CC</sub> ≤ 4V                    | 500 |     |      | ns             |
| t <sub>HOLD</sub>                                      | 4V ≤ V <sub>CC</sub> ≤ 5.5V                    | 60  |     |      | ns             |
|                                                        | 2.7V ≤ V <sub>CC</sub> ≤ 4V                    | 150 |     |      | ns             |
| Output Propagation Delay (Note 6)                      | R <sub>L</sub> = 2.2k, C <sub>L</sub> = 100 pF |     |     |      |                |
| t <sub>PD1</sub> , t <sub>PD0</sub>                    |                                                |     |     |      |                |
| SO, SK                                                 | 4V ≤ V <sub>CC</sub> ≤ 5.5V                    |     |     | 0.7  | μs             |
|                                                        | 2.7V ≤ V <sub>CC</sub> ≤ 4V                    |     |     | 1.75 | μs             |
| All Others                                             | 4V ≤ V <sub>CC</sub> ≤ 5.5V                    |     |     | 1    | μs             |
|                                                        | 2.7V ≤ V <sub>CC</sub> ≤ 4V                    |     |     | 2.5  | μs             |
| MICROWIRE™ Setup Time (t <sub>UWS</sub> ) (Note 6)     | V <sub>CC</sub> ≥ 4V                           | 20  |     |      | ns             |
| MICROWIRE Hold Time (t <sub>UWH</sub> ) (Note 6)       | V <sub>CC</sub> ≥ 4V                           | 56  |     |      | ns             |
| MICROWIRE Output Propagation Delay (t <sub>UPD</sub> ) | V <sub>CC</sub> ≥ 4V                           |     |     | 220  | ns             |
| Input Pulse Width (Note 7)                             |                                                |     |     |      |                |
| Interrupt Input High Time                              |                                                | 1   |     |      | t <sub>C</sub> |
| Interrupt Input Low Time                               |                                                | 1   |     |      | t <sub>C</sub> |
| Timer 1, 2, 3 Input High Time                          |                                                | 1   |     |      | t <sub>C</sub> |
| Timer 1, 2, 3 Input Low Time                           |                                                | 1   |     |      | t <sub>C</sub> |
| Reset Pulse Width                                      |                                                | 1   |     |      | μs             |

**Note 2:** Maximum rate of voltage change must be < 0.5V/ms.

**Note 3:** Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

**Note 4:** The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Measurement of I<sub>DD</sub> HALT is done with device neither sourcing or sinking current; with L, C, and G0–G5 programmed as low outputs and not driving a load; all outputs programmed low and not driving a load; all inputs tied to V<sub>CC</sub>; clock monitor and comparator disabled. Parameter refers to HALT mode entered via setting bit 7 of the G Port data register. Part will pull up CKI during HALT in crystal clock mode.

**Note 5:** Pins G6 and RESET are designed with a high voltage input network. These pins allow input voltages > V<sub>CC</sub> and the pins will have sink current to V<sub>CC</sub> when biased at voltages > V<sub>CC</sub> (the pins do not have source current when biased at a voltage below V<sub>CC</sub>). The effective resistance to V<sub>CC</sub> is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V. WARNING: Voltages in excess of 14V will cause damage to the pins. This warning excludes ESD transients.

**Note 6:** The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

**Note 7:** Parameter characterized but not tested.

**Note 8:** t<sub>C</sub> = Instruction Cycle Time.

## Absolute Maximum Ratings (Note 9)

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

Supply Voltage ( $V_{CC}$ ) 7V  
Voltage at Any Pin  $-0.3V$  to  $V_{CC} + 0.3V$

Total Current into  $V_{CC}$  Pin (Source) 100 mA  
Total Current out of GND Pin (Sink) 110 mA  
Storage Temperature Range  $-65^{\circ}C$  to  $+140^{\circ}C$

**Note 9:** Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.

## DC Electrical Characteristics

$-40^{\circ}C \leq T_A \leq +85^{\circ}C$  unless otherwise specified

| Parameter                         | Conditions                       | Min          | Typ | Max           | Units   |
|-----------------------------------|----------------------------------|--------------|-----|---------------|---------|
| Operating Voltage                 |                                  | 2.7          |     | 5.5           | V       |
| Power Supply Ripple (Note 10)     | Peak-to-Peak                     |              |     | $0.1 V_{CC}$  | V       |
| Supply Current (Note 11)          |                                  |              |     |               |         |
| CKI = 4 MHz                       | $V_{CC} = 5.5V, t_C = 2.5 \mu s$ |              |     | 9.5           | mA      |
| CKI = 4 MHz                       | $V_{CC} = 4V, t_C = 2.5 \mu s$   |              |     | 6.5           | mA      |
| CKI = 1 MHz                       | $V_{CC} = 4V, t_C = 10 \mu s$    |              |     | 5.4           | mA      |
| HALT Current (Note 12)            | $V_{CC} = 5.5V, CKI = 0$ MHz     |              | < 5 | 12            | $\mu A$ |
|                                   | $V_{CC} = 4V, CKI = 0$ MHz       |              | < 3 | 8             | $\mu A$ |
| IDLE Current                      |                                  |              |     |               |         |
| CKI = 4 MHz                       | $V_{CC} = 5.5V, t_C = 2.5 \mu s$ |              |     | 1.5           | mA      |
| CKI = 1 MHz                       | $V_{CC} = 4V, t_C = 10 \mu s$    |              |     | 0.5           | mA      |
| Input Levels ( $V_{IH}, V_{IL}$ ) |                                  |              |     |               |         |
| RESET                             |                                  |              |     |               |         |
| Logic High                        |                                  | $0.8 V_{CC}$ |     |               | V       |
| Logic Low                         |                                  |              |     | $0.2 V_{CC}$  | V       |
| CKI, All Other Inputs             |                                  |              |     |               |         |
| Logic High                        |                                  | $0.7 V_{CC}$ |     |               | V       |
| Logic Low                         |                                  |              |     | $0.2 V_{CC}$  | V       |
| Hi-Z Input Leakage                | $V_{CC} = 5.5V$                  | -2           |     | +2            | $\mu A$ |
| Input Pullup Current              | $V_{CC} = 5.5V, V_{IN} = 0V$     | -40          |     | -250          | $\mu A$ |
| G and L Port Input Hysteresis     | (Note 14)                        |              |     | $0.35 V_{CC}$ | V       |
| Output Current Levels             |                                  |              |     |               |         |
| D Outputs                         |                                  |              |     |               |         |
| Source                            | $V_{CC} = 4V, V_{OH} = 3.3V$     | -0.4         |     |               | mA      |
|                                   | $V_{CC} = 2.7V, V_{OH} = 1.8V$   | -0.2         |     |               | mA      |
| Sink                              | $V_{CC} = 4V, V_{OL} = 1V$       | 10           |     |               | mA      |
|                                   | $V_{CC} = 2.7V, V_{OL} = 0.4V$   | 2.0          |     |               | mA      |
| All Others                        |                                  |              |     |               |         |
| Source (Weak Pull-Up Mode)        | $V_{CC} = 4V, V_{OH} = 2.7V$     | -10          |     | -110          | $\mu A$ |
|                                   | $V_{CC} = 2.7V, V_{OH} = 1.8V$   | -2.5         |     | -33           | $\mu A$ |
| Source (Push-Pull Mode)           | $V_{CC} = 4V, V_{OH} = 3.3V$     | -0.4         |     |               | mA      |
|                                   | $V_{CC} = 2.7V, V_{OH} = 1.8V$   | -0.2         |     |               | mA      |
| Sink (Push-Pull Mode)             | $V_{CC} = 4V, V_{OL} = 0.4V$     | 1.6          |     |               | mA      |
|                                   | $V_{CC} = 2.7V, V_{OL} = 0.4V$   | 0.7          |     |               | mA      |
| TRI-STATE Leakage                 | $V_{CC} = 5.5V$                  | -2           |     | +2            | $\mu A$ |
| Allowable Sink/Source             |                                  |              |     |               |         |
| Current per Pin                   |                                  |              |     |               |         |
| D Outputs (Sink)                  |                                  |              |     | 15            | mA      |
| All others                        |                                  |              |     | 3             | mA      |
| Maximum Input Current             | Room Temp                        |              |     | $\pm 200$     | mA      |
| without Latchup (Note 13)         |                                  |              |     |               |         |
| RAM Retention Voltage, $V_r$      | 500 ns Rise and Fall Time (min)  | 2            |     |               | V       |

## DC Electrical Characteristics (Continued)

–40°C ≤ T<sub>A</sub> ≤ +85°C unless otherwise specified

| Parameter              | Conditions | Min | Typ | Max  | Units |
|------------------------|------------|-----|-----|------|-------|
| Input Capacitance      | (Note 14)  |     |     | 7    | pF    |
| Load Capacitance on D2 | (Note 14)  |     |     | 1000 | pF    |

## AC Electrical Characteristics

–40°C ≤ T<sub>A</sub> ≤ +85°C unless otherwise specified

| Parameter                                              | Conditions                                     | Min | Typ | Max  | Units          |
|--------------------------------------------------------|------------------------------------------------|-----|-----|------|----------------|
| Instruction Cycle Time (Note 16)                       |                                                |     |     |      |                |
| Crystal, Resonator                                     | 2.7V ≤ V <sub>CC</sub> < 4V                    | 2.5 |     | DC   | μs             |
|                                                        | 4V ≤ V <sub>CC</sub> ≤ 5.5V                    | 1.0 |     | DC   | μs             |
| R/C Oscillator                                         | 2.7V ≤ V <sub>CC</sub> < 4V                    | 7.5 |     | DC   | μs             |
|                                                        | 4V ≤ V <sub>CC</sub> < 5.5V                    | 3.0 |     | DC   | μs             |
| Inputs                                                 |                                                |     |     |      |                |
| t <sub>SETUP</sub>                                     | 4V ≤ V <sub>CC</sub> ≤ 5.5V                    | 200 |     |      | ns             |
|                                                        | 2.7V ≤ V <sub>CC</sub> < 4V                    | 500 |     |      | ns             |
| t <sub>HOLD</sub>                                      | 4V ≤ V <sub>CC</sub> ≤ 5.5V                    | 60  |     |      | ns             |
|                                                        | 2.7V ≤ V <sub>CC</sub> < 4V                    | 150 |     |      | ns             |
| Output Propagation Delay (Note 14)                     | R <sub>L</sub> = 2.2k, C <sub>L</sub> = 100 pF |     |     |      |                |
| t <sub>PD1</sub> , t <sub>PD0</sub>                    |                                                |     |     |      |                |
| SO, SK                                                 | 4V ≤ V <sub>CC</sub> ≤ 5.5V                    |     |     | 0.7  | μs             |
|                                                        | 2.7V ≤ V <sub>CC</sub> < 4V                    |     |     | 1.75 | μs             |
| All Others                                             | 4V ≤ V <sub>CC</sub> ≤ 5.5V                    |     |     | 1    | μs             |
|                                                        | 2.7V ≤ V <sub>CC</sub> < 4V                    |     |     | 2.5  | μs             |
| MICROWIRE Setup Time (t <sub>UWS</sub> ) (Note 14)     | V <sub>CC</sub> ≥ 4V                           | 20  |     |      | ns             |
| MICROWIRE Hold Time (t <sub>UWH</sub> ) (Note 14)      | V <sub>CC</sub> ≥ 4V                           | 56  |     |      | ns             |
| MICROWIRE Output Propagation Delay (t <sub>UPD</sub> ) | V <sub>CC</sub> ≥ 4V                           |     |     | 220  | ns             |
| Input Pulse Width (Note 15)                            |                                                |     |     |      |                |
| Interrupt Input High Time                              |                                                | 1   |     |      | t <sub>C</sub> |
| Interrupt Input Low Time                               |                                                | 1   |     |      | t <sub>C</sub> |
| Timer 1, 2, 3 Input High Time                          |                                                | 1   |     |      | t <sub>C</sub> |
| Timer 1, 2, 3 Input Low Time                           |                                                | 1   |     |      | t <sub>C</sub> |
| Reset Pulse Width                                      |                                                | 1   |     |      | μs             |

**Note 10:** Maximum rate of voltage change must be < 0.5 V/ms.

**Note 11:** Supply current is measured after running 2000 cycles with a square wave CKI input, CKO open, inputs at rails and outputs open.

**Note 12:** The HALT mode will stop CKI from oscillating in the RC and the Crystal configurations. Measurement of I<sub>DD</sub> HALT is done with device neither sourcing or sinking current; with L, C, and G0–G5 programmed as low outputs and not driving a load; all outputs programmed low and not driving a load; all inputs tied to V<sub>CC</sub>; clock monitor and comparator disabled. Parameter refers to HALT mode entered via setting bit 7 of the G Port data register. Part will pull up CKI during HALT in crystal clock mode.

**Note 13:** Pins G6 and RESET are designed with a high voltage input network. These pins allow input voltages > V<sub>CC</sub> and the pins will have sink current to V<sub>CC</sub> when biased at voltages greater than V<sub>CC</sub> (the pins do not have source current when biased at a voltage below V<sub>CC</sub>). The effective resistance to V<sub>CC</sub> is 750Ω (typical). These two pins will not latch up. The voltage at the pins must be limited to less than 14V. WARNING: Voltages in excess of 14V will cause damage to the pins. This warning excludes ESD transients.

**Note 14:** The output propagation delay is referenced to the end of the instruction cycle where the output change occurs.

**Note 15:** Parameter characterized but not tested.

**Note 16:** t<sub>C</sub> = Instruction Cycle Time.

## Comparator AC and DC Characteristics

$V_{CC} = 5V$ ,  $-40^{\circ}C \leq T_A \leq +85^{\circ}C$

| Parameter                                                       | Conditions                                | Min                 | Typ          | Max                 | Units   |
|-----------------------------------------------------------------|-------------------------------------------|---------------------|--------------|---------------------|---------|
| Input Offset Voltage                                            | $0.4V < V_{IN} < V_{CC}$<br>$-1.5V$       |                     | 10           | 25                  | mV      |
| Input Common Mode Voltage Range<br>(Note 17)                    |                                           | 0.4                 |              | $V_{CC} - 1.5$      | V       |
| Voltage Gain                                                    |                                           |                     | 300k         |                     | V/V     |
| $V_{CC}/2$ Reference                                            | $4.0V < V_{CC} < 5.5V$                    | $0.5 V_{CC} - 0.04$ | $0.5 V_{CC}$ | $0.5 V_{CC} + 0.04$ | V       |
| DC Supply Current<br>For Comparator (when enabled)              | $V_{CC} = 5.5V$                           |                     |              | 250                 | $\mu A$ |
| DC Supply Current<br>For $V_{CC}/2$ reference (when enabled)    | $V_{CC} = 5.5V$                           |                     | 50           | 80                  | $\mu A$ |
| DC Supply Current<br>For Constant Current Source (when enabled) | $V_{CC} = 5.5V$                           |                     |              | 200                 | $\mu A$ |
| Constant Current Source                                         | $4.0V < V_{CC} < 5.5V$                    | 7                   | 20           | 32                  | $\mu A$ |
| Current Source Variation                                        | $4.0V < V_{CC} < 5.5V$<br>Temp = Constant |                     |              | 2                   | $\mu A$ |
| Current Source Enable Time                                      |                                           |                     | 1.5          | 2                   | $\mu s$ |
| Comparator Response Time                                        | 10 mV overdrive,<br>100 pF load           |                     |              | 1                   | $\mu s$ |

**Note 17:** The device is capable of operating over a common mode voltage range of 0 to  $V_{CC} - 1.5V$ , however increased offset voltage will be observed between 0V and 0.4V.

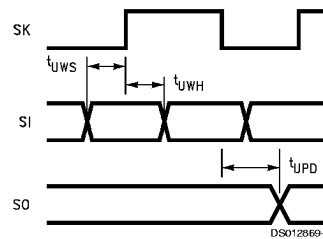
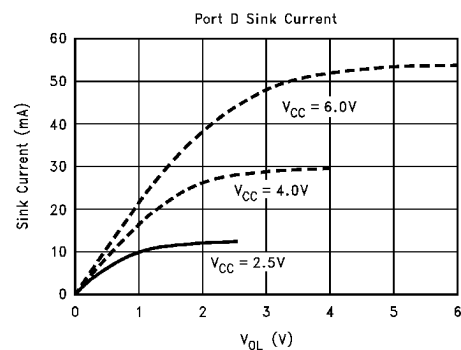
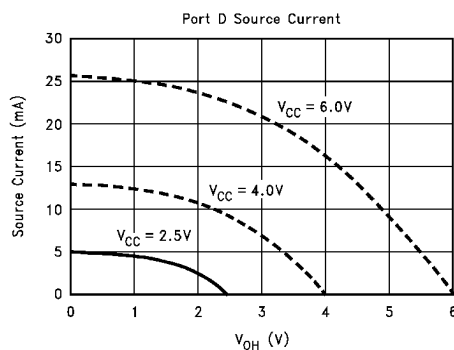


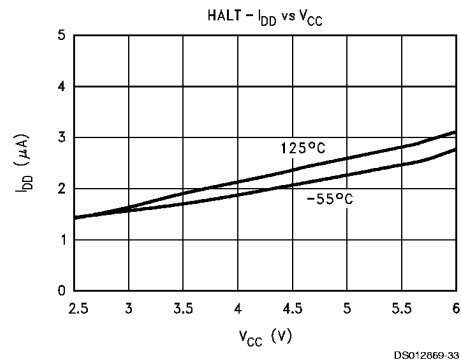
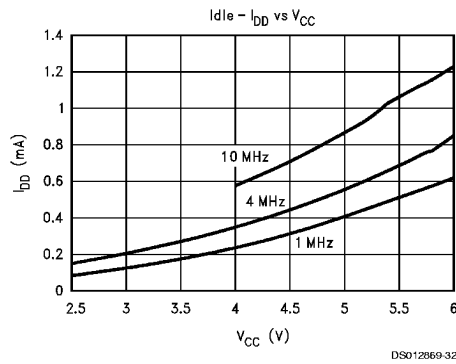
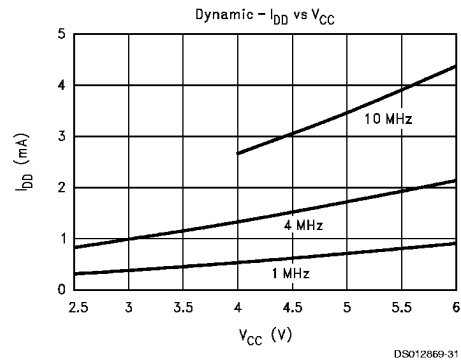
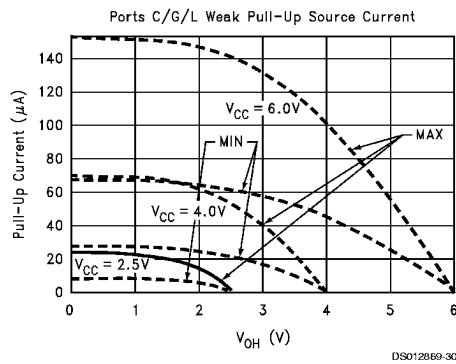
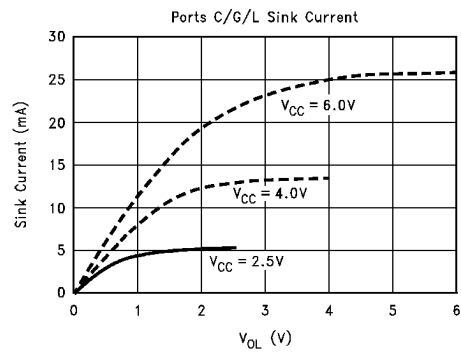
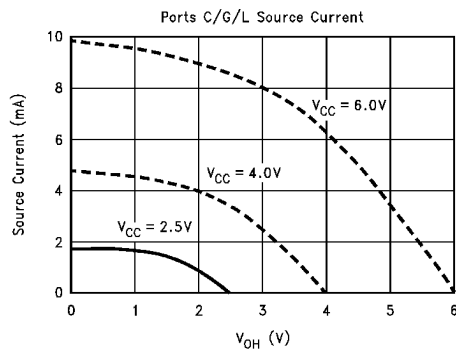
FIGURE 4. MICROWIRE/PLUS Timing

## Typical Performance Characteristics ( $-55^{\circ}C \leq T_A \leq +125^{\circ}C$ )

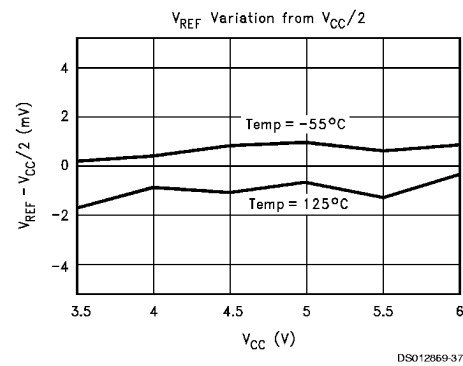
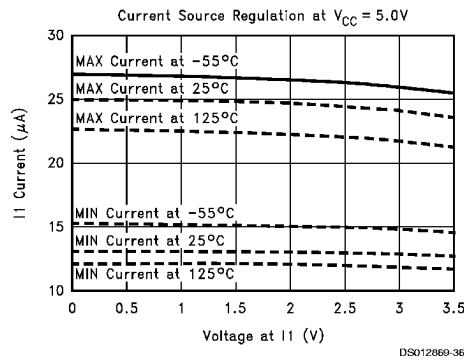
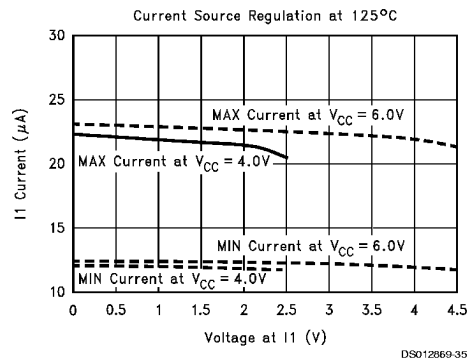
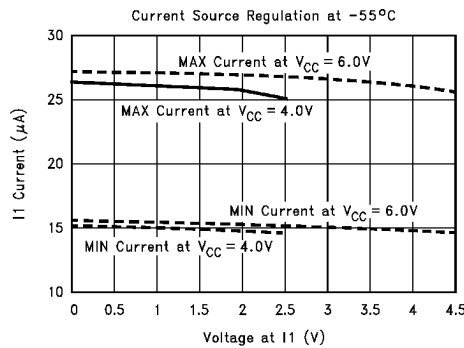




## Typical Performance Characteristics ( $-55^{\circ}\text{C} \leq T_A = +125^{\circ}\text{C}$ ) (Continued)



## Typical Performance Characteristics ( $-55^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$ ) (Continued)



## Pin Descriptions

$V_{CC}$  and GND are the power supply pins. All  $V_{CC}$  and GND pins must be connected.

CKI is the clock input. This can come from an R/C generated oscillator, or a crystal oscillator (in conjunction with CKO). See Oscillator Description section.

RESET is the master reset input. See Reset description section.

The device contains two bidirectional (one 8-bit, one 4-bit) I/O ports (G and L), where each individual bit may be independently configured as a weak pullup input, TRI-STATE (Hi-Z) input or push pull output under program control. Ports G- and L- feature Schmitt trigger inputs. Three data memory address locations are allocated for each of these I/O ports. Each I/O port has two associated 8-bit memory mapped registers, the CONFIGURATION register and the output DATA register. A memory mapped address is also reserved for the input pins of each I/O port. (See the memory map for the various addresses associated with the I/O ports.) Figure 5 shows the I/O port configurations. The DATA and CONFIGURATION registers allow for each port bit to be individually configured under software control as shown below:

**PORT L** is a 4-bit I/O port. All L-pins have Schmitt triggers on the inputs.

The Port L supports Multi-Input Wake Up on all four pins. The Port L has the following alternate features:

- L7 MIWU or external interrupt
- L6 MIWU or external interrupt
- L5 MIWU or external interrupt
- L4 MIWU or external interrupt

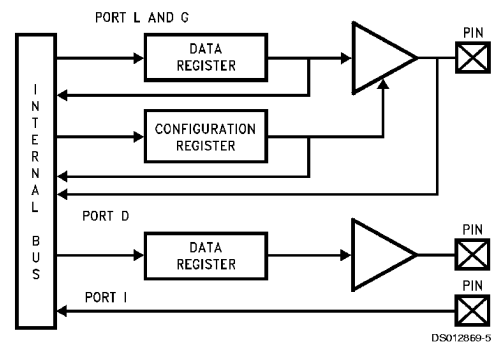


FIGURE 5. I/P Port Configurations

## Pin Descriptions (Continued)

| Configuration Register | Data Register | Port Set-Up                   |
|------------------------|---------------|-------------------------------|
| 0                      | 0             | Hi-Z Input (TRI-STATE Output) |
| 0                      | 1             | Input with Weak Pull-Up       |
| 1                      | 0             | Push-Pull Zero Output         |
| 1                      | 1             | Push-Pull One Output          |

### Please note:

The lower 4 L-bits read all ones (L0:L3). This is independent from the states of the associated bits in the L-port Data- and Configuration register. The lower 4 bits in the L-port Data- and Configuration register can be used as general purpose status indicators (flags).

**Port G** is an 8-bit port with 5 I/O pins (G0, G2–G5), an input pin (G6), and a dedicated output pin (G7). Pins G0 and G2–G6 all have Schmitt Triggers on their inputs. Pin G1 serves as the dedicated WDOUT WATCHDOG output, while pin G7 is either input or output depending on the oscillator mask option selected. With the crystal oscillator option selected, G7 serves as the dedicated output pin for the CKO clock output. With the single-pin R/C oscillator mask option selected, G7 serves as a general purpose input pin but is also used to bring the device out of HALT mode with a low to high transition on G7. There are two registers associated with the G Port, a data register and a configuration register. Therefore, each of the 5 I/O bits (G0, G2–G5) can be individually configured under software control.

Since G6 is an input only pin and G7 is the dedicated CKO clock output pin (crystal clock option) or general purpose input (R/C clock option), the associated bits in the data and configuration registers for G6 and G7 are used for special purpose functions as outlined below. Reading the G6 and G7 data bits will return zeros.

Note that the chip will be placed in the HALT mode by writing a "1" to bit 7 of the Port G Data Register. Similarly the chip will be placed in the IDLE mode by writing a "1" to bit 6 of the Port G Data Register.

Writing a "1" to bit 6 of the Port G Configuration Register enables the MICROWIRE/PLUS to operate with the alternate phase of the SK clock. The G7 configuration bit, if set high, enables the clock start up delay after HALT when the R/C clock configuration is used.

|    | Config Reg.  | Data Reg. |
|----|--------------|-----------|
| G7 | CLKDLY       | HALT      |
| G6 | Alternate SK | IDLE      |

Port G has the following alternate features:

- G6 SI (MICROWIRE Serial Data Input)
- G5 SK (MICROWIRE Serial Clock)
- G4 SO (MICROWIRE Serial Data Output)
- G3 T1A (Timer T1 I/O)
- G2 T1B (Timer T1 Capture Input)
- G0 INTR (External Interrupt Input)

Port G has the following dedicated functions:

- G7 CKO Oscillator dedicated output or general purpose input
- G1 WDOUT WATCHDOG and/or Clock Monitor dedicated output.

**Port I** is an eight-bit Hi-Z input port.

Port I0–I7 are used for the analog function block.

The Port I has the following alternate features:

- I7 C<sub>OUT</sub> (Comparator Output)
- I6 Analog CH6 (Comparator Positive Input 6)
- I5 Analog CH5 (Comparator Positive Input 5)
- I4 Analog CH4 (Comparator Positive Input 4)
- I3 Analog CH3 (Comparator Positive Input 3/Comparator Output)
- I2 Analog CH2 (Comparator Positive Input 2)
- I1 I<sub>SRC</sub> (Comparator Negative Input/Current Source Out)
- I0 Analog CH1 (Comparator Positive Input 1)

**Port D** is a 4-bit output port that is preset high when **RESET** goes low. The user can tie two or more D port outputs (except D2) together in order to get a higher drive.

## Functional Description

The architecture of the device is a modified Harvard architecture. With the Harvard architecture, the control store program memory (ROM) is separated from the data store memory (RAM). Both ROM and RAM have their own separate addressing space with separate address buses. The architecture, though based on the Harvard architecture, permits transfer of data from ROM to RAM.

### CPU REGISTERS

The CPU can do an 8-bit addition, subtraction, logical or shift operation in one instruction ( $t_c$ ) cycle time.

There are six CPU registers:

A is the 8-bit Accumulator Register

PC® is the 15-bit Program Counter Register

PU is the upper 7 bits of the program counter (PC)

PL is the lower 8 bits of the program counter (PC)

B is an 8-bit RAM address pointer, which can be optionally post auto incremented or decremented.

X is an 8-bit alternate RAM address pointer, which can be optionally post auto incremented or decremented.

SP is the 8-bit stack pointer, which points to the subroutine/interrupt stack (in RAM). The SP is initialized to RAM address 06F with reset.

All the CPU registers are memory mapped with the exception of the Accumulator (A) and the Program Counter (PC).

### PROGRAM MEMORY

The program memory consists of 16,384 bytes of OTP EPROM. These bytes may hold program instructions or constant data (data tables for the LAID instruction, jump vectors for the JID instruction, and interrupt vectors for the VIS instruction). The program memory is addressed by the 15-bit program counter (PC). All interrupts in the device vector to program memory location 0FF Hex.

The device can be configured to inhibit external reads of the program memory. This is done by programming the Security Byte.

### SECURITY FEATURE

The program memory array has an associate Security Byte that is located outside of the program address range. This byte can be addressed only from programming mode by a programmer tool.

Security is an optional feature and can only be asserted after the memory array has been programmed and verified. A secured part will read all 00(hex) by a programmer. The part

## Functional Description (Continued)

will fail Blank Check and will fail Verify operations. A Read operation will fill the programmer's memory with 00(hex). The Security Byte itself is always readable with value of 00(hex) if unsecure and FF(hex) if secure.

### DATA MEMORY

The data memory address space includes the on-chip RAM and data registers, the I/O registers (Configuration, Data and Pin), the control registers, the MICROWIRE/PLUS SIO shift register, and the various registers, and counters associated with the timers (with the exception of the IDLE timer). Data memory is addressed directly by the instruction or indirectly by the B, X, and SP pointers.

The data memory consists of 128 bytes of RAM. Sixteen bytes of RAM are mapped as "registers" at addresses 0F0 to 0FF Hex. These registers can be loaded immediately, and also decremented and tested with the DRSZ (decrement register and skip if zero) instruction. The memory pointer registers X, B and SP are memory mapped into this space at address locations 0FC to 0FF Hex respectively, with the other registers being available for general usage.

The instruction set permits any bit in memory to be set, reset or tested. All I/O and registers (except A and PC) are memory mapped; therefore, I/O bits and register bits can be directly and individually set, reset and tested. The accumulator (A) bits can also be directly and individually tested.

**Note:** RAM contents are undefined upon power-up.

### Reset

The **RESET** input when pulled low initializes the microcontroller. Initialization will occur whenever the **RESET** input is pulled low. Upon initialization, the data and configuration registers for ports L and G are cleared, resulting in these Ports being initialized to the TRI-STATE mode. Pin G1 of the G Port is an exception (as noted below) since pin G1 is dedicated as the WATCHDOG and/or Clock Monitor error output pin. Port D is set high. The PC, PSW, ICNTRL and CNTRL-control registers are cleared. The Comparator Select Register is cleared. The S register is initialized to zero. The Multi-Input Wakeup registers WKEN and WKEDG are cleared. Wakeup register WKPND is unknown. The stack pointer, SP, is initialized to 6F Hex.

The device comes out of reset with both the WATCHDOG logic and the Clock Monitor detector armed, with the WATCHDOG service window bits set and the Clock Monitor bit set. The WATCHDOG and Clock Monitor circuits are inhibited during reset. The WATCHDOG service window bits being initialized high default to the maximum WATCHDOG service window of 64k  $t_{CLOCK}$  clock cycles. The Clock Monitor bit being initialized high will cause a Clock Monitor error following reset if the clock has not reached the minimum specified frequency at the termination of reset. A Clock Monitor error will cause an active low error output on pin G1. This error output will continue until 16  $t_{CLOCK}$ -32  $t_{CLOCK}$  clock cycles following the clock frequency reaching the minimum specified value, at which time the G1 output will enter the TRI-STATE mode.

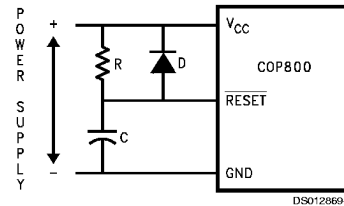
The external RC network shown in Figure 6 should be used to ensure that the **RESET** pin is held low until the power supply to the chip stabilizes.

#### WARNING:

When the device is held in reset for a long time it will consume high current (typically about 7 mA). This is not true for the equivalent ROM device (COP8ACC5).

## Oscillator Circuits

The chip can be driven by a clock input on the CKI input pin which can be between DC and 10 MHz. The CKO output clock is on pin G7 (crystal configuration). The CKI input frequency is divided down by 10 to produce the instruction cycle clock ( $t_{CLOCK}$ ).



$RC > 5 \times \text{POWER SUPPLY RISE TIME}$

**FIGURE 6. Recommended Reset Circuit**

Figure 7 shows the Crystal and R/C Oscillator diagrams.

### CRYSTAL OSCILLATOR

CKI and CKO can be connected to make a closed loop crystal (or resonator) controlled oscillator.

Table 1 shows the component values required for various standard crystal values.

**TABLE 1. Crystal Oscillator Configuration,  $T_A = 25^\circ\text{C}$**

| R1<br>(k $\Omega$ ) | R2<br>(M $\Omega$ ) | C1<br>(pF) | C2<br>(pF) | CKI Freq<br>(MHz) | Conditions    |
|---------------------|---------------------|------------|------------|-------------------|---------------|
| 0                   | 1                   | 30         | 30-36      | 10                | $V_{CC} = 5V$ |
| 0                   | 1                   | 30         | 30-36      | 4                 | $V_{CC} = 5V$ |
| 0                   | 1                   | 200        | 100-150    | 0.455             | $V_{CC} = 5V$ |

### R/C OSCILLATOR

By selecting CKI as a single pin oscillator input, a single pin R/C oscillator circuit can be connected to it. CKO is available as a general purpose input, and/or HALT restart input.

**Note:** Use of the R/C oscillator option will result in higher electromagnetic emissions.

Table 2 shows the variation in the oscillator frequencies as functions of the component (R and C) values.

**TABLE 2. RC Oscillator Configuration,  $T_A = 25^\circ\text{C}$**

| R<br>(k $\Omega$ ) | C<br>(pF) | CKI Freq<br>(MHz) | Instr. Cycle<br>( $\mu\text{s}$ ) | Conditions    |
|--------------------|-----------|-------------------|-----------------------------------|---------------|
| 3.3                | 82        | 2.2 to 2.7        | 3.7 to 4.6                        | $V_{CC} = 5V$ |
| 5.6                | 100       | 1.1 to 1.3        | 7.4 to 9.0                        | $V_{CC} = 5V$ |
| 6.8                | 100       | 0.9 to 1.1        | 8.8 to 10.8                       | $V_{CC} = 5V$ |

**Note 18:**  $3k \leq R \leq 200k$

**Note 19:**  $50 \text{ pF} \leq C \leq 200 \text{ pF}$

## Oscillator Circuits (Continued)

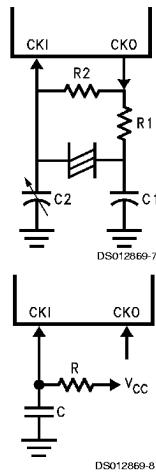


FIGURE 7. Crystal and R/C Oscillator Diagrams

## Control Registers

### CNTRL Register (Address X'00EE)

|       |      |      |      |      |      |     |       |
|-------|------|------|------|------|------|-----|-------|
| T1C3  | T1C2 | T1C1 | T1C0 | MSEL | IEDG | SL1 | SL0   |
| Bit 7 |      |      |      |      |      |     | Bit 0 |

The Timer1 (T1) and MICROWIRE/PLUS control register contains the following bits:

|           |                                                                                                         |
|-----------|---------------------------------------------------------------------------------------------------------|
| T1C3      | Timer T1 mode control bit                                                                               |
| T1C2      | Timer T1 mode control bit                                                                               |
| T1C1      | Timer T1 mode control bit                                                                               |
| T1C0      | Timer T1 Start/Stop control in timer modes 1 and 2, T1 Underflow Interrupt Pending Flag in timer mode 3 |
| MSEL      | Selects G5 and G4 as MICROWIRE/PLUS signals SK and SO respectively                                      |
| IEDG      | External interrupt edge polarity select (0 = Rising edge, 1 = Falling edge)                             |
| SL1 & SL0 | Select the MICROWIRE/PLUS clock divide by (00 = 2, 01 = 4, 1x = 8)                                      |

### PSW Register (Address X'00EF)

|       |   |        |       |       |      |      |       |
|-------|---|--------|-------|-------|------|------|-------|
| HC    | C | T1PNDA | T1ENA | EXPND | BUSY | EXEN | GIE   |
| Bit 7 |   |        |       |       |      |      | Bit 0 |

The PSW register contains the following select bits:

|        |                                                                                                               |
|--------|---------------------------------------------------------------------------------------------------------------|
| HC     | Half Carry Flag                                                                                               |
| C      | Carry Flag                                                                                                    |
| T1PNDA | Timer T1 Interrupt Pending Flag (Autoreload RA in mode 1, T1 Underflow in Mode 2, T1A capture edge in mode 3) |
| T1ENA  | Timer T1 Interrupt Enable for Timer Underflow or T1A Input capture edge                                       |
| EXPND  | External interrupt pending                                                                                    |
| BUSY   | MICROWIRE/PLUS busy shifting flag                                                                             |
| EXEN   | Enable external interrupt                                                                                     |

GIE Global interrupt enable (enables interrupts)

The Half-Carry flag is also affected by all the instructions that affect the Carry flag. The SC (Set Carry) and R/C (Reset Carry) instructions will respectively set or clear both the carry flags. In addition to the SC and R/C instructions, ADC, SUBC, RRC and RLC instructions affect the Carry and Half Carry flags.

### ICNTRL Register (Address X'00E8)

|          |      |       |      |       |      |        |       |
|----------|------|-------|------|-------|------|--------|-------|
| Reserved | LPEN | T0PND | T0EN | μWPND | μWEN | T1PNDB | T1ENB |
| Bit 7    |      |       |      |       |      |        | Bit 0 |

The ICNTRL register contains the following bits:

|          |                                                        |
|----------|--------------------------------------------------------|
| Reserved | This bit is reserved and should be zero.               |
| LPEN     | L Port Interrupt Enable (Multi-Input Wakeup/Interrupt) |
| T0PND    | Timer T0 Interrupt pending                             |
| T0EN     | Timer T0 Interrupt Enable (Bit 12 toggle)              |
| μWPND    | MICROWIRE/PLUS interrupt pending                       |
| μWEN     | Enable MICROWIRE/PLUS interrupt                        |
| T1PNDB   | Timer T1 Interrupt Pending Flag for T1B capture edge   |
| T1ENB    | Timer T1 Interrupt Enable for T1B Input capture edge   |

## Timers

The device contains a very versatile set of timers (T0 and T1). All timers and associated autoreload/capture registers power up containing random data.

### TIMER T0 (IDLE TIMER)

The device supports applications that require maintaining real time and low power with the IDLE mode. This IDLE mode support is furnished by the IDLE timer T0, which is a 16-bit timer. The Timer T0 runs continuously at the fixed rate of the instruction cycle clock,  $t_{\text{C}}$ . The user cannot read or write to the IDLE Timer T0, which is a count down timer.

The Timer T0 supports the following functions:

- Exit out of the Idle Mode (See Idle Mode description)
- WATCHDOG logic (See WATCHDOG description)
- Start up delay out of the HALT mode

is a functional block diagram showing the structure of the IDLE Timer and its associated interrupt logic.

Bits 11 through 15 of the ITMR register can be selected for triggering the IDLE Timer interrupt. Each time the selected bit underflows (every 4k, 8k, 16k, 32k or 64k instruction cycles), the IDLE Timer interrupt pending bit T0PND is set, thus generating an interrupt (if enabled), and bit 6 of the Port G data register is reset, thus causing an exit from the IDLE mode if the device is in that mode.

In order for an interrupt to be generated, the IDLE Timer interrupt enable bit T0EN must be set, and the GIE (Global Interrupt Enable) bit must also be set. The T0PND flag and T0EN bit are bits 5 and 4 of the ICNTRL register, respectively. The interrupt can be used for any purpose. Typically, it is used to perform a task upon exit from the IDLE mode. For more information on the IDLE mode, refer to the Power Save Modes section.

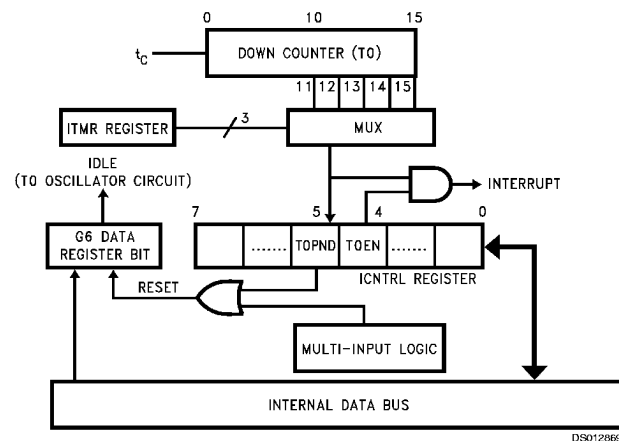
The Idle Timer period is selected by bits 0–2 of the ITMR register Bits 3–7 of the ITMR Register are reserved and should not be used as software flags.

### ITMR Register (Address X'0xCF)

| ITSEL2 | ITSEL1 | ITSEL0 | Idle Timer Period<br>(Instruction Cycles) |
|--------|--------|--------|-------------------------------------------|
| 0      | 1      | 1      | 32,768                                    |
| 1      | X      | X      | 65,536                                    |

| ITSEL2 | ITSEL1 | ITSEL0 | Idle Timer Period<br>(Instruction Cycles) |
|--------|--------|--------|-------------------------------------------|
| 0      | 0      | 0      | 4,096                                     |
| 0      | 0      | 1      | 8,192                                     |
| 0      | 1      | 0      | 16,384                                    |

Any time the IDLE Timer period is changed there is the possibility of generating a spurious IDLE Timer interrupt by setting the T0PND bit. The user is advised to disable IDLE Timer interrupts prior to changing the value of the ITSEL bits of the ITMR Register and then clear the T0PND bit before attempting to synchronize operation to the IDLE Timer.



**FIGURE 8. Functional Block Diagram for Idle Timer T0**

The device has a powerful timer/counter block. The timer consists of a 16-bit timer, T1, and two supporting 16-bit autoreload/capture registers, R1A and R1B. The timer block has two pins associated with it, T1A and T1B. The pin T1A supports I/O required by the timer block, while the pin T1B is an input to the timer block. The powerful and flexible timer block allows the device to easily perform all timer functions with minimal software overhead. The timer block has three operating modes: Processor Independent PWM mode, External Event Counter mode, and Input Capture mode.

the contents of supporting registers, R1A and R1B. The very first underflow of the timer causes the timer to reload from the register R1A. Subsequent underflows cause the timer to be reloaded from the registers alternately beginning with the register R1B.

shows a block diagram of the timer in PWM mode.

Underflows from the timer are alternately latched into two pending flags, T1PNDA and T1PNDB. The user must reset these pending flags under software control. Two control enable flags, T1ENA and T1ENB, allow the interrupts from the timer underflow to be enabled or disabled. Setting the timer enable flag T1ENA will cause an interrupt when a timer underflow causes the R1A register to be reloaded into the timer. Setting the timer enable flag T1ENB will cause an interrupt when a timer underflow causes the R1B register to be reloaded into the timer. Resetting the timer enable flags will disable the associated interrupts.

Either or both of the timer underflow interrupts may be enabled. This gives the user the flexibility of interrupting once

As the name suggests, this mode allows the device to generate a PWM signal with very minimal user intervention. The user only has to define the parameters of the PWM signal (ON time and OFF time). Once begun, the timer block will continuously generate the PWM signal completely independent of the microcontroller. The user software services the timer block only when the PWM parameters require updating.

[www.national.com](http://www.national.com)

## Timers (Continued)

per PWM period on either the rising or falling edge of the PWM output. Alternatively, the user may choose to interrupt on both edges of the PWM output.

### Mode 2. External Event Counter Mode

This mode is quite similar to the processor independent PWM mode previously described. The main difference is that the timer, T1, is clocked by the input signal from the T1A pin. The T1 timer control bits, T1C3, T1C2 and T1C1 allow the timer to be clocked either on a positive or negative edge from the T1A pin. Underflows from the timer are latched into the T1PND A pending flag. Setting the T1ENA control flag will cause an interrupt when the timer underflows.

In this mode the input pin T1B can be used as an independent positive edge sensitive interrupt input if the T1ENB control flag is set. The occurrence of a positive edge on the T1B input pin is latched into the T1PND B flag.

Figure 10 shows a block diagram of the timer in External Event Counter mode.

**Note:** The PWM output is not available in this mode since the T1A pin is being used as the counter input clock.

### Mode 3. Input Capture Mode

The device can precisely measure external frequencies or time external events by placing the timer block, T1, in the input capture mode.

In this mode, the timer T1 is constantly running at the fixed  $t_{\text{C}}$  rate. The two registers, R1A and R1B, act as capture registers. Each register acts in conjunction with a pin. The register R1A acts in conjunction with the T1A pin and the register R1B acts in conjunction with the T1B pin.

The timer value gets copied over into the register when a trigger event occurs on its corresponding pin. Control bits, T1C3, T1C2 and T1C1, allow the trigger events to be specified either as a positive or a negative edge. The trigger condition for each input pin can be specified independently.

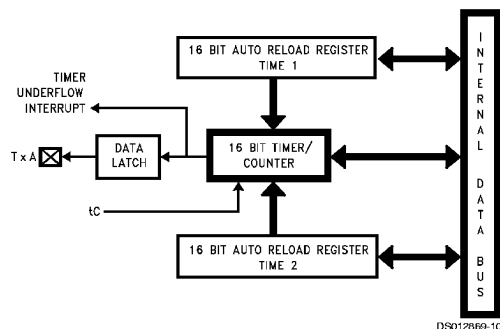


FIGURE 9. Timer in PWM Mode

The trigger conditions can also be programmed to generate interrupts. The occurrence of the specified trigger condition on the T1A and T1B pins will be respectively latched into the pending flags, T1PND A and T1PND B. The control flag T1ENA allows the interrupt on T1A to be either enabled or disabled. Setting the T1ENA flag enables interrupts to be generated when the selected trigger condition occurs on the T1A pin. Similarly, the flag T1ENB controls the interrupts from the T1B pin.

Underflows from the timer can also be programmed to generate interrupts. Underflows are latched into the timer T1C0 pending flag (the T1C0 control bit serves as the timer underflow interrupt pending flag in the Input Capture mode). Consequently, the T1C0 control bit should be reset when entering the Input Capture mode. The timer underflow interrupt is enabled with the T1ENA control flag. When a T1A interrupt occurs in the Input Capture mode, the user must check both the T1PND A and T1C0 pending flags in order to determine whether a T1A input capture or a timer underflow (or both) caused the interrupt.

Figure 11 shows a block diagram of the timer in Input Capture mode.

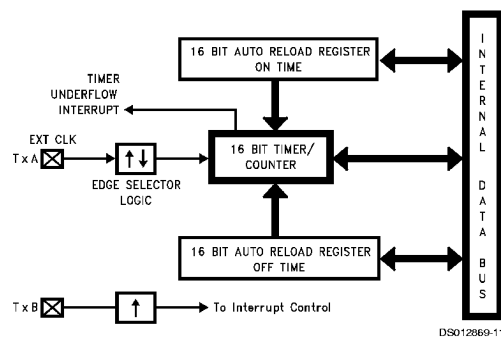


FIGURE 10. Timer in External Event Counter Mode

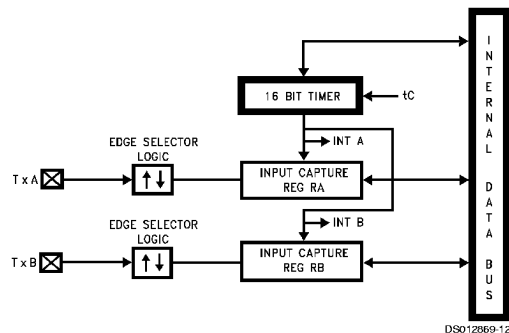


FIGURE 11. Timer in Input Capture Mode

## Timers (Continued)

### TIMER CONTROL FLAGS

The control bits and their functions are summarized below.

|      |                                                                                                                                                                                                 |
|------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| T1C3 | Timer mode control                                                                                                                                                                              |
| T1C2 | Timer mode control                                                                                                                                                                              |
| T1C1 | Timer mode control                                                                                                                                                                              |
| T1C0 | Timer Start/Stop control in Modes 1 and 2 (Processor Independent PWM and External Event Counter), where 1 = Start, 0 = Stop<br>Timer Underflow Interrupt Pending Flag in Mode 3 (Input Capture) |

|         |                                                                                            |
|---------|--------------------------------------------------------------------------------------------|
| T1PND A | Timer Interrupt Pending Flag                                                               |
| T1ENA   | Timer Interrupt Enable Flag<br>1 = Timer Interrupt Enabled<br>0 = Timer Interrupt Disabled |
| T1PND B | Timer Interrupt Pending Flag                                                               |
| T1ENB   | Timer Interrupt Enable Flag<br>1 = Timer Interrupt Enabled<br>0 = Timer Interrupt Disabled |

The timer mode control bits (T1C3, T1C2 and T1C1) are detailed below:

| Mode | T1C3 | T1C2 | T1C1 | Description                              | Interrupt A Source               | Interrupt B Source | Timer Counts On |
|------|------|------|------|------------------------------------------|----------------------------------|--------------------|-----------------|
| 1    | 1    | 0    | 1    | PWM: T1A Toggle                          | Autoreload RA                    | Autoreload RB      | $t_c$           |
|      | 1    | 0    | 0    | PWM: No T1A Toggle                       | Autoreload RA                    | Autoreload RB      | $t_c$           |
| 2    | 0    | 0    | 0    | External Event Counter                   | Timer Underflow                  | Pos. T1B Edge      | Pos. T1A Edge   |
|      | 0    | 0    | 1    | External Event Counter                   | Timer Underflow                  | Pos. T1B Edge      | Pos. T1A Edge   |
| 3    | 0    | 1    | 0    | Captures: T1A Pos. Edge<br>T1B Pos. Edge | Pos. T1A Edge or Timer Underflow | Pos. T1B Edge      | $t_c$           |
|      | 1    | 1    | 0    | Captures: T1A Pos. Edge<br>T1B Neg. Edge | Pos. T1A Edge or Timer Underflow | Neg. T1B Edge      | $t_c$           |
|      | 0    | 1    | 1    | Captures: T1A Neg. Edge<br>T1B Neg. Edge | Neg. T1A Edge or Timer Underflow | Neg. T1B Edge      | $t_c$           |
|      | 1    | 1    | 1    | Captures: T1A Neg. Edge<br>T1B Neg. Edge | Neg. T1A Edge or Timer Underflow | Neg. T1B Edge      | $t_c$           |

### HIGH SPEED CAPTURE TIMER

The device provides a 16-bit high-speed capture timer. The timer consists of a 16-bit up-counter that is clocked with the device clock input frequency (CKI) and an 8-bit control register. The 16-bit counter is mapped as two read/write 8-bit registers. This timer is specifically designed to be used in conjunction with the Analog Function Block (comparator, analog multiplexer, constant current source) to implement a low-cost, high-resolution, single-slope A/D.

The timer is automatically stopped in the event of a capture to allow the software to read the timer value. Coming out of reset the counter is disabled (stopped) and reads all "0".

Setting the Capture Timer Run bit CAPRUN bit in the Capture Control Register (CAPCNTL) will start the counter. The counter will count up until a capture event (negative edge) is received. Upon a capture the counter will be stopped, the Capture Pending bit (CAPPND) is set, and the CAPRUN bit is automatically reset. If capture interrupts are enabled (CAPIEN=1), the capture event will generate an interrupt. Setting the CAPRUN bit again by software will start a new counting cycle. If the Capture Mode bit is reset (CAPMOD=0) the capture timer will be automatically initialized to all "0" with each setting of the CAPRUN bit. If CAPMOD=1

the timer will not be cleared when setting the CAPRUN bit, thus allowing the user's software to pre-load the timer registers with any desired value. This mode can be used in conjunction with the timer's overflow to implement for example a programmable delay counter.

"CAPTURE MODE" is only active when the CAPRUN bit is set, i.e. any capture events received while the timer is stopped (CAPRUN=0) will be ignored and will not cause the CAPPND bit to be set. The capture counter can also be stopped (frozen) by the user's software resetting the CAPRUN bit.

If the user program tries to set the CAPRUN bit at the same time that the hardware gets a capture event and tries to reset the CAPRUN bit, the hardware will have precedence.

Should the counter overflow before a capture condition occurs, the Capture Overflow bit (CAPOVL) bit in the CAPCNTL register will be set. If Capture Interrupts are enabled (CAPIEN=1) an overflow will generate an interrupt. The user software should reset this bit before the next overflow occurs, otherwise subsequent overflow conditions cannot be detected.



## Timers (Continued)

Capture Overflow interrupt and Capture Pending interrupt share the same interrupt vector.

### CAPCNTL Register (Address (X'CE))

| Reserved | CAPMOD | CAPRUN | CAPOVL | CAPPND | CAPIEN |
|----------|--------|--------|--------|--------|--------|
| Bit 7-5  | Bit 4  |        |        |        | Bit 0  |

The CAPCNTL register contains the following bits:

Reserved These bits are reserved and must be zero.

**CAPMOD** Reset Time.

0: reset timer to "0" when CAPRUN bit gets set  
1: DO NOT reset timer to "0" when CAPRUN bit gets set.

**CAPRUN** Capture Timer Run. Setting this bit to one will start the capture timer. This bit gets automatically reset to "0" when a capture events occurs. Writing a "0" by software will also reset the bit and stop the timer.

**CAPOVL** Capture Timer Overflow. Gets set to "1" upon timer overflow. Has to be reset by user's software. If CAPIEN = 1 an interrupt is generated.

**CAPPND** Capture pending.

Gets automatically set when a capture event occurs. If CAPIEN = 1 an interrupt is generated. Has to be reset by the user's software.

**CAPIEN** Capture Interrupt enable,  
1 = enable interrupts, 0 = disable interrupts

## Power Save Modes

The device offers the user two power save modes of operation: HALT and IDLE. In the HALT mode, all microcontroller activities are stopped. In the IDLE mode, the on-board oscillator circuitry and timer T0 are active but all other microcontroller activities are stopped. In either mode, all on-board RAM, registers, I/O states, and timers (with the exception of T0) are unaltered.

### HALT MODE

The device can be placed in the HALT mode by writing a "1" to the HALT flag (G7 data bit). All microcontroller activities, including the clock and timers, are stopped. The WATCHDOG logic on the device is disabled during the HALT mode. However, the clock monitor circuitry, if enabled, remains active and will cause the WATCHDOG output pin (WDOUT) to go low. If the HALT mode is used and the user does not want to activate the WDOUT pin, the Clock Monitor should be disabled after the device comes out of reset (resetting the Clock Monitor control bit with the first write to the WDSVR register). In the HALT mode, the power requirements of the device are minimal and the applied voltage ( $V_{CC}$ ) may be decreased to  $V_r$  ( $V_r = 2.0V$ ) without altering the state of the machine.

The device supports three different ways of exiting the HALT mode. The first method of exiting the HALT mode is with the Multi-Input Wakeup feature on the Port L.

The second method is with a low to high transition on the CKO (G7) pin. This method precludes the use of the crystal clock configuration (since CKO becomes a dedicated output), and so may only be used with an RC clock configuration. The third method of exiting the HALT mode is by pulling the RESET pin low.

Since a crystal or ceramic resonator may be selected as the oscillator, the Wakeup signal is not allowed to start the chip running immediately since crystal oscillators and ceramic resonators have a delayed start up time to reach full amplitude and frequency stability. The IDLE timer is used to generate a fixed delay to ensure that the oscillator has indeed stabilized before allowing instruction execution. In this case, upon detecting a valid Wakeup signal, only the oscillator circuitry is enabled. The IDLE timer is loaded with a value of 256 and is clocked with the  $t_C$  instruction cycle clock. The  $t_C$  clock is derived by dividing the oscillator clock down by a factor of 10. The Schmitt trigger following the CKI inverter on the chip ensures that the IDLE timer is clocked only when the oscillator has a sufficiently large amplitude to meet the Schmitt trigger specifications. This Schmitt trigger is not part of the oscillator closed loop. The startup timeout from the IDLE timer enables the clock signals to be routed to the rest of the chip.

If an RC clock option is being used, the fixed delay is introduced optionally. A control bit, CLKDLY, mapped as configuration bit G7, controls whether the delay is to be introduced or not. The delay is included if CLKDLY is set, and excluded if CLKDLY is reset. The CLKDLY bit is cleared on reset.

The device has two mask options associated with the HALT mode. The first mask option enables the HALT mode feature, while the second mask option disables the HALT mode. With the HALT mode enable mask option, the device will enter and exit the HALT mode as described above. With the HALT disable mask option, the device cannot be placed in the HALT mode (writing a "1" to the HALT flag will have no effect, the HALT flag will remain "0").

### IDLE MODE

In the IDLE mode, program execution stops and power consumption is reduced to a very low level as with the HALT mode. However, the on-board oscillator, IDLE Timer (Timer T0), and Clock Monitor continue to operate, allowing real time to be maintained. The device remains idle for a selected amount of time up to 65,536 instruction cycles, or 65.536 milliseconds with a 1 MHz instruction clock frequency, and then automatically exits the IDLE mode and returns to normal program execution.

The device is placed in the IDLE mode under software control by setting the IDLE bit (bit 6 of the Port G data register).

The IDLE timer window is selectable from one of five values, 4k, 8k, 16k, 32k or 64k instruction cycles. Selection of this value is made through the ITMR register.

The IDLE mode uses the on-chip IDLE Timer (Timer T0) to keep track of elapsed time in the IDLE state. The IDLE timer runs continuously at the instruction clock rate, whether or not the device is in the IDLE mode. Each time the bit of the timer associated with the selected window toggles, the T0PND bit is set, an interrupt is generated (if enabled), and the device exits the IDLE mode if in that mode. If the IDLE timer interrupt is enabled, the interrupt is serviced before execution of the main program resumes. (However, the instruction which was started as the part entered the IDLE mode is completed before the interrupt is serviced. This instruction should be a NOP which should follow the enter IDLE instruction.) The user must reset the IDLE timer pending flag (T0PND) before entering the IDLE mode.

As with the HALT mode, this device can also be returned to normal operation with a reset, or with a Multi-Input Wakeup input. Upon reset the ITMR register is cleared and the ITMR register selects the 4,096 instruction cycle tap of the Idle Timer.

## Power Save Modes (Continued)

The IDLE timer cannot be started or stopped under software control, and it is not memory mapped, so it cannot be read or written by the software. Its state upon Reset is unknown. Therefore, if the device is put into the IDLE mode at an arbitrary time, it will stay in the IDLE mode for somewhere between 1 and the selected number of instruction cycles.

In order to precisely time the duration of the IDLE state, entry into the IDLE mode must be synchronized to the state of the IDLE Timer. The best way to do this is to use the IDLE Timer interrupt, which occurs on every underflow of the bit of the IDLE Timer which is associated with the selected window. Another method is to poll the state of the IDLE Timer pending bit TOPND, which is set on the same occurrence. The Idle Timer interrupt is enabled by setting bit T0EN in the ICNTRL register.

Any time the IDLE Timer window length is changed there is the possibility of generating a spurious IDLE Timer interrupt by setting the TOPND bit. The user is advised to disable IDLE Timer interrupts prior to changing the value of the ITSEL bits of the ITMR Register and then clear the TOPND bit before attempting to synchronize operation to the IDLE Timer.

**Note:** As with the HALT mode, it is necessary to program two NOP's to allow clock resynchronization upon return from the IDLE mode. The NOP's are placed either at the beginning of the IDLE timer interrupt routine or immediately following the "enter IDLE mode" instruction.

For more information on the IDLE Timer and its associated interrupt, see the description in the Timers section.

## Multi-Input Wakeup

The Multi-Input Wakeup feature is used to return (wakeup) the device from either the HALT or IDLE modes. Alternately Multi-Input Wakeup/Interrupt feature may also be used to generate up to 4 edge selectable external interrupts.

Figure 12 shows the Multi-Input Wakeup logic.

The Multi-Input Wakeup feature utilizes the L Port. The user selects which particular L port bit (or combination of L Port bits) will cause the device to exit the HALT or IDLE modes. The selection is done through the register WKEN. The register WKEN is an 8-bit read/write register, which contains a control bit for every L port bit. Setting a particular WKEN bit enables a Wakeup from the associated L port pin.

The user can select whether the trigger condition on the selected L Port pin is going to be either a positive edge (low to high transition) or a negative edge (high to low transition). This selection is made via the register WKEDG, which is an 8-bit control register with a bit assigned to each L Port pin. Setting the control bit will select the trigger condition to be a negative edge on that particular L Port pin. Resetting the bit selects the trigger condition to be a positive edge. Changing an edge select entails several steps in order to avoid a Wakeup condition as a result of the edge change. First, the associated WKEN bit should be reset, followed by the edge select change in WKEDG. Next, the associated WKPND bit should be cleared, followed by the associated WKEN bit being re-enabled.

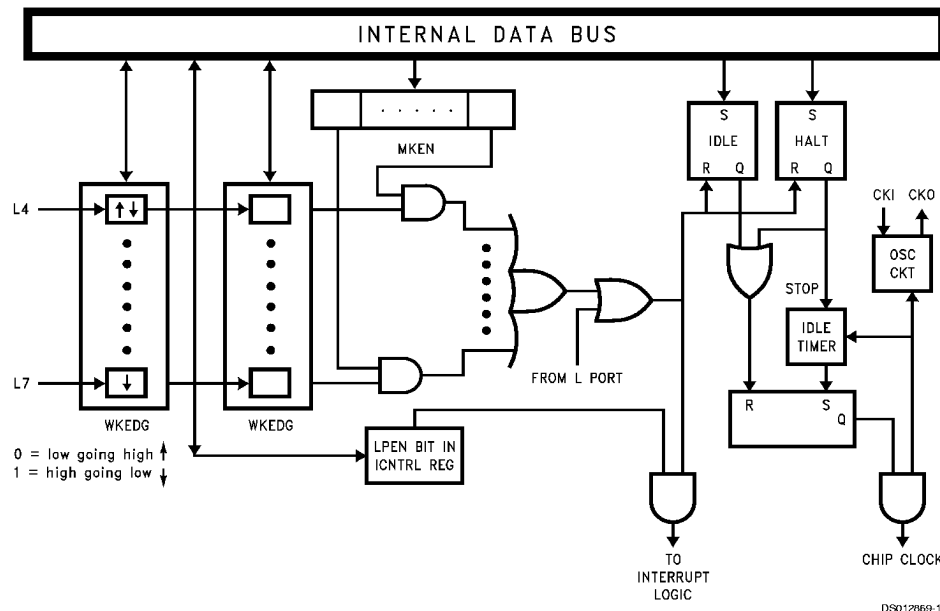


FIGURE 12. Multi-Input Wake Up Logic

## Multi-Input Wakeup (Continued)

An example may serve to clarify this procedure. Suppose we wish to change the edge select from positive (low going high) to negative (high going low) for L Port bit 5, where bit 5 has previously been enabled for an input interrupt. The program would be as follows:

```
RBIT 5, WKEN ; Disable MIWU
SBIT 5, WKEDG ; Change edge polarity
RBIT 5, WKPND ; Reset pending flag
SBIT 5, WKEN ; Enable MIWU
```

If the L port bits have been used as outputs and then changed to inputs with Multi-Input Wakeup/Interrupt, a safety procedure should also be followed to avoid wakeup conditions. After the selected L port bits have been changed from output to input but before the associated WKEN bits are enabled, the associated edge select bits in WKEDG should be set or reset for the desired edge selects, followed by the associated WKPND bits being cleared.

This same procedure should be used following reset, since the L port inputs are left floating as a result of reset.

The occurrence of the selected trigger condition for Multi-Input Wakeup is latched into a pending register called WKPND. The respective bits of the WKPND register will be set on the occurrence of the selected trigger edge on the corresponding Port L pin. The user has the responsibility of clearing these pending flags. Since WKPND is a pending register for the occurrence of selected wakeup conditions, the device will not enter the HALT mode if any Wakeup bit is both enabled and pending. Consequently, the user must clear the pending flags before attempting to enter the HALT mode.

WKEN, WKPND and WKEDG are all read/write registers, and are cleared at reset.

### PORT L INTERRUPTS

Port L provides the user with an additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (Global Interrupt Enable) bit enables the interrupt function.

A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE

modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation. (See HALT MODE for clock option wakeup information.)

## Analog Function Block

This device contains an analog function block with the intent to provide a function which allows for single slope, low cost, A/D conversion of up to 6 channels.

### CMPSL REGISTER (ADDRESS X'00B7)

| CMPT2B | CMPISEL2 | CMPISEL1 | CMPISEL0 | CMPOE | CSEN | CM PEN | CM PNEG |
|--------|----------|----------|----------|-------|------|--------|---------|
| Bit 7  |          |          |          |       |      |        | Bit 0   |

The CMPSL register contains the following bits:

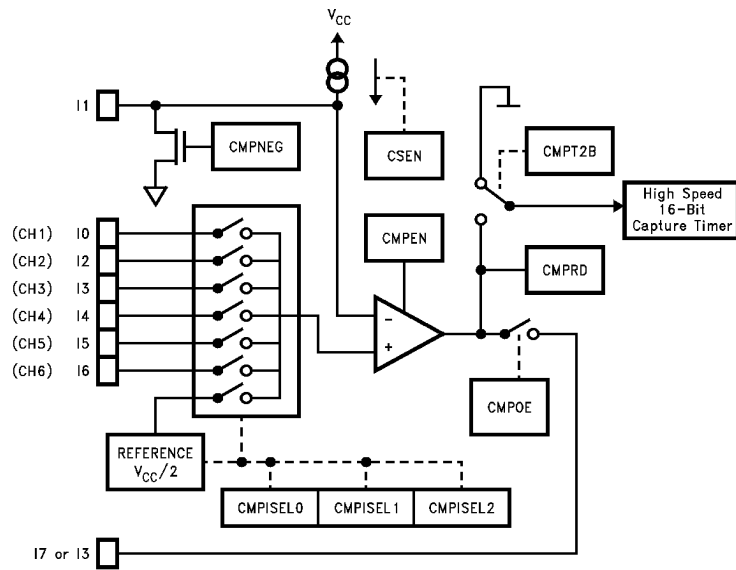
|              |                                                                                                                                                                                                                                                                                                                           |
|--------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| CMPT2B       | Selects the "High Speed 16-bit Capture Timer" input to be driven directly by the comparator output. If the comparator is disabled (CMPEN=0), this function is disabled, i.e. the Capture Timer input is connected to GND.                                                                                                 |
| CMPISEL0/1/2 | Will select one of seven possible sources (I0/I2/I3/I4/I5/I6/internal reference) as a positive input to the comparator (see Table 4 for more information)                                                                                                                                                                 |
| CMPOE        | Enables the comparator output to either pin I3 or pin I7 ("1"=enable) depending on the value of CMPISEL0/1/2.                                                                                                                                                                                                             |
| CSEN         | Enables the internal constant current source. This current source provides a nominal 20 $\mu$ A constant current at the I1 pin. This current can be used to ensure a linear charging rate on an external capacitor. This bit has no affect and the current source is disabled if the comparator is not enabled (CMPEN=0). |
| CM PEN       | Enable the comparator ("1" = enable)                                                                                                                                                                                                                                                                                      |
| CM PNEG      | Will drive I1 to a low level. This bit can be used to discharge an external capacitor. This bit is disabled if the comparator is not enabled (CMPEN=0).                                                                                                                                                                   |

The Comparator Select Register is cleared on RESET (the comparator is disabled). To save power the program should also disable the comparator before the  $\mu$ C enters the HALT/IDLE modes. Disabling the comparator will turn off the constant current source and the  $V_{CC}/2$  reference, disconnect the comparator output from the Capture Timer input and pin I3/I7 and remove the low on I1 caused by CMPNEG.

It is often useful for the user's program to read the result of a comparator operation. Since I1 is always selected to be COMPIN—when the comparator is enabled (CMPEN=1), the comparator output can be read internally by reading bit 1 (CMPRD) of register PORTI (RAM address 0xD7).

The following table lists the comparator inputs and outputs versus the value of the CMPISEL0/1/2 bits. The output will only be driven if the CMPOE bit is set to 1.

### Analog Function Block (Continued)



**FIGURE 13. Analog Function Block**

**TABLE 4. Comparator Input Selection**

| Control Bit |         |          | Comparator Input Source |                         | Comparator Output |
|-------------|---------|----------|-------------------------|-------------------------|-------------------|
| CMPISL2     | CMPISL1 | CMPISEL0 | Neg. Input              | Pos. Input              |                   |
| 0           | 0       | 0        | I1                      | I2 CH2                  | I3                |
| 0           | 0       | 1        | I1                      | I2 CH2                  | I7                |
| 0           | 1       | 0        | I1                      | I3 CH3                  | I7                |
| 0           | 1       | 1        | I1                      | I0 CH1                  | I7                |
| 1           | 0       | 0        | I1                      | I4 CH4                  | I7                |
| 1           | 0       | 1        | I1                      | I5 CH5                  | I7                |
| 1           | 1       | 0        | I1                      | I6 CH6                  | I7                |
| 1           | 1       | 1        | I1                      | V <sub>CC</sub> /2 Ref. | I7                |

## Reset

The state of the Analog Block immediately after RESET is as follows:

1. The CMPSL Register is set to all zeros
2. The Comparator is disabled
3. The Constant Current Source is disabled
4. CMPNEG is turned off
5. The Port I inputs are electrically isolated from the comparator
6. The Capture Timer input is connected to GND
7. CMPISEL0–CMPISEL2 are set to zero
8. All Port I inputs are selected to the default digital input mode

The comparator outputs have the same specification as Ports L and G except that the rise and fall times are symmetrical.

## Interrupts

## INTRODUCTION

Each device supports eight vectored interrupts. Interrupt sources include Timer 0, Timer 1, Timer 2, Timer 3, Port L Wakeup, Software Trap, MICROWIRE/PLUS, and External Input.

All interrupts force a branch to location 00FF Hex in program memory. The VIS instruction may be used to vector to the appropriate service routine from location 00FF Hex.

The Software trap has the highest priority while the default VIS has the lowest priority.

Figure 14 shows the Interrupt Block Diagram.

**FIGURE 14. Interrupt Block Diagram**

DS012869-15

Upon Reset, all pending bits, individual enable bits, and the GIE bit are reset to zero. Thus, a maskable interrupt condition cannot trigger an interrupt until the program enables it by setting both the GIE bit and the individual enable bit. When enabling an interrupt, the user should consider whether or not a previously activated (set) pending bit should be acknowledged. If, at the time an interrupt is enabled, any previous occurrences of the interrupt should be ignored, the as-

If the user wishes to allow nested interrupts, the interrupts service routine may set the GIE bit to 1 by writing to the PSW register, and thus allow other maskable interrupts to interrupt the current service routine. If nested interrupts are allowed, caution must be exercised. The user must write the program in such a way as to prevent stack overflow, loss of saved context information, and other unwanted conditions.

## Interrupts (Continued)

The interrupt service routine stored at location 00FF Hex should use the VIS instruction to determine the cause of the interrupt, and jump to the interrupt handling routine corresponding to the highest priority enabled and active interrupt. Alternately, the user may choose to poll all interrupt pending and enable bits to determine the source(s) of the interrupt. If more than one interrupt is active, the user's program must decide which interrupt to service.

Within a specific interrupt service routine, the associated pending bit should be cleared. This is typically done as early as possible in the service routine in order to avoid missing the next occurrence of the same type of interrupt event. Thus, if the same event occurs a second time, even while the first occurrence is still being serviced, the second occurrence will be serviced immediately upon return from the current interrupt routine.

An interrupt service routine typically ends with an RETI instruction. This instruction sets the GIE bit back to 1, pops the address stored on the stack, and restores that address to the program counter. Program execution then proceeds with the next instruction that would have been executed had there been no interrupt. If there are any valid interrupts pending, the highest-priority interrupt is serviced immediately upon return from the previous interrupt.

### VIS INSTRUCTION

The general interrupt service routine, which starts at address 00FF Hex, must be capable of handling all types of interrupts. The VIS instruction, together with an interrupt vector table, directs the device to the specific interrupt handling routine based on the cause of the interrupt.

VIS is a single-byte instruction, typically used at the very beginning of the general interrupt service routine at address 00FF Hex, or shortly after that point, just after the code used for context switching. The VIS instruction determines which enabled and pending interrupt has the highest priority, and causes an indirect jump to the address corresponding to that interrupt source. The jump addresses (vectors) for all possible interrupts sources are stored in a vector table.

The vector table may be as long as 32 bytes (maximum of 16 vectors) and resides at the top of the 256-byte block containing the VIS instruction. However, if the VIS instruction is at the very top of a 256-byte block (such as at 00FF Hex), the vector table resides at the top of the next 256-byte block. Thus, if the VIS instruction is located somewhere between 00FF and 01DF Hex (the usual case), the vector table is located between addresses 01E0 and 01FF Hex. If the VIS instruction is located between 01FF and 02DF Hex, then the vector table is located between addresses 02E0 and 02FF Hex, and so on.

Each vector is 15 bits long and points to the beginning of a specific interrupt service routine somewhere in the 32 kbyte memory space. Each vector occupies two bytes of the vector table, with the higher-order byte at the lower address. The vectors are arranged in order of interrupt priority. The vector of the maskable interrupt with the lowest rank is located to 0yE0 (higher-order byte) and 0yE1 (lower-order byte). The next priority interrupt is located at 0yE2 and 0yE3, and so forth in increasing rank. The Software Trap has the highest

rank and its vector is always located at 0yFE and 0yFF. The number of interrupts which can become active defines the size of the table.

Table 5 shows the types of interrupts, the interrupt arbitration ranking, and the locations of the corresponding vectors in the vector table.

The vector table should be filled by the user with the memory locations of the specific interrupt service routines. For example, if the Software Trap routine is located at 0310 Hex, then the vector location 0yFE and -0yFF should contain the data 03 and 10 Hex, respectively. When a Software Trap interrupt occurs and the VIS instruction is executed, the program jumps to the address specified in the vector table.

The interrupt sources in the vector table are listed in order of rank, from highest to lowest priority. If two or more enabled and pending interrupts are detected at the same time, the one with the highest priority is serviced first. Upon return from the interrupt service routine, the next highest-level pending interrupt is serviced.

If the VIS instruction is executed, but no interrupts are enabled and pending, the lowest-priority interrupt vector is used, and a jump is made to the corresponding address in the vector table. This is an unusual occurrence, and may be the result of an error. It can legitimately result from a change in the enable bits or pending flags prior to the execution of the VIS instruction, such as executing a single cycle instruction which clears an enable flag at the same time that the pending flag is set. It can also result, however, from inadvertent execution of the VIS command outside of the context of an interrupt.

The default VIS interrupt vector can be useful for applications in which time critical interrupts can occur during the servicing of another interrupt. Rather than restoring the program context (A, B, X, etc.) and executing the RETI instruction, an interrupt service routine can be terminated by returning to the VIS instruction. In this case, interrupts will be serviced in turn until no further interrupts are pending and the default VIS routine is started. After testing the GIE bit to ensure that execution is not erroneous, the routine should restore the program context and execute the RETI to return to the interrupted program.

This technique can save up to fifty instruction cycles ( $t_c$ ), or more, (50 $\mu$ s at 10 MHz oscillator) of latency for pending interrupts with a penalty of fewer than ten instruction cycles if no further interrupts are pending.

To ensure reliable operation, the user should always use the VIS instruction to determine the source of an interrupt. Although it is possible to poll the pending bits to detect the source of an interrupt, this practice is not recommended. The use of polling allows the standard arbitration ranking to be altered, but the reliability of the interrupt system is compromised. The polling routine must individually test the enable and pending bits of each maskable interrupt. If a Software Trap interrupt should occur, it will be serviced last, even though it should have the highest priority. Under certain conditions, a Software Trap could be triggered but not serviced, resulting in an inadvertent "locking out" of all maskable interrupts by the Software Trap pending flag. Problems such as this can be avoided by using VIS instruction.

## Interrupts (Continued)

**TABLE 5. Interrupt Vector Table**

| ARBITRATION RANKING | SOURCE DESCRIPTION       |                                      | VECTOR* ADDRESS (Hi-Low Byte) |
|---------------------|--------------------------|--------------------------------------|-------------------------------|
| (1) Highest         | Software                 | INTR Instruction                     | 0yFE–0yFF                     |
| (2)                 | Reserved                 |                                      | 0yFC–0yFD                     |
| (3)                 | External                 | G0                                   | 0yFA–0yFB                     |
| (4)                 | Timer T0                 | Idle Timer                           | 0yF8–0yF9                     |
| (5)                 | Timer T1                 | T1A/Underflow                        | 0yF6–0yF7                     |
| (6)                 | Timer T1                 | T1B                                  | 0yF4–0yF5                     |
| (7)                 | MICROWIRE/PLUS           | Busy Low                             | 0yF2–0yF3                     |
| (8)                 | Reserved                 |                                      | 0yF0–0yF1                     |
| (9)                 | Reserved                 |                                      | 0yEE–0yEF                     |
| (10)                | Reserved                 |                                      | 0yEC–0yED                     |
| (11)                | High Speed Capture Timer | Capture Overflow/<br>Capture Pending | 0yEA–0yEB                     |
| (12)                | Reserved                 |                                      | 0yE8–0yE9                     |
| (13)                | Reserved                 |                                      | 0yE6–0yE7                     |
| (14)                | Reserved                 |                                      | 0yE4–0yE5                     |
| (15)                | Port L/Wakeup            | Port L Edge                          | 0yE2–0yE3                     |
| (16) Lowest         | Default VIS              | Reserved                             | 0yE0–0yE1                     |

**Note 20:** \*y is a variable which represents the VIS block. VIS and the vector table must be located in the same 256-byte block except if VIS is located at the last address of a block. In this case, the table must be in the next block.

### VIS Execution

When the VIS instruction is executed it activates the arbitration logic. The arbitration logic generates an even number between E0 and FE (E0, E2, E4, E6 etc...) depending on which active interrupt has the highest arbitration ranking at the time of the 1st cycle of VIS is executed. For example, if the software trap interrupt is active, FE is generated. If the external interrupt is active and the software trap interrupt is not, then FA is generated and so forth. If the only active interrupt is software trap, then E0 is generated. This number replaces the lower byte of the PC. The upper byte of the PC remains unchanged. The new PC is therefore pointing to the

vector of the active interrupt with the highest arbitration ranking. This vector is read from program memory and placed into the PC which is now pointed to the 1st instruction of the service routine of the active interrupt with the highest arbitration ranking.

*Figure 15* illustrates the different steps performed by the VIS instruction. *Figure 16* shows a flowchart for the VIS instruction.

The non-maskable interrupt pending flag is cleared by the RPND (Reset Non-Maskable Pending Bit) instruction (under certain conditions) and upon RESET.

## Interrupts (Continued)

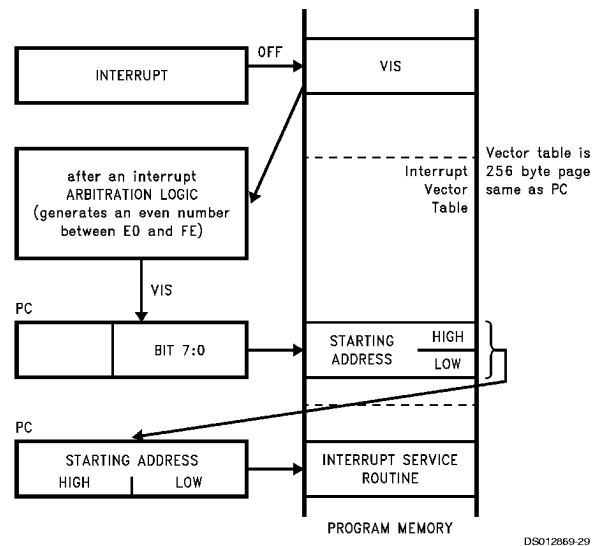


FIGURE 15. VIS Operation

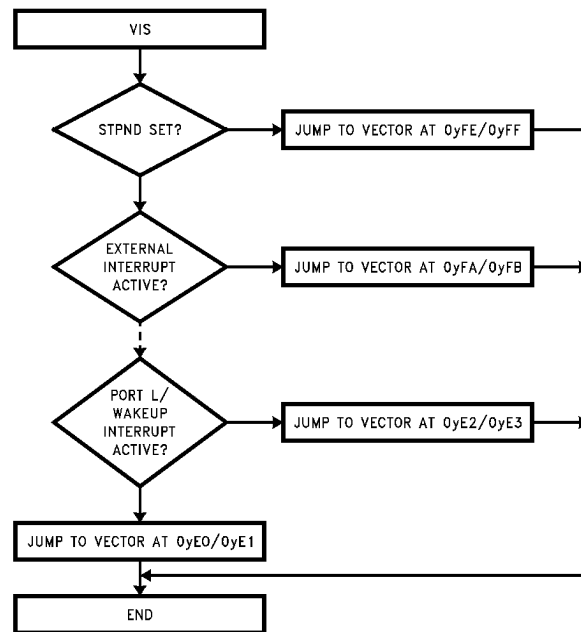


FIGURE 16. VIS Flowchart



## Interrupts (Continued)

### Programming Example: External Interrupt

```
        PSW          =00EF
        CNTRL        =00EE
        RBIT         0,PORTGC
        RBIT         0,PORTGD      ; G0 pin configured Hi-Z
        SBIT         IEDG, CNTRL   ; Ext interrupt polarity; falling edge
        SBIT         EXEN, PSW     ; Enable the external interrupt
        SBIT         GIE, PSW     ; Set the GIE bit
WAIT:   JP          WAIT          ; Wait for external interrupt
        .
        .
        .=0FF              ; The interrupt causes a
        VIS              ; branch to address 0FF
                          ; The VIS causes a branch to
                          ; interrupt vector table
        .
        .
        .=01FA            ; Vector table (within 256 byte
        .ADDRW SERVICE    ; of VIS inst.) containing the ext
                          ; interrupt service routine
        .
        .
INT_EXIT: RETI
        .
        .
SERVICE: RBIT      EXPND, PSW      ; Interrupt Service Routine
                          ; Reset ext interrupt pend. bit
        .
        .
        .
        JP      INT_EXIT          ; Return, set the GIE bit
```

## Interrupts (Continued)

### NON-MASKABLE INTERRUPT

#### Pending Flag

There is a pending flag bit associated with the non-maskable interrupt, called STPND. This pending flag is not memory-mapped and cannot be accessed directly by the software.

The pending flag is reset to zero when a device Reset occurs. When the non-maskable interrupt occurs, the associated pending bit is set to 1. The interrupt service routine should contain an RPND instruction to reset the pending flag to zero. The RPND instruction always resets the STPND flag.

#### Software Trap

The Software Trap is a special kind of non-maskable interrupt which occurs when the INTR instruction (used to acknowledge interrupts) is fetched from program memory and placed in the instruction register. This can happen in a variety of ways, usually because of an error condition. Some examples of causes are listed below.

If the program counter incorrectly points to a memory location beyond the available program memory space, the non-existent or unused memory location returns zeroes which is interpreted as the INTR instruction.

If the stack is popped beyond the allowed limit (address 06F Hex), a 7FFF will be loaded into the PC, if this last location in program memory is unprogrammed or unavailable, a Software Trap will be triggered.

A Software Trap can be triggered by a temporary hardware condition such as a brownout or power supply glitch.

The Software Trap has the highest priority of all interrupts. When a Software Trap occurs, the STPND bit is set. The GIE bit is not affected and the pending bit (not accessible by the user) is used to inhibit other interrupts and to direct the program to the ST service routine with the VIS instruction. Nothing can interrupt a Software Trap service routine except for another Software Trap. The STPND can be reset only by the RPND instruction or a chip Reset.

The Software Trap indicates an unusual or unknown error condition. Generally, returning to normal execution at the point where the Software Trap occurred cannot be done reliably. Therefore, the Software Trap service routine should reinitialize the stack pointer and perform a recovery procedure that restarts the software at some known point, similar to a device Reset, but not necessarily performing all the same functions as a device Reset. The routine must also execute the RPND instruction to reset the STPND flag. Otherwise, all other interrupts will be locked out. To the extent possible, the interrupt routine should record or indicate the context of the device so that the cause of the Software Trap can be determined.

If the user wishes to return to normal execution from the point at which the Software Trap was triggered, the user must first execute RPND, followed by RETSK rather than RETI or RET. This is because the return address stored on the stack is the address of the INTR instruction that triggered the interrupt. The program must skip that instruction in order to proceed with the next one. Otherwise, an infinite loop of Software Traps and returns will occur.

Programming a return to normal execution requires careful consideration. If the Software Trap routine is interrupted by another Software Trap, the RPND instruction in the service routine for the second Software Trap will reset the STPND

flag; upon return to the first Software Trap routine, the STPND flag will have the wrong state. This will allow maskable interrupts to be acknowledged during the servicing of the first Software Trap. To avoid problems such as this, the user program should contain the Software Trap routine to perform a recovery procedure rather than a return to normal execution.

Under normal conditions, the STPND flag is reset by a RPND instruction in the Software Trap service routine. If a programming error or hardware condition (brownout, power supply glitch, etc.) sets the STPND flag without providing a way for it to be cleared, all other interrupts will be locked out. To alleviate this condition, the user can use extra RPND instructions in the main program and in the WATCHDOG service routine (if present). There is no harm in executing extra RPND instructions in these parts of the program.

### PORT L INTERRUPTS

Port L provides the user with an additional eight fully selectable, edge sensitive interrupts which are all vectored into the same service subroutine.

The interrupt from Port L shares logic with the wake up circuitry. The register WKEN allows interrupts from Port L to be individually enabled or disabled. The register WKEDG specifies the trigger condition to be either a positive or a negative edge. Finally, the register WKPND latches in the pending trigger conditions.

The GIE (Global Interrupt Enable) bit enables the interrupt function.

A control flag, LPEN, functions as a global interrupt enable for Port L interrupts. Setting the LPEN flag will enable interrupts and vice versa. A separate global pending flag is not needed since the register WKPND is adequate.

Since Port L is also used for waking the device out of the HALT or IDLE modes, the user can elect to exit the HALT or IDLE modes either with or without the interrupt enabled. If he elects to disable the interrupt, then the device will restart execution from the instruction immediately following the instruction that placed the microcontroller in the HALT or IDLE modes. In the other case, the device will first execute the interrupt service routine and then revert to normal operation. (See HALT MODE for clock option wakeup information.)

### INTERRUPT SUMMARY

The device uses the following types of interrupts, listed below in order of priority:

1. The Software Trap non-maskable interrupt, triggered by the INTR (00 opcode) instruction. The Software Trap is acknowledged immediately. This interrupt service routine can be interrupted only by another Software Trap. The Software Trap should end with two RPND instructions followed by a restart procedure.
2. Maskable interrupts, triggered by an on-chip peripheral block or an external device connected to the device. Under ordinary conditions, a maskable interrupt will not interrupt any other interrupt routine in progress. A maskable interrupt routine in progress can be interrupted by the non-maskable interrupt request. A maskable interrupt routine should end with an RETI instruction or, prior to restoring context, should return to execute the VIS instruction. This is particularly useful when exiting long interrupt service routines if the time between interrupts is short. In this case the RETI instruction would only be executed when the default VIS routine is reached.

## WATCHDOG

The devices contain a WATCHDOG and clock monitor. The WATCHDOG is designed to detect the user program getting stuck in infinite loops resulting in loss of program control or “runaway” programs. The Clock Monitor is used to detect the absence of a clock or a very slow clock below a specified rate on the CKI pin.

The WATCHDOG consists of two independent logic blocks: WD UPPER and WD LOWER. WD UPPER establishes the upper limit on the service window and WD LOWER defines the lower limit of the service window.

Servicing the WATCHDOG consists of writing a specific value to a WATCHDOG Service Register named WDSVR which is memory mapped in the RAM. This value is composed of three fields, consisting of a 2-bit Window Select, a 5-bit Key Data field, and the 1-bit Clock Monitor Select field. Table 6 shows the WDSVR register.

**TABLE 6. WATCHDOG Service Register (WDSVR)**

| Window Select |   | Key Data |   |   |   |   | Clock Monitor |
|---------------|---|----------|---|---|---|---|---------------|
| X             | X | 0        | 1 | 1 | 0 | 0 | Y             |
| 7             | 6 | 5        | 4 | 3 | 2 | 1 | 0             |

The lower limit of the service window is fixed at 2048 instruction cycles. Bits 7 and 6 of the WDSVR register allow the user to pick an upper limit of the service window.

Table 7 shows the four possible combinations of lower and upper limits for the WATCHDOG service window. This flexibility in choosing the WATCHDOG service window prevents any undue burden on the user software.

Bits 5, 4, 3, 2 and 1 of the WDSVR register represent the 5-bit Key Data field. The key data is fixed at 01100. Bit 0 of the WDSVR Register is the Clock Monitor Select bit.

**TABLE 7. WATCHDOG Service Window Select**

| WDSVR Bit 7 | WDSVR Bit 6 | Service Window (Lower-Upper Limits) |
|-------------|-------------|-------------------------------------|
| 0           | 0           | 2k–8k $t_C$ Cycles                  |
| 0           | 1           | 2k–16k $t_C$ Cycles                 |
| 1           | 0           | 2k–32k $t_C$ Cycles                 |
| 1           | 1           | 2k–64k $t_C$ Cycles                 |

## Clock Monitor

The Clock Monitor aboard the device can be selected or deselected under program control. The Clock Monitor is guaranteed not to reject the clock if the instruction cycle clock ( $1/t_C$ ) is greater or equal to 10 kHz. This equates to a clock input rate on CKI of greater or equal to 100 kHz.

## WATCHDOG Operation

The WATCHDOG and Clock Monitor are disabled during reset. The device comes out of reset with the WATCHDOG armed, the WATCHDOG Window Select bits (bits 6, 7 of the WDSVR Register) set, and the Clock Monitor bit (bit 0 of the WDSVR Register) enabled. Thus, a Clock Monitor error will occur after coming out of reset, if the instruction cycle clock frequency has not reached a minimum specified value, including the case where the oscillator fails to start.

The WDSVR register can be written to only once after reset and the key data (bits 5 through 1 of the WDSVR Register) must match to be a valid write. This write to the WDSVR register involves two irrevocable choices: (i) the selection of the WATCHDOG service window (ii) enabling or disabling of the Clock Monitor. Hence, the first write to WDSVR Register involves selecting or deselecting the Clock Monitor, select the WATCHDOG service window and match the WATCHDOG key data. Subsequent writes to the WDSVR register will compare the value being written by the user to the WATCHDOG service window value and the key data (bits 7 through 1) in the WDSVR Register. Table IX shows the sequence of events that can occur.

The user must service the WATCHDOG at least once before the upper limit of the service window expires. The WATCHDOG may not be serviced more than once in every lower limit of the service window. The user may service the WATCHDOG as many times as wished in the time period between the lower and upper limits of the service window. The first write to the WDSVR Register is also counted as a WATCHDOG service.

The WATCHDOG has an output pin associated with it. This is the WDOUT pin, on pin 1 of the port G. WDOUT is active low. The WDOUT pin is in the high impedance state in the inactive state. Upon triggering the WATCHDOG, the logic will pull the WDOUT (G1) pin low for an additional 16  $t_C$ –32  $t_C$  cycles after the signal level on WDOUT pin goes below the lower Schmitt trigger threshold. After this delay, the device will stop forcing the WDOUT output low. The WATCHDOG service window will restart when the WDOUT pin goes high. It is recommended that the user tie the WDOUT pin back to  $V_{CC}$  through a resistor in order to pull WDOUT high.

A WATCHDOG service while the WDOUT signal is active will be ignored. The state of the WDOUT pin is not guaranteed on reset, but if it powers up low then the WATCHDOG will time out and WDOUT will enter high impedance state.

The Clock Monitor forces the G1 pin low upon detecting a clock frequency error. The Clock Monitor error will continue until the clock frequency has reached the minimum specified value, after which the G1 output will enter the high impedance TRI-STATE mode following 16  $t_C$ –32  $t_C$  clock cycles. The Clock Monitor generates a continual Clock Monitor error if the oscillator fails to start, or fails to reach the minimum specified frequency. The specification for the Clock Monitor is as follows:

$1/t_C > 10 \text{ kHz}$  — No clock rejection.

$1/t_C < 10 \text{ Hz}$  — Guaranteed clock rejection.

## WATCHDOG AND CLOCK MONITOR SUMMARY

The following salient points regarding the WATCHDOG and CLOCK MONITOR should be noted:

- Both the WATCHDOG and CLOCK MONITOR detector circuits are inhibited during RESET.
- Following RESET, the WATCHDOG and CLOCK MONITOR are both enabled, with the WATCHDOG having the maximum service window selected.
- The WATCHDOG service window and CLOCK MONITOR enable/disable option can only be changed once, during the initial WATCHDOG service following RESET.
- The initial WATCHDOG service must match the key data value in the WATCHDOG Service register WDSVR in order to avoid a WATCHDOG error.

## WATCHDOG Operation (Continued)

- Subsequent WATCHDOG services must match all three data fields in WDSVR in order to avoid WATCHDOG errors.
- The correct key data value cannot be read from the WATCHDOG Service register WDSVR. Any attempt to read this key data value of 01100 from WDSVR will read as key data value of all 0's.
- The WATCHDOG detector circuit is inhibited during both the HALT and IDLE modes.
- The CLOCK MONITOR detector circuit is active during both the HALT and IDLE modes. Consequently, the device inadvertently entering the HALT mode will be detected as a CLOCK MONITOR error (provided that the CLOCK MONITOR enable option has been selected by the program).
- With the single-pin R/C oscillator mask option selected and the CLKDLY bit reset, the WATCHDOG service window will resume following HALT mode from where it left off before entering the HALT mode.
- With the crystal oscillator mask option selected, or with the single-pin R/C oscillator mask option selected and the CLKDLY bit set, the WATCHDOG service window will be

set to its selected value from WDSVR following HALT. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following HALT, but must be serviced within the selected window to avoid a WATCHDOG error.

- The IDLE timer T0 is not initialized with RESET.
- The user can sync in to the IDLE counter cycle with an IDLE counter (T0) interrupt or by monitoring the TOPND flag. The TOPND flag is set whenever the thirteenth bit of the IDLE counter toggles (every 4096 instruction cycles). The user is responsible for resetting the TOPND flag.
- A hardware WATCHDOG service occurs just as the device exits the IDLE mode. Consequently, the WATCHDOG should not be serviced for at least 2048 instruction cycles following IDLE, but must be serviced within the selected window to avoid a WATCHDOG error.
- Following RESET, the initial WATCHDOG service (where the service window and the CLOCK MONITOR enable/disable must be selected) may be programmed anywhere within the maximum service window (65,536 instruction cycles) initialized by RESET. Note that this initial WATCHDOG service may be programmed within the initial 2048 instruction cycles without causing a WATCHDOG error.

TABLE 8. WATCHDOG Service Actions

| Key Data   | Window Data | Clock Monitor | Action                                |
|------------|-------------|---------------|---------------------------------------|
| Match      | Match       | Match         | Valid Service: Restart Service Window |
| Don't Care | Mismatch    | Don't Care    | Error: Generate WATCHDOG Output       |
| Mismatch   | Don't Care  | Don't Care    | Error: Generate WATCHDOG Output       |
| Don't Care | Don't Care  | Mismatch      | Error: Generate WATCHDOG Output       |

## Detection of Illegal Conditions

The device can detect various illegal conditions resulting from coding errors, transient noise, power supply voltage drops, runaway programs, etc.

Reading of undefined ROM gets zeros. The opcode for software interrupt is 00. If the program fetches instructions from undefined ROM, this will force a software interrupt, thus signaling that an illegal condition has occurred.

The subroutine stack grows down for each call (jump to subroutine), interrupt, or PUSH, and grows up for each return or POP. The stack pointer is initialized to RAM location 06F Hex during reset. Consequently, if there are more returns than calls, the stack pointer will point to addresses 070 and 071 Hex (which are undefined RAM). Undefined RAM from addresses 070 to 07F (Segment 0), and all other segments (i.e., Segments 4... etc.) is read as all 1's, which in turn will cause the program to return to address 7FFF Hex. This is an undefined ROM location and the instruction fetched (all 0's) from this location will generate a software interrupt signaling an illegal condition.

Thus, the chip can detect the following illegal conditions:

1. Executing from undefined ROM
2. Over "POP"ing the stack by having more returns than calls.

When the software interrupt occurs, the user can re-initialize the stack pointer and do a recovery procedure before restarting (this recovery program is probably similar to that following reset, but might not contain the same program initialization procedures). The recovery program should reset the software interrupt pending bit using the RPNL instruction.

## MICROWIRE/PLUS

MICROWIRE/PLUS is a serial synchronous communications interface. The MICROWIRE/PLUS capability enables the device to interface with any of National Semiconductor's MICROWIRE peripherals (i.e. A/D converters, display drivers, E2PROMs etc.) and with other microcontrollers which support the MICROWIRE interface. It consists of an 8-bit serial shift register (SIO) with serial data input (SI), serial data output (SO) and serial shift clock (SK). Figure 17 shows a block diagram of the MICROWIRE/PLUS logic.

The shift clock can be selected from either an internal source or an external source. Operating the MICROWIRE/PLUS arrangement with the internal clock source is called the Master mode of operation. Similarly, operating the MICROWIRE/PLUS arrangement with an external shift clock is called the Slave mode of operation.

The CNTRL register is used to configure and control the MICROWIRE/PLUS mode. To use the MICROWIRE/PLUS, the MSEL bit in the CNTRL register is set to one. In the master mode, the SK clock rate is selected by the two bits, SL0 and SL1, in the CNTRL register. Table 9 details the different clock rates that may be selected.

TABLE 9. MICROWIRE/PLUS Master Mode Clock Select

| SL1 | SL0 | SK period          |
|-----|-----|--------------------|
| 0   | 0   | 2 X t <sub>C</sub> |
| 0   | 1   | 4 X t <sub>C</sub> |
| 1   | x   | 8 X t <sub>C</sub> |

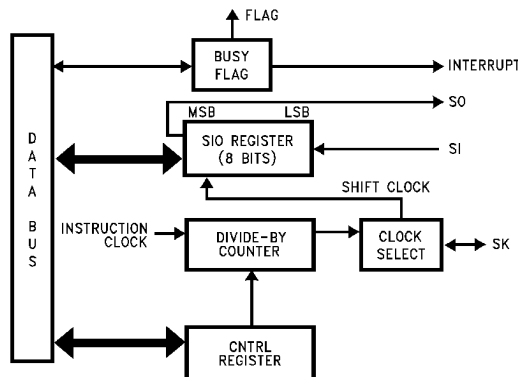
Where t<sub>C</sub> is the instruction cycle clock

## MICROWIRE/PLUS (Continued)

### MICROWIRE/PLUS OPERATION

Setting the BUSY bit in the PSW register causes the MICROWIRE/PLUS to start shifting the data. It gets reset when eight data bits have been shifted. The user may reset the BUSY bit by software to allow less than 8 bits to shift. If

enabled, an interrupt is generated when eight data bits have been shifted. The device may enter the MICROWIRE/PLUS mode either as a Master or as a Slave. Figure 18 shows how two devices, microcontrollers and several peripherals may be interconnected using the MICROWIRE/PLUS arrangements.



DS012869-16

FIGURE 17. MICROWIRE/PLUS Block Diagram

### WARNING

The SIO register should only be loaded when the SK clock is low. Loading the SIO register while the SK clock is high will result in undefined data in the SIO register. SK clock is normally low when not shifting.

Setting the BUSY flag when the input SK clock is high in the MICROWIRE/PLUS slave mode may cause the current SK clock for the SIO shift register to be narrow. For safety, the BUSY flag should only be set when the input SK clock is low.

### MICROWIRE/PLUS Master Mode Operation

In the MICROWIRE/PLUS Master mode of operation the shift clock (SK) is generated internally. The MICROWIRE Master always initiates all data exchanges. The MSEL bit in the CNTRL register must be set to enable the SO and SK functions onto the G Port. The SO and SK pins must also be selected as outputs by setting appropriate bits in the Port G configuration register. Table 10 summarizes the bit settings required for Master mode of operation.

### MICROWIRE/PLUS Slave Mode Operation

In the MICROWIRE/PLUS Slave mode of operation the SK clock is generated by an external source. Setting the MSEL bit in the CNTRL register enables the SO and SK functions onto the G Port. The SK pin must be selected as an input and the SO pin is selected as an output pin by setting and resetting the appropriate bits in the Port G configuration register. Table XI summarizes the settings required to enter the Slave mode of operation.

The user must set the BUSY flag immediately upon entering the Slave mode. This will ensure that all data bits sent by the Master will be shifted properly. After eight clock pulses the BUSY flag will be cleared and the sequence may be repeated.

TABLE 10. MICROWIRE/PLUS Mode Settings

This table assumes that the control flag MSEL is set.

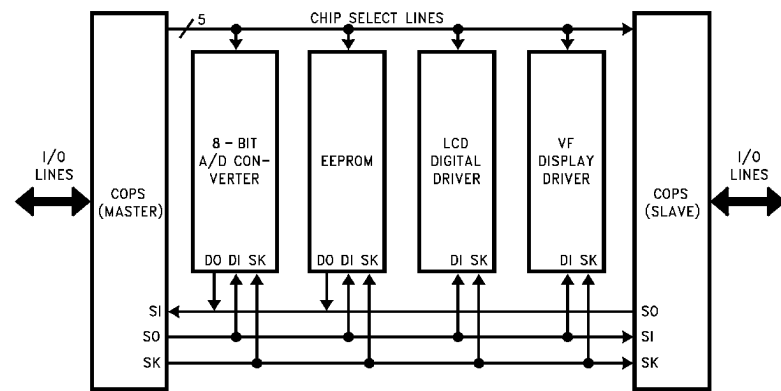
| G4 (SO)<br>Config. Bit | G5 (SK)<br>Config. Bit | G4<br>Fun. | G5<br>Fun. | Operation             |
|------------------------|------------------------|------------|------------|-----------------------|
| 1                      | 1                      | SO         | Int. SK    | MICROWIRE/PLUS Master |
| 0                      | 1                      | TRI-STATE  | Int. SK    | MICROWIRE/PLUS Master |
| 1                      | 0                      | SO         | Ext. SK    | MICROWIRE/PLUS Slave  |
| 0                      | 0                      | TRI-STATE  | Ext. SK    | MICROWIRE/PLUS Slave  |

### Alternate SK Phase Operation

The device allows either the normal SK clock or an alternate phase SK clock to shift data in and out of the SIO register. In both the modes the SK is normally low. In the normal mode data is shifted in on the rising edge of the SK clock and the data is shifted out on the falling edge of the SK clock. The SIO register is shifted on each falling edge of the SK clock. In the alternate SK phase operation, data is shifted in on the falling edge of the SK clock and shifted out on the rising edge of the SK clock.

A control flag, SKSEL, allows either the normal SK clock or the alternate SK clock to be selected. Resetting SKSEL causes the MICROWIRE/PLUS logic to be clocked from the normal SK signal. Setting the SKSEL flag selects the alternate SK clock. The SKSEL is mapped into the G6 configuration bit. The SKSEL flag will power up in the reset condition, selecting the normal SK signal.

## MICROWIRE/PLUS (Continued)



DS012869-17

FIGURE 18. MICROWIRE/PLUS Application

## Memory Map

All RAM, ports and registers (except A and PC) are mapped into data memory address space.

| Address<br>S/ADD<br>REG | Contents                                        |
|-------------------------|-------------------------------------------------|
| 0000 to 006F            | On-Chip RAM bytes (112 bytes)                   |
| 0070 to 007F            | Unused RAM Address Space (Reads As All Ones)    |
| xx80 to xxAF            | Unused RAM Address Space (Reads Undefined Data) |
| xxB0                    | Reserved                                        |
| XXB1                    | Reserved                                        |
| xxB2                    | Reserved                                        |
| xxB3                    | Reserved                                        |
| xxB4                    | Reserved                                        |
| xxB5                    | Reserved                                        |
| xxB6                    | Reserved                                        |
| xxB7                    | Comparator Select Register (CMPSL)              |
| xxB8 to xxBF            | Reserved                                        |
| xxC0                    | Reserved                                        |
| xxC1                    | Reserved                                        |
| xxC2                    | Reserved                                        |
| xxC3                    | Reserved                                        |
| xxC4                    | Reserved                                        |
| xxC5                    | Reserved                                        |
| xxC6                    | Reserved                                        |
| xxC7                    | WATCHDOG Service Register (Reg:WDSVR)           |
| xxC8                    | MIWU Edge Select Register (Reg:WKEDG)           |
| xxC9                    | MIWU Enable Register (Reg:WKEN)                 |
| xxCA                    | MIWU Pending Register (Reg:WKPNL)               |
| xxCB                    | Reserved                                        |
| xxCC                    | CAPTLO (Capture Timer Low-Byte)                 |
| xxCD                    | CAPTHI (Capture Timer High-Byte)                |
| xxCE                    | CAPCNTL (Capture Timer Control Register)        |
| xxCF                    | Idle Timer Control Register                     |
| xxD0                    | Port L Data Register                            |
| xxD1                    | Port L Configuration Register                   |
| xxD2                    | Port L Input Pins (Read Only)                   |
| xxD3                    | Reserved                                        |
| xxD4                    | Port G Data Register                            |
| xxD5                    | Port G Configuration Register                   |
| xxD6                    | Port G Input Pins (Read Only)                   |
| xxD7                    | Port I Input Pins (Read Only)                   |
| xxD8                    | Reserved                                        |
| xxD9                    | Reserved                                        |
| xxDA                    | Reserved                                        |

| Address<br>S/ADD<br>REG | Contents                                   |
|-------------------------|--------------------------------------------|
| xxDB                    | Reserved                                   |
| xxDC                    | Port D                                     |
| xxDD to DF              | Reserved                                   |
| xxE0 to xxE5            | Reserved                                   |
| xxE6                    | Timer T1 Autoload Register T1RB Lower Byte |
| xxE7                    | Timer T1 Autoload Register T1RB Upper Byte |
| xxE8                    | ICNTRL Register                            |
| xxE9                    | MICROWIRE/PLUS Shift Register              |
| xxEA                    | Timer T1 Lower Byte                        |
| xxEB                    | Timer T1 Upper Byte                        |
| xxEC                    | Timer T1 Autoload Register T1RA Lower Byte |
| xxED                    | Timer T1 Autoload Register T1RA Upper Byte |
| xxEE                    | CNTRL Control Register                     |
| xxEF                    | PSW Register                               |
| xxF0 to FB              | On-Chip RAM Mapped as Registers            |
| xxFC                    | X Register                                 |
| xxFD                    | SP Register                                |
| xxFE                    | B Register                                 |
| xxFF                    | Reserved                                   |
| 0100-017F               | Reserved                                   |

Reading memory locations 0070H-007FH (Segment 0) will return all ones. Reading unused memory locations 0080H-00AFH (Segment 0) will return undefined data. Reading memory locations from other Segments (i.e., Segment 2, Segment 3,...etc.) will return undefined data.

## Addressing Modes

There are ten addressing modes, six for operand addressing and four for transfer of control.

### OPERAND ADDRESSING MODES

#### Register Indirect

This is the "normal" addressing mode. The operand is the data memory addressed by the B pointer or X pointer.

#### Register Indirect (with auto post increment or decrement of pointer)

This addressing mode is used with the LD and X instructions. The operand is the data memory addressed by the B pointer or X pointer. This is a register indirect mode that automatically post increments or decrements the B or X register after executing the instruction.

#### Direct

The instruction contains an 8-bit address field that directly points to the data memory for the operand.

## Addressing Modes (Continued)

### Immediate

This addressing mode contains an 8-bit immediate field as the operand.

### Short Immediate

This addressing mode is used with the Load B Immediate instruction. The instruction contains a 4-bit immediate field as the operand.

### Indirect

This addressing mode is used with the LAID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a data operand from the program memory.

## TRANSFER OF CONTROL ADDRESSING MODES

### Relative

This mode is used for the JP instruction, with the instruction field being added to the program counter to get the new program location. JP has a range from -31 to +32 to allow a 1-byte relative jump (JP + 1 is implemented by a NOP instruction). There are no "pages" when using JP, since all 15 bits of PC are used.

### Absolute

This mode is used with the JMP and JSR instructions, with the instruction field of 12 bits replacing the lower 12 bits of the program counter (PC). This allows jumping to any location in the current 4k program memory segment.

### Absolute Long

This mode is used with the JMWL and JSRL instructions, with the instruction field of 15 bits replacing the entire 15 bits of the program counter (PC). This allows jumping to any location up to 32k in the program memory space.

### Indirect

This mode is used with the JID instruction. The contents of the accumulator are used as a partial address (lower 8 bits of PC) for accessing a location in the program memory. The contents of this program memory location serve as a partial address (lower 8 bits of PC) for the jump to the next instruction.

## INSTRUCTION SET

**Note:** The VIS is a special case of the Indirect Transfer of Control addressing mode, where the double byte vector associated with the interrupt is transferred from adjacent addresses in the program memory into the program counter (PC) in order to jump to the associated interrupt service routine.

## Instruction Set

### Register and Symbol Definition

| Registers |                                                            |
|-----------|------------------------------------------------------------|
| A         | 8-Bit Accumulator Register                                 |
| B         | 8-Bit Address Register                                     |
| X         | 8-Bit Address Register                                     |
| SP        | 8-Bit Stack Pointer Register                               |
| PC        | 15-Bit Program Counter Register                            |
| PU        | Upper 7 Bits of PC                                         |
| PL        | Lower 8 Bits of PC                                         |
| C         | 1-Bit of PSW Register for Carry                            |
| HC        | 1-Bit of PSW Register for Half Carry                       |
| GIE       | 1-Bit of PSW Register for Global Interrupt Enable          |
| VU        | Interrupt Vector Upper Byte                                |
| VL        | Interrupt Vector Lower Byte                                |
| Symbols   |                                                            |
| [B]       | Memory Indirectly Addressed by B Register                  |
| [X]       | Memory Indirectly Addressed by X Register                  |
| MD        | Direct Addressed Memory                                    |
| Mem       | Direct Addressed Memory or [B]                             |
| Meml      | Direct Addressed Memory or [B] or Immediate Data           |
| Imm       | 8-Bit Immediate Data                                       |
| Reg       | Register Memory: Addresses F0 to FF (Includes B, X and SP) |
| Bit       | Bit Number (0 to 7)                                        |
| ←         | Loaded with                                                |
| ↔         | Exchanged with                                             |

|       |        |                                  |                                                                                                |
|-------|--------|----------------------------------|------------------------------------------------------------------------------------------------|
| ADD   | A,Meml | ADD                              | $A \leftarrow A + \text{Meml}$                                                                 |
| ADC   | A,Meml | ADD with Carry                   | $A \leftarrow A + \text{Meml} + C, C \leftarrow \text{Carry}, HC \leftarrow \text{Half Carry}$ |
| SUBC  | A,Meml | Subtract with Carry              | $A \leftarrow A - \text{Meml} + C, C \leftarrow \text{Carry}, HC \leftarrow \text{Half Carry}$ |
| AND   | A,Meml | Logical AND                      | $A \leftarrow A \text{ and } \overline{\text{Meml}}$                                           |
| ANDSZ | A,Imm  | Logical AND Immed., Skip if Zero | Skip next if $(A \text{ and } \text{Imm}) = 0$                                                 |
| OR    | A,Meml | Logical OR                       | $A \leftarrow A \text{ or } \text{Meml}$                                                       |
| XOR   | A,Meml | Logical EXclusive OR             | $A \leftarrow A \text{ xor } \text{Meml}$                                                      |
| IFEQ  | MD,Imm | IF Equal                         | Compare MD and Imm, Do next if $MD = \text{Imm}$                                               |
| IFEQ  | A,Meml | IF Equal                         | Compare A and Meml, Do next if $A = \text{Meml}$                                               |
| IFNE  | A,Meml | IF Not Equal                     | Compare A and Meml, Do next if $A \neq \text{Meml}$                                            |
| IFGT  | A,Meml | IF Greater Than                  | Compare A and Meml, Do next if $A > \text{Meml}$                                               |
| IFBNE | #      | If B Not Equal                   | Do next if lower 4 bits of $B \neq \text{Imm}$                                                 |
| DRSZ  | Reg    | Decrement Reg., Skip if Zero     | $\text{Reg} \leftarrow \text{Reg} - 1$ , Skip if $\text{Reg} = 0$                              |



| Instruction Set (Continued) |          |                                     |                                                                                                                            |
|-----------------------------|----------|-------------------------------------|----------------------------------------------------------------------------------------------------------------------------|
| SBIT                        | #,Mem    | Set BIT                             | 1 to bit, Mem (bit = 0 to 7 immediate)                                                                                     |
| RBIT                        | #,Mem    | Reset BIT                           | 0 to bit, Mem                                                                                                              |
| IFBIT                       | #,Mem    | IF BIT                              | If bit #,A or Mem is true do next instruction                                                                              |
| RPND                        |          | Reset PeNDing Flag                  | Reset Software Interrupt Pending Flag                                                                                      |
| X                           | A,Mem    | EXchange A with Memory              | $A \leftrightarrow \text{Mem}$                                                                                             |
| X                           | A,[X]    | EXchange A with Memory [X]          | $A \leftrightarrow [X]$                                                                                                    |
| LD                          | A,Meml   | LoaD A with Memory                  | $A \leftarrow \text{Meml}$                                                                                                 |
| LD                          | A,[X]    | LoaD A with Memory [X]              | $A \leftarrow [X]$                                                                                                         |
| LD                          | B,Imm    | LoaD B with Immed.                  | $B \leftarrow \text{Imm}$                                                                                                  |
| LD                          | Mem,Imm  | LoaD Memory Immed                   | $\text{Mem} \leftarrow \text{Imm}$                                                                                         |
| LD                          | Reg,Imm  | LoaD Register Memory Immed.         | $\text{Reg} \leftarrow \text{Imm}$                                                                                         |
| X                           | A, [B±]  | EXchange A with Memory [B]          | $A \leftrightarrow [B], (B \leftarrow B \pm 1)$                                                                            |
| X                           | A, [X±]  | EXchange A with Memory [X]          | $A \leftrightarrow [X], (X \leftarrow X \pm 1)$                                                                            |
| LD                          | A, [B±]  | LoaD A with Memory [B]              | $A \leftarrow [B], (B \leftarrow B \pm 1)$                                                                                 |
| LD                          | A, [X±]  | LoaD A with Memory [X]              | $A \leftarrow [X], (X \leftarrow X \pm 1)$                                                                                 |
| LD                          | [B±],Imm | LoaD Memory [B] Immed.              | $[B] \leftarrow \text{Imm}, (B \leftarrow B \pm 1)$                                                                        |
| CLR                         | A        | CLear A                             | $A \leftarrow 0$                                                                                                           |
| INC                         | A        | INCrement A                         | $A \leftarrow A + 1$                                                                                                       |
| DEC                         | A        | DECrement A                         | $A \leftarrow A - 1$                                                                                                       |
| LAID                        |          | Load A InDirect from ROM            | $A \leftarrow \text{ROM (PU,A)}$                                                                                           |
| DCOR                        | A        | Decimal CORrect A                   | $A \leftarrow \text{BCD correction of A (follows ADC, SUBC)}$                                                              |
| RRC                         | A        | Rotate A Right thru C               | $C \rightarrow A7 \rightarrow \dots \rightarrow A0 \rightarrow C$                                                          |
| RLC                         | A        | Rotate A Left thru C                | $C \leftarrow A7 \leftarrow \dots \leftarrow A0 \leftarrow C$                                                              |
| SWAP                        | A        | SWAP nibbles of A                   | $A7\dots A4 \leftrightarrow A3\dots A0$                                                                                    |
| SC                          |          | Set C                               | $C \leftarrow 1, \text{HC} \leftarrow 1$                                                                                   |
| RC                          |          | Reset C                             | $C \leftarrow 0, \text{HC} \leftarrow 0$                                                                                   |
| IFC                         |          | IF C                                | If C is true, do next instruction                                                                                          |
| IFNC                        |          | IF Not C                            | If C is not true, do next instruction                                                                                      |
| POP                         | A        | POP the stack into A                | $\text{SP} \leftarrow \text{SP} + 1, A \leftarrow [\text{SP}]$                                                             |
| PUSH                        | A        | PUSH A onto the stack               | $[\text{SP}] \leftarrow A, \text{SP} \leftarrow \text{SP} - 1$                                                             |
| VIS                         |          | Vector to Interrupt Service Routine | $\text{PU} \leftarrow [\text{VU}], \text{PL} \leftarrow [\text{VL}]$                                                       |
| JMPL                        | Addr.    | Jump absolute Long                  | $\text{PC} \leftarrow \text{ii} (\text{ii} = 15 \text{ bits}, 0 \text{ to } 32\text{k})$                                   |
| JMP                         | Addr.    | Jump absolute                       | $\text{PC9}\dots 0 \leftarrow \text{i} (\text{i} = 12 \text{ bits})$                                                       |
| JP                          | Disp.    | Jump relative short                 | $\text{PC} \leftarrow \text{PC} + r (r \text{ is } -31 \text{ to } +32, \text{ except } 1)$                                |
| JSRL                        | Addr.    | Jump SubRoutine Long                | $[\text{SP}] \leftarrow \text{PL}, [\text{SP}-1] \leftarrow \text{PU}, \text{SP}-2, \text{PC} \leftarrow \text{ii}$        |
| JSR                         | Addr     | Jump SubRoutine                     | $[\text{SP}] \leftarrow \text{PL}, [\text{SP}-1] \leftarrow \text{PU}, \text{SP}-2, \text{PC9}\dots 0 \leftarrow \text{i}$ |
| JID                         |          | Jump InDirect                       | $\text{PL} \leftarrow \text{ROM (PU,A)}$                                                                                   |
| RET                         |          | RETurn from subroutine              | $\text{SP} + 2, \text{PL} \leftarrow [\text{SP}], \text{PU} \leftarrow [\text{SP}-1]$                                      |
| RETSK                       |          | RETurn and SKip                     | $\text{SP} + 2, \text{PL} \leftarrow [\text{SP}], \text{PU} \leftarrow [\text{SP}-1], \text{skip next instruction}$        |
| RETI                        |          | RETurn from Interrupt               | $\text{SP} + 2, \text{PL} \leftarrow [\text{SP}], \text{PU} \leftarrow [\text{SP}-1], \text{GIE} \leftarrow 1$             |
| INTR                        |          | Generate an Interrupt               | $[\text{SP}] \leftarrow \text{PL}, [\text{SP}-1] \leftarrow \text{PU}, \text{SP}-2, \text{PC} \leftarrow \text{OFF}$       |
| NOP                         |          | No OPeration                        | $\text{PC} \leftarrow \text{PC} + 1$                                                                                       |

## Instruction Set (Continued)

### Instruction Execution Time

Most instructions are single byte (with immediate addressing mode instructions taking two bytes).

Most single byte instructions take one cycle time to execute.

Skipped instructions require x number of cycles to be skipped, where x equals the number of bytes in the skipped instruction opcode.

See the BYTES and CYCLES per INSTRUCTION table for details.

### Bytes and Cycles per Instruction

The following table shows the number of bytes and cycles for each instruction in the format of byte/cycle.

#### Arithmetic and Logic Instructions

|       | [B] | Direct | Immed |
|-------|-----|--------|-------|
| ADD   | 1/1 | 3/4    | 2/2   |
| ADC   | 1/1 | 3/4    | 2/2   |
| SUBC  | 1/1 | 3/4    | 2/2   |
| AND   | 1/1 | 3/4    | 2/2   |
| OR    | 1/1 | 3/4    | 2/2   |
| XOR   | 1/1 | 3/4    | 2/2   |
| IFEQ  | 1/1 | 3/4    | 2/2   |
| IFGT  | 1/1 | 3/4    | 2/2   |
| IFBNE | 1/1 |        |       |
| DRSZ  | 1/1 | 1/3    |       |
| SBIT  | 1/1 | 3/4    |       |
| RBIT  | 1/1 | 3/4    |       |
| IFBIT | 1/1 | 3/4    |       |

|      |     |
|------|-----|
| RPND | 1/1 |
|------|-----|

### Instructions Using A and C

|       |     |
|-------|-----|
| CLRA  | 1/1 |
| INCA  | 1/1 |
| DECA  | 1/1 |
| LAID  | 1/3 |
| DCORA | 1/1 |
| RRCA  | 1/1 |
| RLCA  | 1/1 |
| SWAPA | 1/1 |
| SC    | 1/1 |
| RC    | 1/1 |
| IFC   | 1/1 |
| IFNC  | 1/1 |
| PUSHA | 1/3 |
| POPA  | 1/3 |
| ANDSZ | 2/2 |

### Transfer of Control Instructions

|       |     |
|-------|-----|
| JMPL  | 3/4 |
| JMP   | 2/3 |
| JP    | 1/3 |
| JSRL  | 3/5 |
| JSR   | 2/5 |
| JID   | 1/3 |
| VIS   | 1/5 |
| RET   | 1/5 |
| RETSK | 1/5 |
| RETI  | 1/5 |
| INTR  | 1/7 |
| NOP   | 1/1 |

### Memory Transfer Instructions

|                 | Register Indirect |     | Direct | Immed. | Register Indirect Auto Incr and Decr |          |             |
|-----------------|-------------------|-----|--------|--------|--------------------------------------|----------|-------------|
|                 | [B]               | [X] |        |        | [B+, B-]                             | [X+, X-] |             |
| X A, (Note 21)  | 1/1               | 1/3 | 2/3    |        | 1/2                                  | 1/3      |             |
| LD A, (Note 21) | 1/1               | 1/3 | 2/3    | 2/2    | 1/2                                  | 1/3      |             |
| LD B,Imm        |                   |     |        | 1/1    |                                      |          | (If B < 16) |
| LD B,Imm        |                   |     |        | 2/2    |                                      |          | (If B > 15) |
| LD Mem,Imm      | 2/2               |     | 3/3    |        | 2/2                                  |          |             |
| LD Reg,Imm      |                   |     | 2/3    |        |                                      |          |             |
| IFEQ MD,Imm     |                   |     | 3/3    |        |                                      |          |             |

**Note 21:** Memory location addressed by B or X or directly.

# Instruction Set (Continued)

Opcode Table

| UPPER NIBBLE |       |            |          |            |              |             |             |              |             |           |          |               | LOWER NIBBLE  |       |         |  |  |  |  |  |  |  |  |  |  |
|--------------|-------|------------|----------|------------|--------------|-------------|-------------|--------------|-------------|-----------|----------|---------------|---------------|-------|---------|--|--|--|--|--|--|--|--|--|--|
| F            | E     | D          | C        | B          | A            | 9           | 8           | 7            | 6           | 5         | 4        | 3             | 2             | 1     | 0       |  |  |  |  |  |  |  |  |  |  |
| JP-15        | JP-31 | LD 0F0, #i | DRSZ 0F0 | RRCA       | RC           | ADC A, #i   | ADC A, [B]  | IFBIT 0, [B] | ANDSZ A, #i | LD B, #0F | IFBNE 0  | JSR x000-x0FF | JMP x000-x0FF | JP+17 | INTR 0  |  |  |  |  |  |  |  |  |  |  |
| JP-14        | JP-30 | LD 0F1, #i | DRSZ 0F1 | *          | SC           | SUBC A, #i  | SUBC A, [B] | IFBIT 1, [B] | *           | LD B, #0E | IFBNE 1  | JSR x100-x1FF | JMP x100-x1FF | JP+18 | JP+2 1  |  |  |  |  |  |  |  |  |  |  |
| JP-13        | JP-29 | LD 0F2, #i | DRSZ 0F2 | X          | X            | IFEQ A, #i  | IFEQ A, [B] | IFBIT 2, [B] | *           | LD B, #0D | IFBNE 2  | JSR x200-x2FF | JMP x200-x2FF | JP+19 | JP+3 2  |  |  |  |  |  |  |  |  |  |  |
| JP-12        | JP-28 | LD 0F3, #i | DRSZ 0F3 | X          | X            | IFGT A, #i  | IFGT A, [B] | IFBIT 3, [B] | *           | LD B, #0C | IFBNE 3  | JSR x300-x3FF | JMP x300-x3FF | JP+20 | JP+4 3  |  |  |  |  |  |  |  |  |  |  |
| JP-11        | JP-27 | LD 0F4, #i | DRSZ 0F4 | VIS        | LAID         | ADD A, #i   | ADD A, [B]  | IFBIT 4, [B] | CLRA        | LD B, #0B | IFBNE 4  | JSR x400-x4FF | JMP x400-x4FF | JP+21 | JP+5 4  |  |  |  |  |  |  |  |  |  |  |
| JP-10        | JP-26 | LD 0F5, #i | DRSZ 0F5 | RPND       | JID          | AND A, #i   | AND A, [B]  | IFBIT 5, [B] | SWAPA       | LD B, #0A | IFBNE 5  | JSR x500-x5FF | JMP x500-x5FF | JP+22 | JP+6 5  |  |  |  |  |  |  |  |  |  |  |
| JP-9         | JP-25 | LD 0F6, #i | DRSZ 0F6 | X A, [X]   | X            | XOR A, #i   | XOR A, [B]  | IFBIT 6, [B] | DCORA       | LD B, #09 | IFBNE 6  | JSR x600-x6FF | JMP x600-x6FF | JP+23 | JP+7 6  |  |  |  |  |  |  |  |  |  |  |
| JP-8         | JP-24 | LD 0F7, #i | DRSZ 0F7 | *          | *            | OR A, #i    | OR A, [B]   | IFBIT 7, [B] | PUSHA       | LD B, #08 | IFBNE 7  | JSR x700-x7FF | JMP x700-x7FF | JP+24 | JP+8 7  |  |  |  |  |  |  |  |  |  |  |
| JP-7         | JP-23 | LD 0F8, #i | DRSZ 0F8 | NOP        | RLCA         | LD A, #i    | IFC         | SBIT 0, [B]  | RBIT 0, [B] | LD B, #07 | IFBNE 8  | JSR x800-x8FF | JMP x800-x8FF | JP+25 | JP+9 8  |  |  |  |  |  |  |  |  |  |  |
| JP-6         | JP-22 | LD 0F9, #i | DRSZ 0F9 | IFNE       | IFEQ Md, #i  | IFNE A, #i  | IFNC        | SBIT 1, [B]  | RBIT 1, [B] | LD B, #06 | IFBNE 9  | JSR x900-x9FF | JMP x900-x9FF | JP+26 | JP+10 9 |  |  |  |  |  |  |  |  |  |  |
| JP-5         | JP-21 | LD 0FA, #i | DRSZ 0FA | LD A, [X+] | LD A, [B+]   | LD [B+], #i | INCA        | SBIT 2, [B]  | RBIT 2, [B] | LD B, #05 | IFBNE 0A | JSR xA00-xAFF | JMP xA00-xAFF | JP+27 | JP+11 A |  |  |  |  |  |  |  |  |  |  |
| JP-4         | JP-20 | LD 0FB, #i | DRSZ 0FB | LD A, [X-] | LD A, [B-]   | [B-], #i    | DECA        | SBIT 3, [B]  | RBIT 3, [B] | LD B, #04 | IFBNE 0B | JSR xB00-xBFF | JMP xB00-xBFF | JP+28 | JP+12 B |  |  |  |  |  |  |  |  |  |  |
| JP-3         | JP-19 | LD 0FC, #i | DRSZ 0FC | LD Md, #i  | JMPL X A, Md | X A, Md     | POPA        | SBIT 4, [B]  | RBIT 4, [B] | LD B, #03 | IFBNE 0C | JSR xC00-xCFF | JMP xC00-xCFF | JP+29 | JP+13 C |  |  |  |  |  |  |  |  |  |  |
| JP-2         | JP-18 | LD 0FD, #i | DRSZ 0FD | DIR        | JSRL         | LD A, Md    | RETSK       | SBIT 5, [B]  | RBIT 5, [B] | LD B, #02 | IFBNE 0D | JSR xD00-xDFF | JMP xD00-xDFF | JP+30 | JP+14 D |  |  |  |  |  |  |  |  |  |  |
| JP-1         | JP-17 | LD 0FE, #i | DRSZ 0FE | LD A, [X]  | LD A, [B]    | [B], #i     | RET         | SBIT 6, [B]  | RBIT 6, [B] | LD B, #01 | IFBNE 0E | JSR xE00-xEFF | JMP xE00-xEFF | JP+31 | JP+15 E |  |  |  |  |  |  |  |  |  |  |
| JP-0         | JP-16 | LD 0FF, #i | DRSZ 0FF | *          | *            | LD B, #i    | RETI        | SBIT 7, [B]  | RBIT 7, [B] | LD B, #00 | IFBNE 0F | JSR xF00-xFFF | JMP xF00-xFFF | JP+32 | JP+16 F |  |  |  |  |  |  |  |  |  |  |

where,  
i is the immediate data  
Md is a directly addressed memory location  
\* is an unused opcode  
The opcode 60 Hex is also the opcode for IFBIT #i, A

## Development Tools Support

### OVERVIEW

National is engaged with an international community of independent 3rd party vendors who provide hardware and software development tool support. Through National's interaction and guidance, these tools cooperate to form a choice of solutions that fits each developer's needs.

This section provides a summary of the tool and development kits currently available. Up-to-date information, selection guides, free tools, demos, updates, and purchase information can be obtained at our web site at: [www.national.com/cop8](http://www.national.com/cop8).

### SUMMARY OF TOOLS

#### COP8 Evaluation Tools

- **COP8-NSEVAL:** Free Software Evaluation package for Windows. A fully integrated evaluation environment for COP8, including versions of WCOP8 IDE (Integrated Development Environment), COP8-NSASM, COP8-MLSIM, COP8C, DriveWay™ COP8, Manuals, and other COP8 information.
- **COP8-MLSIM:** Free Instruction Level Simulator tool for Windows. For testing and debugging software instructions only (No I/O or interrupt support).
- **COP8-EPU:** Very Low cost COP8 Evaluation & Programming Unit. Windows based evaluation and hardware-simulation tool, with COP8 device programmer and erasable samples. Includes COP8-NSDEV, DriveWay COP8 Demo, MetaLink Debugger, I/O cables and power supply.
- **COP8-EVAL-ICUxx:** Very Low cost evaluation and design test board for COP8ACC and COP8SGx Families, from ICU. Real-time environment with add-on A/D, D/A, and EEPROM. Includes software routines and reference designs.
- **Manuals, Applications Notes, Literature:** Available free from our web site at: [www.national.com/cop8](http://www.national.com/cop8).

#### COP8 Integrated Software/Hardware Design Development Kits

- **COP8-EPU:** Very Low cost Evaluation & Programming Unit. Windows based development and hardware-simulation tool for COPSx/xG families, with COP8 device programmer and samples. Includes COP8-NSDEV, DriveWay COP8 Demo, MetaLink Debugger, cables and power supply.
- **COP8-DM:** Moderate cost Debug Module from MetaLink. A Windows based, real-time in-circuit emulation tool with COP8 device programmer. Includes COP8-NSDEV, DriveWay COP8 Demo, MetaLink Debugger, power supply, emulation cables and adapters.

#### COP8 Development Languages and Environments

- **COP8-NSASM:** Free COP8 Assembler v5 for Win32. Macro assembler, linker, and librarian for COP8 software development. Supports all COP8 devices. (DOS/Win16 v4.10.2 available with limited support). (Compatible with WCOP8 IDE, COP8C, and DriveWay COP8).
- **COP8-NSDEV:** Very low cost Software Development Package for Windows. An integrated development environment for COP8, including WCOP8 IDE, COP8-NSASM, COP8-MLSIM.
- **COP8C:** Moderately priced C Cross-Compiler and Code Development System from Byte Craft (no code limit). In-

cludes BCLIDE (Byte Craft Limited Integrated Development Environment) for Win32, editor, optimizing C Cross-Compiler, macro cross assembler, BC-Linker, and MetaLink tools support. (DOS/SUN versions available; Compiler is installable under WCOP8 IDE; Compatible with DriveWay COP8).

- **EW COP8-KS:** Very Low cost ANSI C-Compiler and Embedded Workbench from IAR (Kickstart version: COP8Sx/Fx only with 2k code limit; No FP). A fully integrated Win32 IDE, ANSI C-Compiler, macro assembler, editor, linker, Librarian, C-Spy simulator/debugger, PLUS MetaLink EPU/DM emulator support.
- **EW COP8-AS:** Moderately priced COP8 Assembler and Embedded Workbench from IAR (no code limit). A fully integrated Win32 IDE, macro assembler, editor, linker, librarian, and C-Spy high-level simulator/debugger with I/O and interrupts support. (Upgradeable with optional C-Compiler and/or MetaLink Debugger/Emulator support).
- **EW COP8-BL:** Moderately priced ANSI C-Compiler and Embedded Workbench from IAR (Baseline version: All COP8 devices; 4k code limit; no FP). A fully integrated Win32 IDE, ANSI C-Compiler, macro assembler, editor, linker, librarian, and C-Spy high-level simulator/debugger. (Upgradeable; CWCOP8-M MetaLink tools interface support optional).
- **EW COP8:** Full featured ANSI C-Compiler and Embedded Workbench for Windows from IAR (no code limit). A fully integrated Win32 IDE, ANSI C-Compiler, macro assembler, editor, linker, librarian, and C-Spy high-level simulator/debugger. (CWCOP8-M MetaLink tools interface support optional).
- **EW COP8-M:** Full featured ANSI C-Compiler and Embedded Workbench for Windows from IAR (no code limit). A fully integrated Win32 IDE, ANSI C-Compiler, macro assembler, editor, linker, librarian, C-Spy high-level simulator/debugger, PLUS MetaLink debugger/hardware interface (CWCOP8-M).

#### COP8 Productivity Enhancement Tools

- **WCOP8 IDE:** Very Low cost IDE (Integrated Development Environment) from KKD. Supports COP8C, COP8-NSASM, COP8-MLSIM, DriveWay COP8, and MetaLink debugger under a common Windows Project Management environment. Code development, debug, and emulation tools can be launched from the project window framework.
- **DriveWay-COP8:** Low cost COP8 Peripherals Code Generation tool from Aisys Corporation. Automatically generates tested and documented C or Assembly source code modules containing I/O drivers and interrupt handlers for each on-chip peripheral. Application specific code can be inserted for customization using the integrated editor. (Compatible with COP8-NSASM, COP8C, and WCOP8 IDE.)
- **COP8-UTILS:** Free set of COP8 assembly code examples, device drivers, and utilities to speed up code development.
- **COP8-MLSIM:** Free Instruction Level Simulator tool for Windows. For testing and debugging software instructions only (No I/O or interrupt support).

## Development Tools Support

(Continued)

### COP8 Real-Time Emulation Tools

- **COP8-DM:** MetaLink Debug Module. A moderately priced real-time in-circuit emulation tool, with COP8 device programmer. Includes COP8-NSDEV, DriveWay COP8 Demo, MetaLink Debugger, power supply, emulation cables and adaptors.
- **IM-COP8:** MetaLink iceMASTER®. A full featured, real-time in-circuit emulator for COP8 devices. Includes MetaLink Windows Debugger, and power supply. Package-specific probes and surface mount adaptors are ordered separately.

### COP8 Device Programmer Support

- MetaLink's EPU and Debug Module include development device programming capability for COP8 devices.
- Third-party programmers and automatic handling equipment cover needs from engineering prototype and pilot production, to full production environments.
- Factory programming available for high-volume requirements.

### TOOLS ORDERING NUMBERS FOR THE COP8ACC7 FAMILY DEVICES

| Vendor                                                                                          | Tools                    | Order Number                                                  | Cost  | Notes                                                                                                                                                    |
|-------------------------------------------------------------------------------------------------|--------------------------|---------------------------------------------------------------|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| National                                                                                        | COP8-NSEVAL              | COP8-NSEVAL                                                   | Free  | Web site download                                                                                                                                        |
|                                                                                                 | COP8-NSASM               | COP8-NSASM                                                    | Free  | Included in EPU and DM. Web site download                                                                                                                |
|                                                                                                 | COP8-MLSIM               | COP8-MLSIM                                                    | Free  | Included in EPU and DM. Web site download                                                                                                                |
|                                                                                                 | COP8-NSDEV               | COP8-NSDEV                                                    | VL    | Included in EPU and DM. Order CD from website                                                                                                            |
|                                                                                                 | COP8-EPU                 | Not available for this device                                 |       |                                                                                                                                                          |
|                                                                                                 | COP8-DM                  | Contact MetaLink                                              |       |                                                                                                                                                          |
|                                                                                                 | Development Devices      | COP8ACC7                                                      | VL    | 16k OTP devices; 20/28 pin.                                                                                                                              |
|                                                                                                 | OTP Programming Adapters | PN# EDI 28D (SO)/40D-Z-COP8LXC                                | L     | For programming 20/28 SOIC and DIP on any programmer.                                                                                                    |
|                                                                                                 | IM-COP8                  | Contact MetaLink                                              |       |                                                                                                                                                          |
| MetaLink                                                                                        | COP8-EPU                 | Not available for this device                                 |       |                                                                                                                                                          |
|                                                                                                 | COP8-DM                  | DM4-COP8-ACx (10 MHz), plus PS-10, plus DM-COP8/xxx (ie. 28D) | M     | Included p/s (PS-10), target cable of choice (DIP or PLCC; i.e. DM-COP8/28D), EDI programming sockets. Add target adapter (if needed)                    |
|                                                                                                 | DM Target Adapters       | MHW-CNV38 or 39                                               | L     | DM target converters for 20SO or 28SO; (i.e. MHW-CNV38 for 20 pin DIP to SO package converter)                                                           |
|                                                                                                 | OTP Programming Adapters | PN# EDI 28D (SO)/40D-Z-COP8LXC                                | L     | For programming 20/28 SOIC and DIP on any programmer.                                                                                                    |
|                                                                                                 | IM-COP8                  | IM-COP8-AD-464 (-220) (10 MHz maximum)                        | H     | Base unit 10 MHz; -220 = 220V; add probe card (required) and target adapter (if needed); included software and manuals                                   |
|                                                                                                 |                          | PC-8AC28DW-AD-10                                              | M     | 10 MHz 20/28 DIP probe card; 2.5V to 5.5V                                                                                                                |
|                                                                                                 | IM Probe Target Adapter  | MHW-SOIC28                                                    | L     | 28 pin SOIC adapter for probe card                                                                                                                       |
| ICU                                                                                             | COP8-EVAL                | COP8-EVAL_ICUAC                                               | L     | No power supply                                                                                                                                          |
| KKD                                                                                             | WCOP8-IDE                | WCOP8-IDE                                                     | VL    | Included in EPU and DM                                                                                                                                   |
| IAR                                                                                             | EWOP8-xx                 | See summary above                                             | L - H | Included all software and manuals                                                                                                                        |
| Byte Craft                                                                                      | COP8C                    | COP8C                                                         | M     | Included all software and manuals                                                                                                                        |
| Aisys                                                                                           | DriveWay COP8            | DriveWay COP8                                                 | L     | Included all software and manuals                                                                                                                        |
| OTP Programmers                                                                                 |                          | Contact vendors                                               | L - H | For approved programmer listings and vendor information, go to our OTP support page at: <a href="http://www.national.com/cop8">www.national.com/cop8</a> |
| Cost: Free; VL =< \$100; L = \$100 - \$300; M = \$300 - \$1k; H = \$1k - \$3k; VH = \$3k - \$5k |                          |                                                               |       |                                                                                                                                                          |

## Development Tools Support (Continued)

### WHERE TO GET TOOLS

Tools are ordered directly from the following vendors. Please go to the vendor's web site for current listings of distributors.

| Vendor     | Home Office                                                      | Electronic Sites                                                                                                                                                                                                                                                | Other Main Offices                                                                                                                                                                         |
|------------|------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Aisys      | U.S.A.: Santa Clara, CA<br>1-408-327-8820<br>fax: 1-408-327-8830 | <a href="http://www.aisysinc.com">www.aisysinc.com</a><br><a href="mailto:info@aisysinc.com">info@aisysinc.com</a>                                                                                                                                              | Distributors                                                                                                                                                                               |
| Byte Craft | U.S.A.<br>1-519-888-6911<br>fax: 1-519-746-6751                  | <a href="http://www.bytecraft.com">www.bytecraft.com</a><br><a href="mailto:info@bytecraft.com">info@bytecraft.com</a>                                                                                                                                          | Distributors                                                                                                                                                                               |
| IAR        | Sweden: Uppsala<br>+46 18 16 78 00<br>fax: +46 18 16 78 38       | <a href="http://www.iar.se">www.iar.se</a><br><a href="mailto:info@iar.se">info@iar.se</a><br><a href="mailto:info@iar.com">info@iar.com</a><br><a href="mailto:info@iarsys.co.uk">info@iarsys.co.uk</a><br><a href="mailto:info@iar.de">info@iar.de</a>        | U.S.A.: San Francisco<br>1-415-765-5500<br>fax: 1-415-765-5503<br>U.K.: London<br>+44 171 924 33 34<br>fax: +44 171 924 53 41<br>Germany: Munich<br>+49 89 470 6022<br>fax: +49 89 470 956 |
| ICU        | Sweden: Polygonvaegen<br>+46 8 630 11 20<br>fax: +46 8 630 11 70 | <a href="http://www.icu.se">www.icu.se</a><br><a href="mailto:support@icu.se">support@icu.se</a><br><a href="mailto:support@icu.ch">support@icu.ch</a>                                                                                                          | Switzerland: Hoehe<br>+41 34 497 28 20<br>fax: +41 34 497 28 21                                                                                                                            |
| KKD        | Denmark:                                                         | <a href="http://www.kkd.dk">www.kkd.dk</a>                                                                                                                                                                                                                      |                                                                                                                                                                                            |
| MetaLink   | U.S.A.: Chandler, AZ<br>1-800-638-2423<br>fax: 1-602-926-1198    | <a href="http://www.metaice.com">www.metaice.com</a><br><a href="mailto:sales@metaice.com">sales@metaice.com</a><br><a href="mailto:support@metaice.com">support@metaice.com</a><br>bbs: 1-602-962-0013<br><a href="http://www.metalink.de">www.metalink.de</a> | Germany: Kirchseeon<br>80-91-5696-0<br>fax: 80-91-2386<br><a href="mailto:islanger@metalink.de">islanger@metalink.de</a><br>Distributors Worldwide                                         |
| National   | U.S.A.: Santa Clara, CA<br>1-800-272-9959<br>fax: 1-800-737-7018 | <a href="http://www.national.com/cop8">www.national.com/cop8</a><br><a href="mailto:support@nsc.com">support@nsc.com</a><br><a href="mailto:europe.support@nsc.com">europe.support@nsc.com</a>                                                                  | Europe: +49 (0) 180 530 8585<br>fax: +49 (0) 180 530 8586<br>Distributors Worldwide                                                                                                        |

The following companies have approved COP8 programmers in a variety of configurations. Contact your local office or distributor. You can link to their web sites and get the latest listing of approved programmers from National's COP8 OTP Support page at: [www.national.com/cop8](http://www.national.com/cop8).

Advantech; Advin; BP Microsystems; Data I/O; Hi-Lo Systems; ICE Technology; Lloyd Research; Logical Devices; MQP; Needhams; Phytos; SMS; Stag Programmers; System General; Tribal Microsystems; Xeltek.

### Customer Support

Complete product information and technical support is available from National's customer response centers, and from our on-line COP8 customer support sites.

**Physical Dimensions** inches (millimeters) unless otherwise noted

