



***Z88C00***

***CMOS Super8 ROMless  
MCU***

**Product Specification**

PS014602-0103



This publication is subject to replacement by a later edition. To determine whether a later edition exists, or to request copies of publications, contact:

**ZiLOG Worldwide Headquarters**

532 Race Street  
San Jose, CA 95126-3432  
Telephone: 408.558.8500  
Fax: 408.558.8300  
[www.ZiLOG.com](http://www.ZiLOG.com)

Windows is a registered trademark of Microsoft Corporation.

**Document Disclaimer**

© 2003 by ZiLOG, Inc. All rights reserved. Information in this publication concerning the devices, applications, or technology described is intended to suggest possible uses and may be superseded. ZILOG, INC. DOES NOT ASSUME LIABILITY FOR OR PROVIDE A REPRESENTATION OF ACCURACY OF THE INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED IN THIS DOCUMENT. ZILOG ALSO DOES NOT ASSUME LIABILITY FOR INTELLECTUAL PROPERTY INFRINGEMENT RELATED IN ANY MANNER TO USE OF INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED HEREIN OR OTHERWISE. Except with the express written approval ZILOG, use of information, devices, or technology as critical components of life support systems is not authorized. No licenses or other rights are conveyed, implicitly or otherwise, by this document under any intellectual property rights.



# *Table of Contents*

|                                  |    |
|----------------------------------|----|
| FEATURES .....                   | 1  |
| GENERAL DESCRIPTION .....        | 1  |
| Protopack .....                  | 4  |
| ARCHITECTURE .....               | 5  |
| PIN DESCRIPTIONS .....           | 6  |
| REGISTERS .....                  | 7  |
| Working Register Window .....    | 8  |
| Register List .....              | 9  |
| MODE AND CONTROL REGISTERS ..... | 12 |
| I/O PORTS .....                  | 17 |
| UART .....                       | 20 |
| Pins .....                       | 20 |
| ADDRESS SPACE .....              | 21 |
| CPU Program Memory .....         | 22 |
| ROMless .....                    | 22 |
| ROM and Protopack .....          | 22 |
| CPU Data Memory .....            | 22 |
| INSTRUCTION SET .....            | 23 |
| Instruction Pointer .....        | 23 |
| Flag Register .....              | 24 |
| Condition Codes .....            | 25 |
| Addressing Modes .....           | 25 |
| INSTRUCTION SUMMARY .....        | 29 |
| SUPER-8 OPCODE MAP .....         | 38 |
| INSTRUCTIONS .....               | 39 |
| INTERRUPTS .....                 | 42 |
| Sources .....                    | 42 |
| Vectors .....                    | 42 |
| Levels .....                     | 42 |
| Enables .....                    | 43 |
| Service Routines .....           | 44 |
| Fast Interrupt Processing .....  | 44 |
| Level or Edge Triggered .....    | 45 |
| STACK OPERATION .....            | 45 |
| User-Defined Stacks .....        | 45 |
| COUNTER/TIMERS .....             | 46 |



DMA ..... 46

ABSOLUTE MAXIMUM RATINGS ..... 47

STANDARD TEST CONDITIONS ..... 47

DC CHARACTERISTICS ..... 48

INPUT HANDSHAKE TIMING ..... 48

AC CHARACTERISTICS (20 MHz) ..... 49

    Input Handshake ..... 49

OUTPUT HANDSHAKE TIMING ..... 50

AC CHARACTERISTICS (12 MHz, 20 MHz) ..... 50

    Output Handshake ..... 50

AC CHARACTERISTICS (12 MHz) ..... 51

    Read /Write ..... 51

AC CHARACTERISTICS (20 MHz) ..... 52

    Read /Write ..... 52

AC CHARACTERISTICS (20 MHz) ..... 54

    EPROM Read Cycle ..... 54

    Example: ..... 54

Packaging Information ..... 54



## *List of Figures*

|                                                         |    |
|---------------------------------------------------------|----|
| Figure 1. Pin Assignments 88-Pin PLCC .....             | 2  |
| Figure 2. Pin Assignments 44-Pin PLCC .....             | 2  |
| Figure 3. Pin Assignments 48-Pin DIP .....              | 3  |
| Figure 4. Pin Functions 48-Pin DIP .....                | 3  |
| Figure 5. Pin Assignments 28-Pin Piggyback Socket ..... | 4  |
| Figure 6. Pin Functions 28-Pin Piggyback Socket .....   | 4  |
| Figure 7. Functional Block Diagram .....                | 5  |
| Figure 8. Super8 Registers .....                        | 8  |
| Figure 9. Working Register Window .....                 | 9  |
| Figure 10. Mode and Control Registers .....             | 12 |
| Figure 11. Mode and Control Registers (Continued) ..... | 13 |
| Figure 12. Mode and Control Registers (Continued) ..... | 14 |
| Figure 13. Mode and Control Registers (Continued) ..... | 15 |
| Figure 14. Mode and Control Registers (Continued) ..... | 16 |
| Figure 15. Mode and Control Registers (Continued) ..... | 17 |
| Figure 16. Program and Data Memory Address Spaces ..... | 23 |
| Figure 17. Instruction Formats .....                    | 28 |
| Figure 18. Instruction Formats (Continued) .....        | 28 |
| Figure 19. Opcode Map .....                             | 38 |
| Figure 20. Interrupt Levels and Vectors .....           | 43 |
| Figure 21. Standard Test Load .....                     | 47 |
| Figure 22. Fully Interlocked Mode .....                 | 48 |
| Figure 23. Strobed Mode .....                           | 49 |
| Figure 24. Fully Interlocked Mode .....                 | 50 |
| Figure 25. Strobed Mode .....                           | 50 |
| Figure 26. External Memory Read and Write Timing .....  | 53 |
| Figure 27. EPROM Read Timing .....                      | 53 |
| Figure 28. 44-Pin PLCC .....                            | 54 |
| Figure 29. 48-Pin DIP .....                             | 55 |



## *List of Tables*

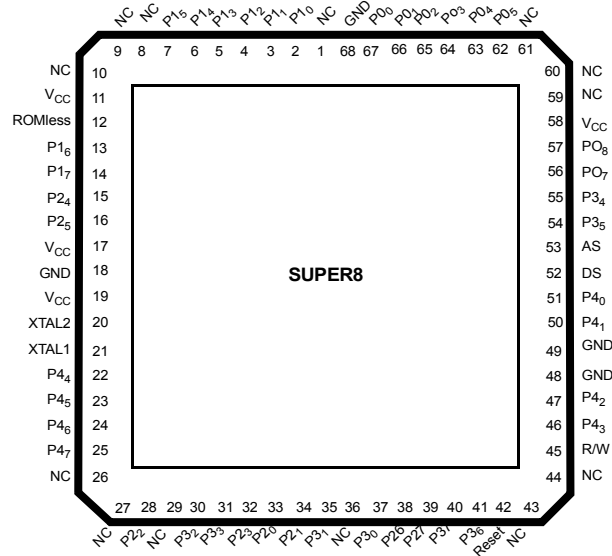
|                                                                      |    |
|----------------------------------------------------------------------|----|
| Table 1. Super-8 Registers .....                                     | 10 |
| Table 2. Port Configuration .....                                    | 18 |
| Table 3. Pin Assignments for Ports 2 and 3 .....                     | 19 |
| Table 4. Condition Codes and Meanings .....                          | 25 |
| Table 5. Instruction Set Notations .....                             | 27 |
| Table 6. Instruction Summary .....                                   | 29 |
| Table 7. Second Nibble .....                                         | 37 |
| Table 8. Super8 Instructions .....                                   | 39 |
| Table 9. DC Characteristics .....                                    | 48 |
| Table 10. AC Characteristics (20 MHz) Input Handshake .....          | 49 |
| Table 11. AC Characteristics (12 MHz, 20 MHz) Output Handshake ..... | 50 |
| Table 12. AC Characteristics (12 MHz) Read/Write .....               | 51 |
| Table 13. AC Characteristics (20 MHz) Read/Write .....               | 52 |
| Table 14. AC Characteristics (20 MHz) EPROM Read Cycle .....         | 54 |

## FEATURES

- Improved Z8® instruction set includes multiply and divide instructions, Boolean and BCD operations.
- Additional instructions support threaded-code languages, such as "Forth."
- 325 byte registers, including 272 general-purpose registers, and 53 mode and control registers.
- Addressing of up to 128K bytes of memory. Two register pointers allow use of short and fast instructions to access register groups within 600 nsec.
- Direct Memory Access controller (DMA).
- Two 16-bit counter/timers.
- Up to 32 bit-programmable and 8 byte-programmable I/O lines, with 2 handshake channels.
- Interrupt structure supports:
  - 27 interrupt sources
  - 16 interrupt vectors (2 reserved for future versions)
  - 8 interrupt levels
  - Servicing in 600 nsec. (1 level only)
- Full-duplex UART with special features.
- On-chip oscillator.
- 20 MHz clock.
- 8K byte ROM for Z8820

## GENERAL DESCRIPTION

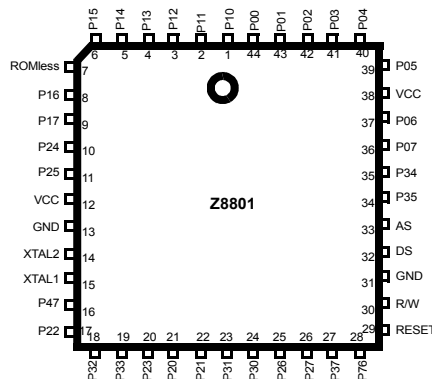
The Zilog Super8 single-chip MCU can be used for development and production. It can be used as I/O- or memory-intensive computers, or configured to address external memory while still supporting many I/O lines.



**Figure 1. Pin Assignments 88-Pin PLCC**

The Super8 features a full-duplex universal asynchronous receiver/ transmitter (UART) with on-chip baud rate generator, two programmable counter/timers, a direct memory access (DMA) controller, and an on-chip oscillator.

The Super8 is also available as a 48-pin and 68-pin ROMless microcomputer with four byte-wide I/O ports plus a byte-wide address/data bus. Additional address bits can be configured, up to a total of 16.



**Figure 2. Pin Assignments 44-Pin PLCC**



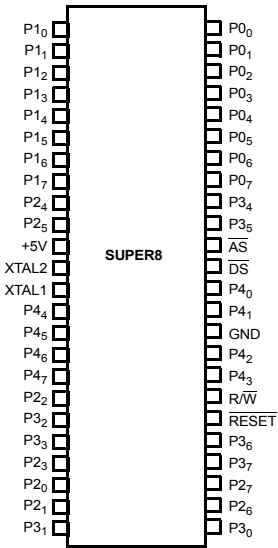


Figure 3.Pin Assignments 48-Pin DIP

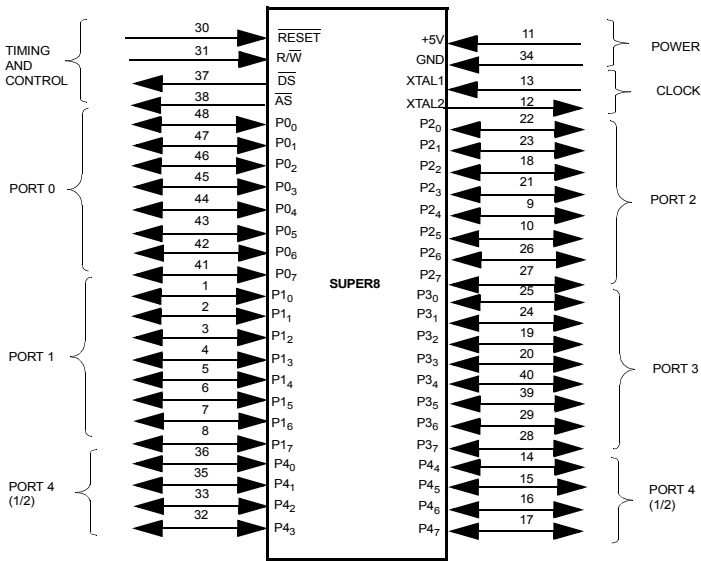


Figure 4.Pin Functions 48-Pin DIP

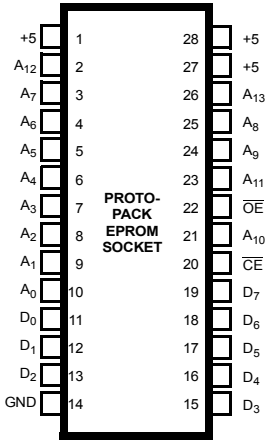


Figure 5.Pin Assignments 28-Pin Piggyback Socket

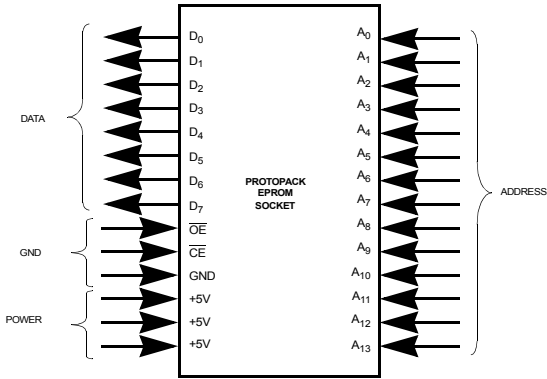


Figure 6.Pin Functions 28-Pin Piggyback Socket

### Protopack

This part functions as an emulator for the basic microcomputer. It uses the same package and pin-out as the basic microcomputer but also has a 28-pin "piggy back" socket on the top into which a ROM or EPROM can be installed. The socket is designed to accept a type 2764 EPROM.

This package permits the protopack to be used in prototype and final PC boards while still permitting user program development. When a final program is devel-

oped, it can be mask-programmed into the production microcomputer device, directly replacing the emulator. The protopack part is also useful in situations where the cost of mask-programming is prohibitive or where program flexibility is desired.

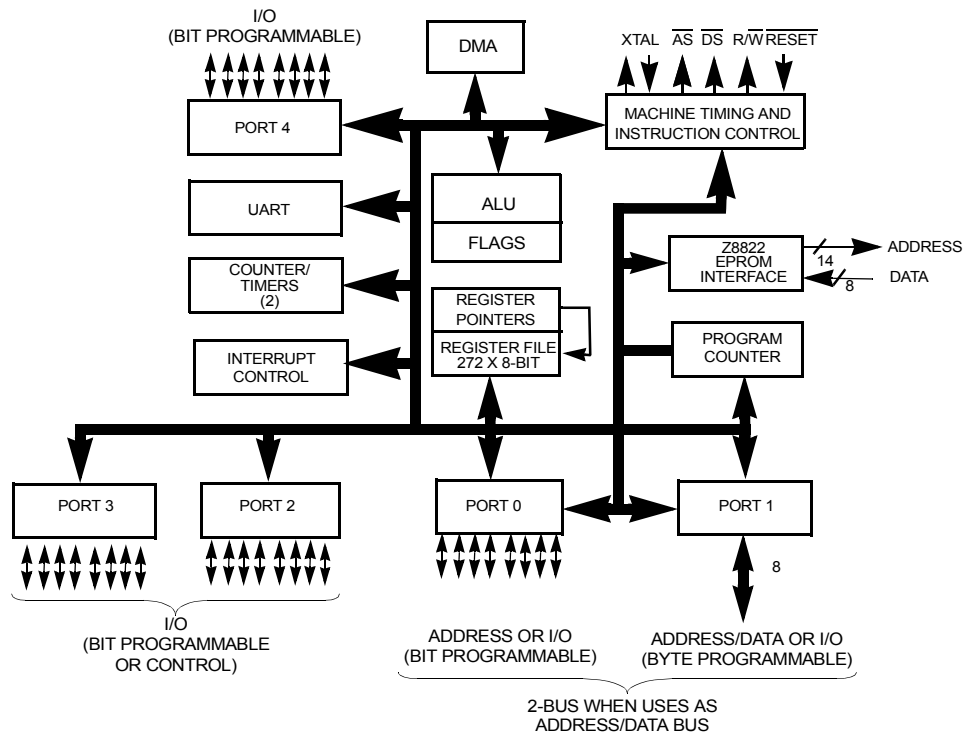


Figure 7. Functional Block Diagram

## ARCHITECTURE

The Super8 architecture includes 325 byte-wide internal registers. 272 of these are available for general purpose use; the remaining 53 provide control and mode functions.

The instruction set is specially designed to deal with this large register set. It includes a full complement of 8-bit arithmetic and logical operations, including multiply and divide instructions and provisions for BCD operations. Addresses and counters can be incremented and decremented as 16-bit quantities. Rotate, shift, and bit manipulation instructions are provided. Three new instructions support threaded-code languages. The UART is a full-function multipurpose asynchronous serial channel with many premium features.

The 16-bit counters can operate independently or be cascaded to perform 32-bit counting and timing operations. The DMA controller handles transfers to and from the register file or memory. DMA can use the UART or one of two ports with handshake capability.

The architecture appears in the block diagram (Figure 7).

## PIN DESCRIPTIONS

The Super8 connects to external devices via the following TTL-compatible pins:

**$\overline{AS}$ .** *Address Strobe* (output, active Low).  $\overline{AS}$  is pulsed Low once at the beginning of each machine cycle. The rising edge indicates that addresses R/W and DM, when used, are valid.

**$\overline{DS}$ .** *Data Strobe* (output, active Low).  $\overline{DS}$  provides timing for data movement between the address/data bus and external memory. During write cycles, data output is valid at the leading edge of  $\overline{DS}$ . During read cycles, data input must be valid prior to the trailing edge of  $\overline{DS}$ .

**P0<sub>0</sub>-P0<sub>7</sub>, P1<sub>0</sub>-P1<sub>7</sub>, P2<sub>0</sub>-P2<sub>7</sub>, P3<sub>0</sub>-P3<sub>7</sub>, P4<sub>0</sub>-P4<sub>7</sub>.** *Port I/O Lines* (input/output).

These 40 lines are divided into five 8-bit I/O ports that can be configured under program control for I/O or external memory interface.

In the ROMless devices, Port 1 is dedicated as a multiplexed address/data port, and Port 0 pins can be assigned as additional address lines; Port 0 non-address pins may be assigned as I/O. In the ROM and protopack, Port 1 can be assigned as input or output, and Port 0 can be assigned as input or output on a bit by bit basis.

Ports 2 and 3 can be assigned on a bit-for-bit basis as general I/O or interrupt lines. They can also be used as special-purpose I/O lines to support the UART, counter/timers, or handshake channels.

Port 4 is used for general I/O.

During reset, all port pins are configured as inputs (high impedance) except for Port 1 and Port 0 in the ROMless devices. In these, Port 1 is configured as a multiplexed address/data bus, and Port 0 pins P0<sub>0</sub>-P0<sub>4</sub> are configured as address out, while pins P0<sub>5</sub>-P0<sub>7</sub> are configured as inputs.

**$\overline{RESET}$ .** *Reset* (input, active Low). Reset initializes and starts the Super8. When it is activated, it halts all processing; when it is deactivated, the Super8 begins processing at address 0020H.

**ROMless.** (input, active High). This input controls the operation mode of a 68-pin Super8. When connected to VCC, the part functions as a ROMless Z8800. When connected to GND, the part functions as a Z8820 ROM part.

**$\overline{R/\overline{W}}$ .** *Read/Write* (output).  $\overline{R/\overline{W}}$  determines the direction of data transfer for external memory transactions. It is Low when writing to program memory or data memory, and High for everything else.

**XTAL1, XTAL2.** (Crystal oscillator input.) These pins connect a parallel resonant crystal or an external clock source to the on-board clock oscillator and buffer.

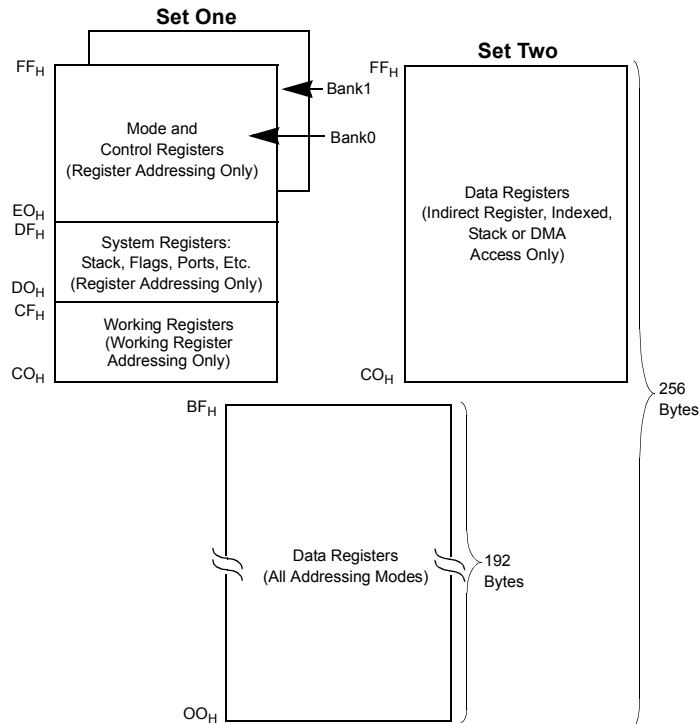
## REGISTERS

The Super8 contains a 256-byte internal register space. However, by using the upper 64 bytes of the register space more than once, a total of 325 registers are available.

Registers from 00 to BF are used only once. They can be accessed by any register command. Register addresses C0 to FF contain two separate sets of 64 registers. One set, called control registers, can only be accessed by register direct commands. The other set can only be addressed by register indirect, indexed, stack, and DMA commands.

The uppermost 32 register direct registers (E0 to FF) are further divided into two banks (0 and 1), selected by the Bank Select bit in the Flag register. When a Register Direct command accesses a register between E0 and FF, it looks at the Bank Select bit in the Flag register to select one of the banks.

The register space is shown in Figure 8.



**Figure 8. Super8 Registers**

## Working Register Window

Control registers R214 and R215 are the register pointers, RPO and RP1. They each define a moveable, 8-register section of the register space. The registers within these spaces are called working registers.

Working registers can be accessed using short 4-bit addresses. The process, shown in section a of Figure 9, works as follows:

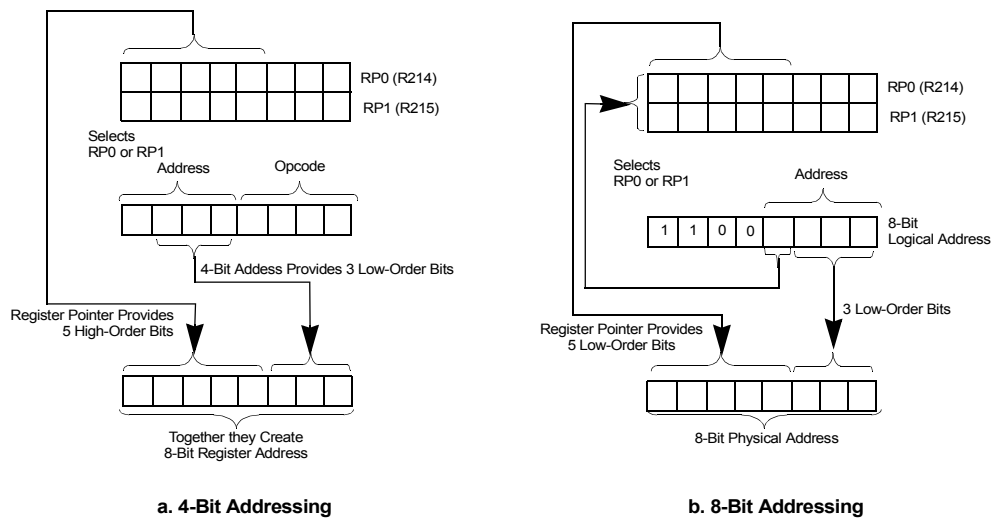
- The high-order bit of the 4-bit address selects one of the two register pointers (0 selects RPO; 1 selects RP1).
- The five high-order bits in the register pointer select an 8-register (contiguous) slice of the register space.
- The three low-order bits of the 4-bit address select one of the eight registers in the slice.

The net effect is to concatenate the five bits from the register pointer to the three bits from the address to form an 8-bit address. As long as the address in the regis-

ter pointer remains unchanged, the three bits from the address always point to an address within the same eight registers.

The register pointers can be moved by changing the five high bits in control registers R214 for RP0 and R215 for RP1.

The working registers can also be accessed by using full 8-bit addressing. When an 8-bit logical address in the range 192 to 207 (CO to CF) is specified, the lower nibble is used similarly to the 4-bit addressing described above. This is shown in section b of Figure 9.



**Figure 9. Working Register Window**

Since any direct access to logical addresses 192 to 207 involves the register pointers, the physical registers 192 to 207 can be accessed only when selected by a register pointer. After a reset, RP0 points to R192 and RP1 points to R200.

## Register List

Super-8 Registers lists the Super8 registers. For more details, see Figure 10.

Table 15.Super-8 Registers

| Address                           |             |        |          |                                     |
|-----------------------------------|-------------|--------|----------|-------------------------------------|
| Decimal                           | Hexadecimal |        | Mnemonic | Function                            |
| <b>General-Purpose Registers</b>  |             |        |          |                                     |
| 000-192                           | 00-BF       |        | -        | General purpose (all address modes) |
| 192-207                           | C0-CF       |        | -        | Working register (direct only)      |
| 192-255                           | C0-FF       |        | -        | General purpose (indirect only)     |
| <b>Mode and Control Registers</b> |             |        |          |                                     |
| 208                               | D0          |        | P0       | Port 0 I/O bits                     |
| 209                               | D1          |        | P1       | Port 1 (I/O only)                   |
| 210                               | D2          |        | P2       | Port 2                              |
| 211                               | D3          |        | P3       | Port 3                              |
| 212                               | D4          |        | P4       | Port 4                              |
| 213                               | D5          |        | FLAGS    | System Flags Register               |
| 214                               | D6          |        | RP0      | Register Pointer 0                  |
| 215                               | D7          |        | RP1      | Register Pointer 1                  |
| 216                               | D8          |        | SPH      | Stack Pointer High Byte             |
| 217                               | D9          |        | SPL      | Stack Pointer Low Byte              |
| 218                               | DA          |        | IPH      | Instruction Pointer High Byte       |
| 219                               | DB          |        | IPL      | Instruction Pointer Low Byte        |
| 220                               | DC          |        | IRQ      | Interrupt Request                   |
| 221                               | DD          |        | IMR      | Interrupt Mask Register             |
| 222                               | DE          |        | SYM      | System Mode                         |
| 224                               | E0          | Bank 0 | C0CT     | CTR 0 Control                       |
|                                   |             | Bank 1 | COM      | CTR 0 Mode                          |
| 225                               | E1          | Bank 0 | C1CT     | CTR 1 Control                       |
|                                   |             | Bank 1 | C1M      | CTR 1 Mode                          |
| 226                               | E2          | Bank 0 | C0CH     | CTR 0 Capture Register, bits 8-15   |
|                                   |             | Bank 1 | CTCH     | CTR 0 Timer Constant, bits 8-15     |
| 227                               | E3          | Bank 0 | C0CL     | CTR 0 Capture Register, bits 0-7    |



Table 15. Super-8 Registers (Continued)

| Address |             |        |          |                                     |
|---------|-------------|--------|----------|-------------------------------------|
| Decimal | Hexadecimal |        | Mnemonic | Function                            |
|         |             | Bank 1 | CTCL     | CTR 0 Time Constant, bits 0-7       |
| 228     | E4          | Bank 0 | C1CH     | CTR 1 Capture Register, bits 8-15   |
|         |             | Bank 1 | C1TCH    | CTR 1 Time Constant, bits 8-15      |
| 229     | E5          | Bank 0 | C1CL     | CTR 1 Capture Register, bits 0-7    |
|         |             | Bank 1 | C1TCL    | CTR 1 Time Constant, bits 0-7       |
| 235     | EB          | Bank 0 | UTC      | UART Transmit Control               |
| 236     | EC          | Bank 0 | URC      | UART Receive Control                |
| 237     | ED          | Bank 0 | UIE      | UART Interrupt Enable               |
| 239     | EF          | Bank 0 | UIO      | UART Data                           |
| 240     | F0          | Bank 0 | POM      | Port 0 Mode                         |
|         |             | Bank 1 | DCH      | DMA Count, bits 8-15                |
| 241     | F1          | Bank 0 | PM       | Port Mode Register                  |
|         |             | Bank 1 | DCL      | DMA Count, bits 0-7                 |
| 244     | F4          | Bank 0 | HOC      | Handshake Channel 0 Control         |
| 245     | F5          | Bank 0 | H1C      | Handshake Channel I Control         |
| 246     | F6          | Bank 0 | P4D      | Port 4 Direction                    |
| 247     | F7          | Bank 0 | P40D     | Port 4 Open Drain                   |
| 248     | F8          | Bank 0 | P2AM     | Port 2/3 A Mode                     |
|         |             | Bank 1 | UBGH     | UART Baud Rate Generator, bits 8-15 |
| 249     | F9          | Bank 0 | P2BM     | Port 2/3 B Mode                     |
|         |             | Bank 1 | UBGL     | UART Baud Rate Generator, bits 0-7  |
| 250     | FA          | Bank 0 | P2CM     | Port 2/3 C Mode                     |
|         |             | Bank 1 | UMA      | UART Mode A                         |
| 251     | FB          | Bank 0 | P2DM     | Port 2/3 D Mode                     |
|         |             | Bank 1 | UMB      | UART Mode B                         |
| 252     | FC          | Bank 0 | P2AIP    | Port 2/3 A Interrupt Pending        |
| 253     | FD          | Bank 0 | P2BIP    | Port 2/3 B Interrupt Pending        |
| 254     | FE          | Bank 0 | EMT      | External Memory Timing              |



Table 15.Super-8 Registers (Continued)

| Address |             | Mnemonic        | Function                    |
|---------|-------------|-----------------|-----------------------------|
| Decimal | Hexadecimal |                 |                             |
| 255     | FF          | Bank 1<br>WUMCH | Wakeup Match Register       |
|         |             | Bank 0<br>IPR   | Interrupt Priority Register |
|         |             | Bank 1<br>WUMSK | Wakeup Match Register       |

MODE AND CONTROL REGISTERS

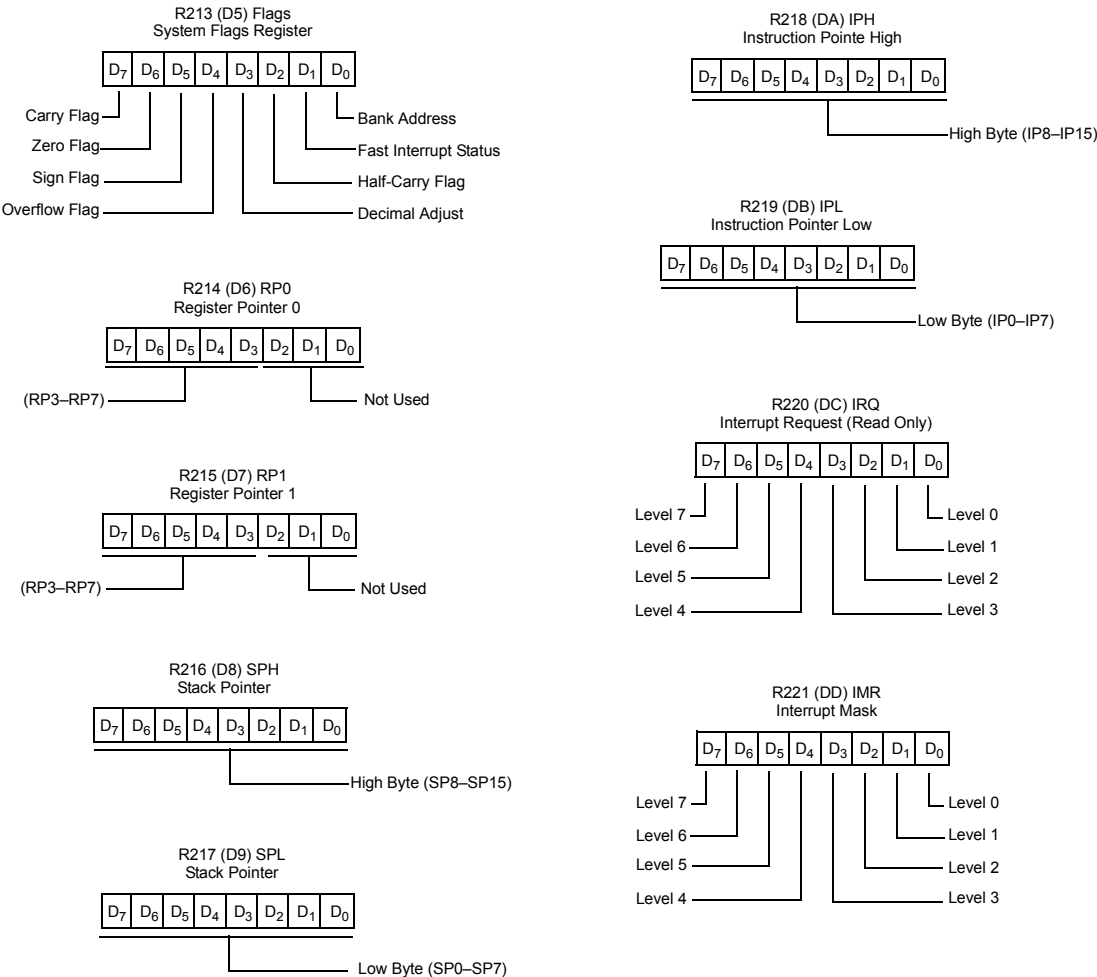


Figure 10.Mode and Control Registers

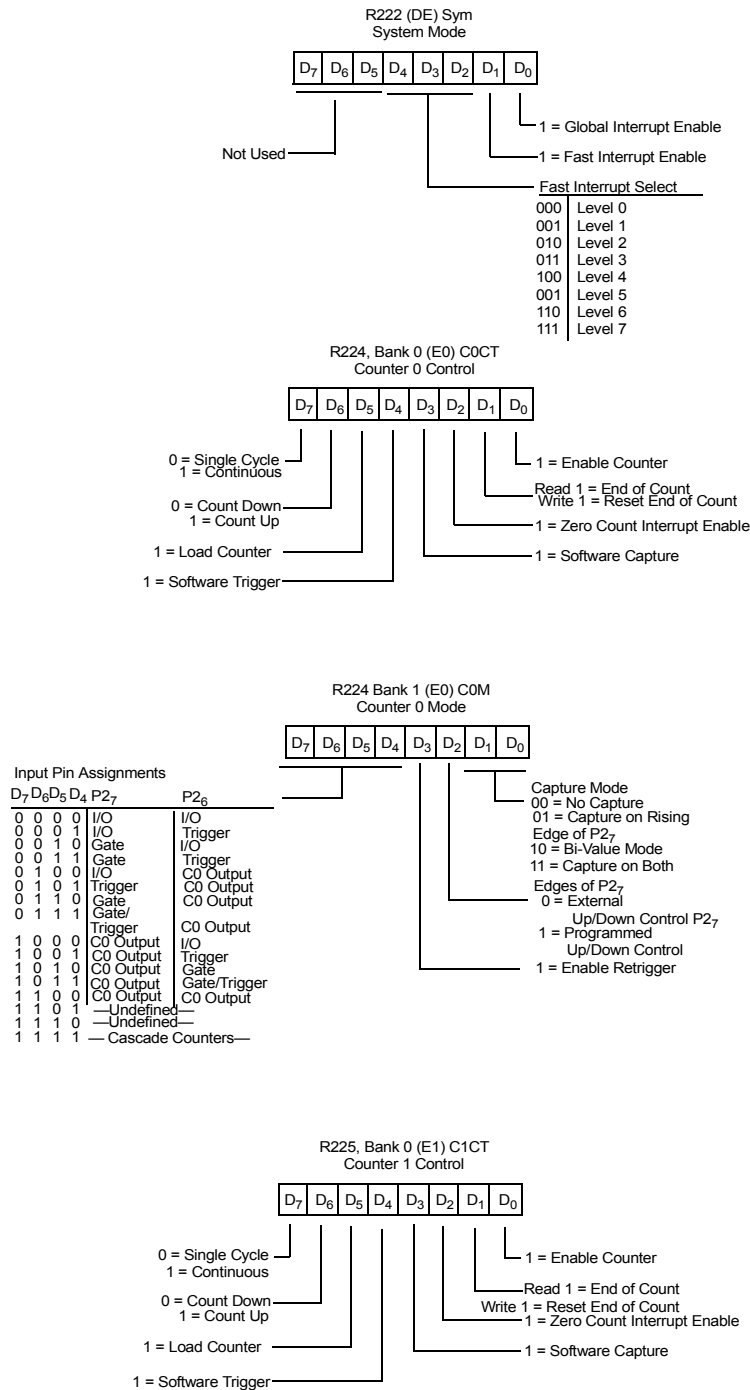


Figure 11.Mode and Control Registers (Continued)

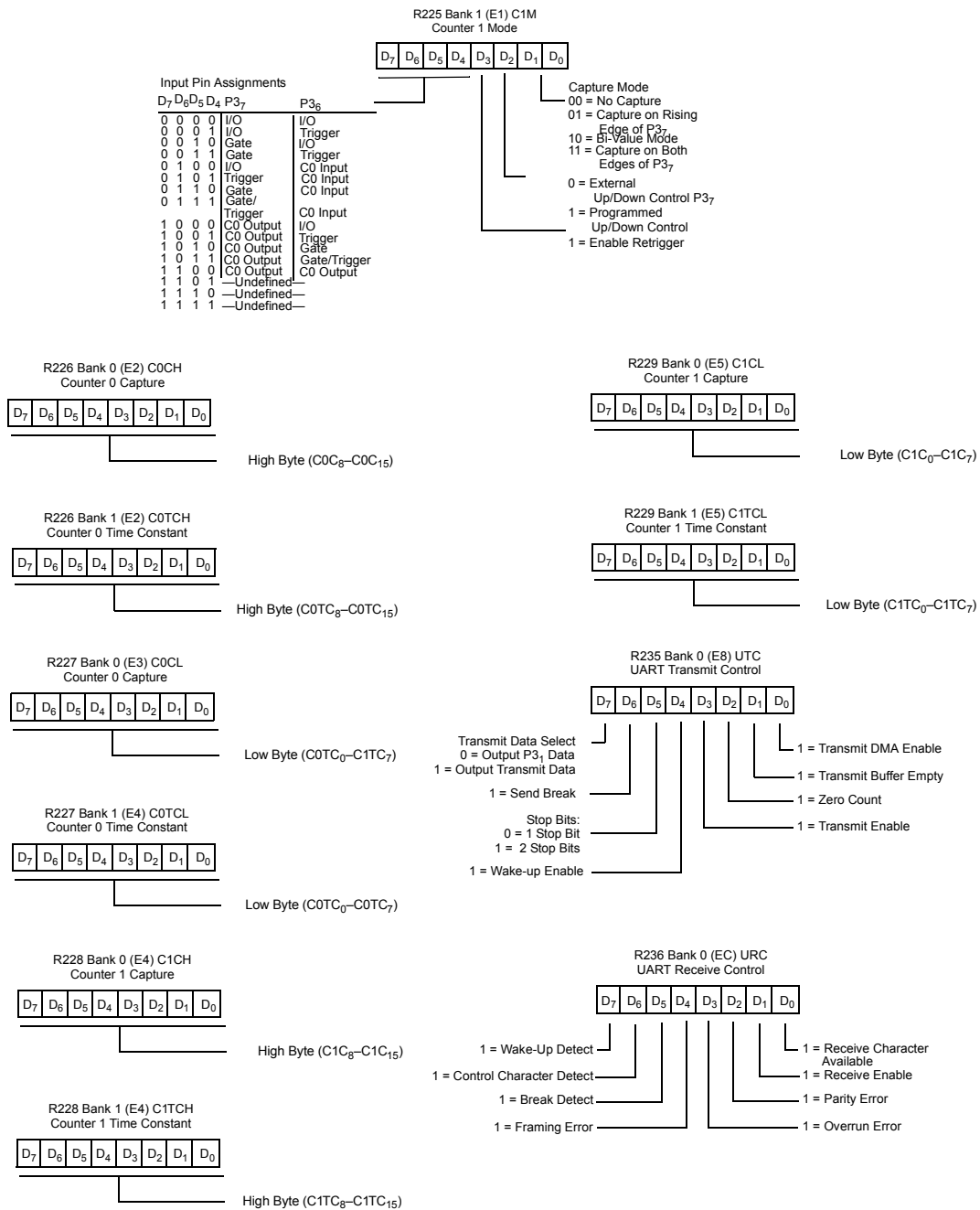


Figure 12.Mode and Control Registers (Continued)

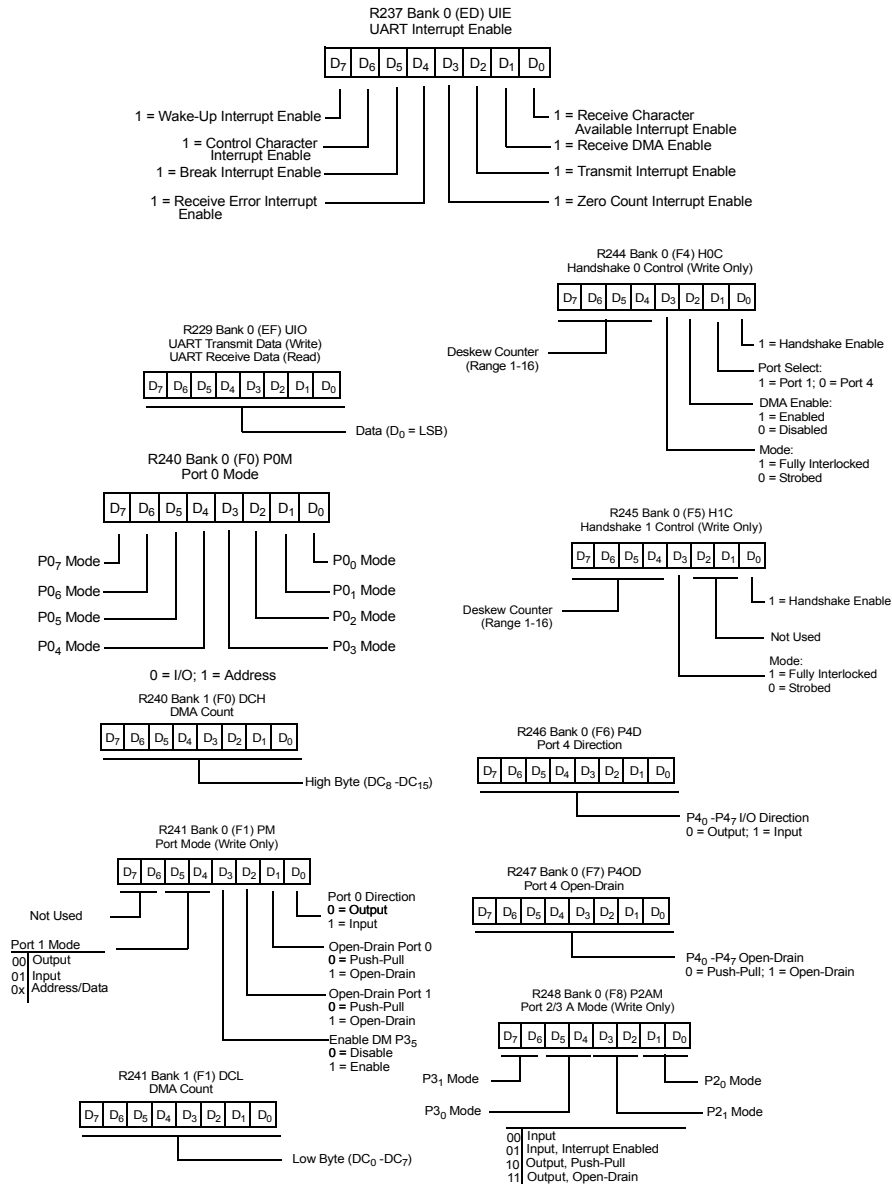


Figure 13. Mode and Control Registers (Continued)

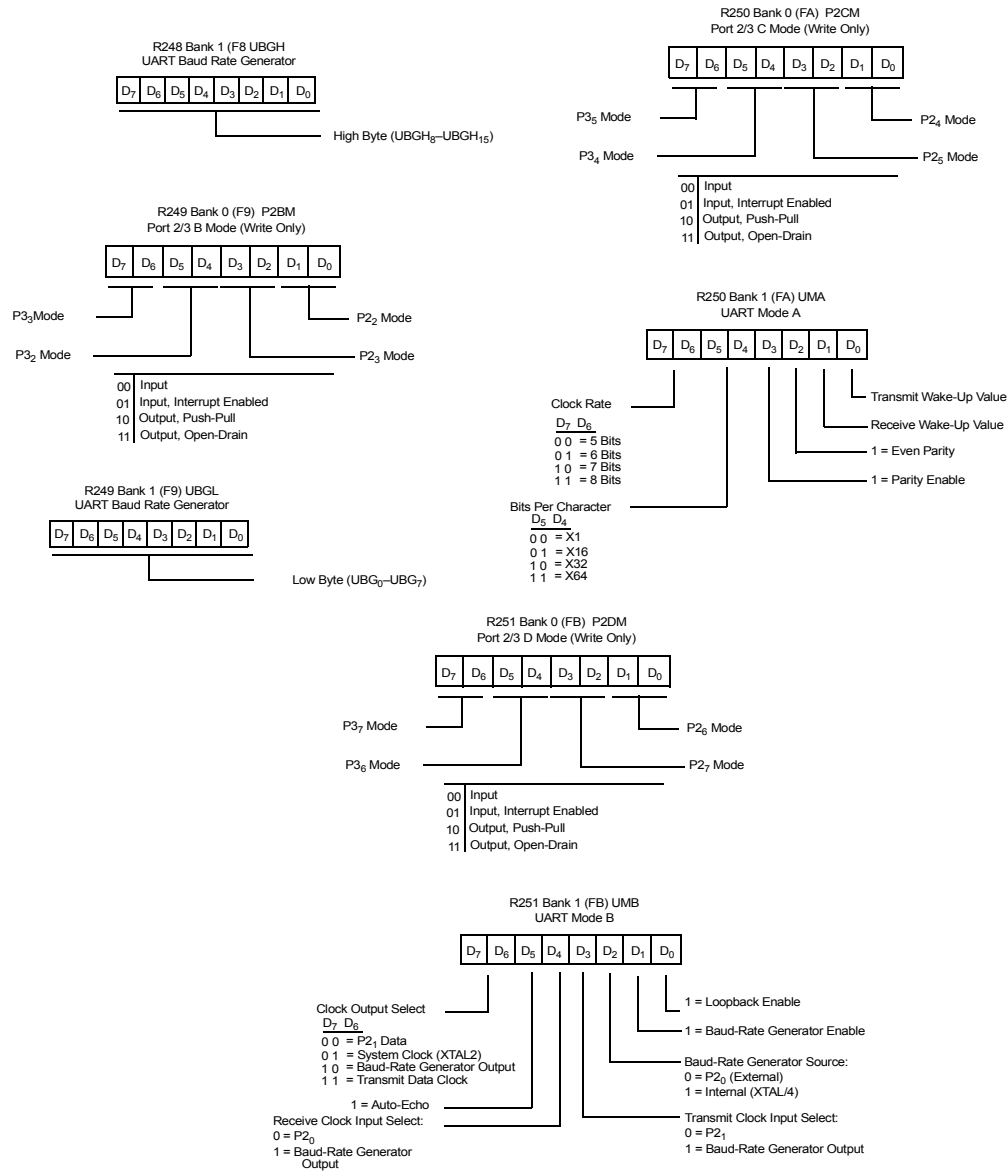


Figure 14. Mode and Control Registers (Continued)

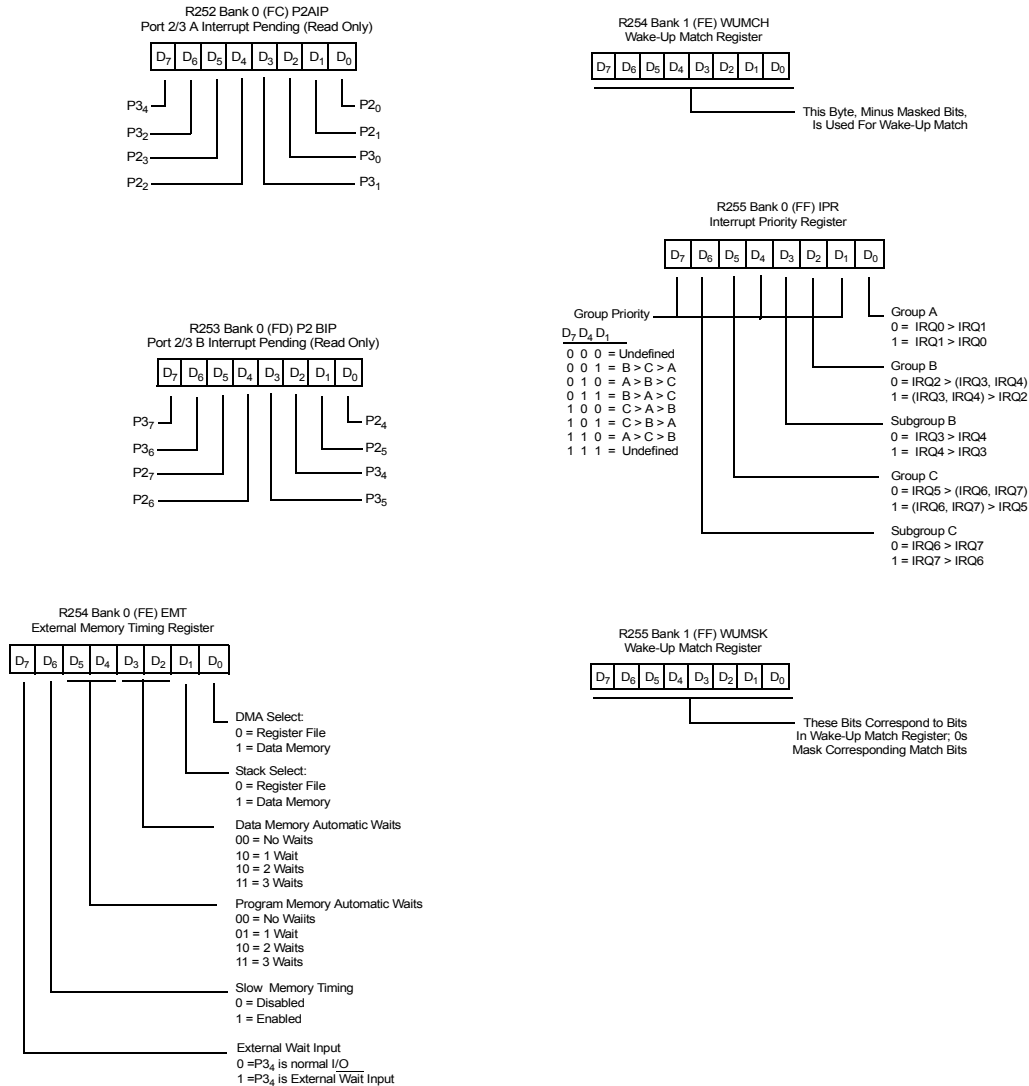


Figure 15. Mode and Control Registers (Continued)

## I/O PORTS

The Super8 has 40 I/O lines arranged into five 8-bit ports. These lines are all TTL-compatible, and can be configured as inputs or outputs. Some can also be configured as address/data lines.

Each port has an input register, an output register, and a register address. Data coming into the port is stored in the input register, and data to be written to a port is stored in the output register. Reading a port's register address returns the value in the input register; writing a port's register address loads the value in the output register. If the port is configured for an output, this value appears on the external pins.

When the CPU reads the bits configured as outputs, the data on the external pins is returned. Under normal output loading, this has the same effect as reading the output register, unless the bits are configured as open-drain outputs.

The ports can be configured as shown in Table 2, Port Configuration.

**Table 16. Port Configuration**

| Port    | Configuration Choices                                                                                  |
|---------|--------------------------------------------------------------------------------------------------------|
| 0       | Address outputs and/or general I/O                                                                     |
| 1       | Multiplexed address/data (or I/O, only for ROM and Protopack)                                          |
| 2 and 3 | Control I/O for UART, handshake channels, and counter/timers; also general I/O and external interrupts |
| 4       | General I/O                                                                                            |

### Port 0

Port 0 can be configured as an I/O port or an output for addressing external memory, or it can be divided and used as both. The bits configured as I/O can be either all outputs or all inputs; they cannot be mixed. If configured for outputs, they can be push-pull or open-drain type.

Any bits configured for I/O can be accessed via R208. To write to the port, specify R208 as the destination (dst) of an instruction; to read the port, specify R208 as the source (src).

Port 0 bits configured as I/O can be placed under handshake control of handshake channel 1.

Port 0 bits configured as address outputs cannot be accessed via the register.

In ROMless devices, initially the four lower bits are configured as address eight through twelve.



## Port 1

In the ROMless device, Port 1 is configured as a byte-wide address/data port. It provides a byte-wide multiplexed address/data path. Additional address lines can be added by configuring Port 0.

The ROM and Protopack Port 1 can be configured as above or as an I/O port; it can be a byte-wide input, open-drain output, or push-pull output. It can be placed under handshake control or handshake channel 0.

## Ports 2 and 3

Ports 2 and 3 provide external control inputs and outputs for the UART, handshake channels, and counter/timers. The pin assignments appear in Table 3.

Bits not used for control I/O can be configured as general-purpose I/O lines and/or external interrupt inputs.

Those bits configured for general I/O can be configured individually for input or output. Those configured for output can be individually configured for open-drain or push-pull output.

All Port 2 and 3 input pins are Schmitt-triggered.

The port address for Port 2 is R210, and for Port 3 is R211.

**Table 17. Pin Assignments for Ports 2 and 3**

| Port 2 |                     | Port 3 |                                             |
|--------|---------------------|--------|---------------------------------------------|
| Bit    | Function            | Bit    | Function                                    |
| 0      | UART receive clock  | 0      | UART receive data                           |
| 1      | UART transmit clock | 1      | UART transmit data                          |
| 2      | Reserved            | 2      | Reserved                                    |
| 3      | Reserved            | 3      | Reserved                                    |
| 4      | Handshake 0 input   | 4      | Handshake 1 input/ $\overline{\text{WAIT}}$ |
| 5      | Handshake 0 output  | 5      | Handshake 1 output/ $\overline{\text{DM}}$  |
| 6      | Counter 0 input     | 6      | Counter 1 input                             |
| 7      | Counter 0 I/O       | 7      | Counter 1 I/O                               |

## Port 4

Port 4 can be configured as I/O only. Each bit can be configured individually as input or output, with either push-pull or open-drain outputs. All Port 4 inputs are Schmitt-triggered.

Port 4 can be placed under handshake control of handshake channel 0. Its register address is R212.

## UART

The UART is a full-duplex asynchronous channel. It transmits and receives independently with 5 to 8 bits per character, has options for even or odd bit parity, and a wake-up feature.

Data can be read into or out of the UART via R239, Bank 0. This single address is able to serve a full-duplex channel because it contains two complete 8-bit registers—one for the transmitter and the other for the receiver.

## Pins

The UART uses the following Port 2 and 3 pins:

| Port/Pin | UART Function  |
|----------|----------------|
| 2/0      | Receive Clock  |
| 3/0      | Receive Data   |
| 2/1      | Transmit Clock |
| 3/1      | Transmit Data  |

### Transmitter

When the UART's register address is specified as the destination (dst) of an operation, the data is output on the UART, which automatically adds the start bit, the programmed parity bit, and the programmed number of stop bits. It can also add a wake-up bit if that option is selected.

If the UART is programmed for a 5-, 6-, or 7-bit character, the extra bits in R239 are ignored.

Serial data is transmitted at a rate equal to 1, 1/16, 1/32 or 1/64 of the transmitter clock rate, depending on the programmed data rate. All data is sent out on the falling edge of the clock input.

When the UART has no data to send, it holds the output marking (High). It may be programmed with the Send Break command to hold the output Low (Spacing), which it continues until the command is cleared.

### Receiver

The UART begins receive operation when Receive Enable (URC, bit 0) is set High. After this, a Low on the receive input pin for longer than half a bit time is interpreted as a start bit. The UART samples the data on the input pin in the middle of each clock cycle until a complete byte is assembled. This is placed in the Receive Data register.

If the 1 X clock mode is selected, external bit synchronization must be provided, and the input data is sampled on the rising edge of the clock.

For character lengths of less than eight bits, the UART inserts ones into the unused bits, and, if parity is enabled, the parity bit is not stripped. The data bits, extra ones, and the parity bit are placed in the UART Data register (UIO).

While the UART is assembling a byte in its input shift register, the CPU has time to service an interrupt and manipulate the data character in UIO.

Once a complete character is assembled, the UART checks it and performs the following:

- If it is an-ASCII control character, the UART sets the Control Character status bit.
- It checks the wake-up settings and completes any indicated action.
- If parity is enabled, the UART checks to see if the calculated parity matches the programmed parity bit. If they do not match, it sets the Parity Error bit in URC (R236 Bank 0), which remains set until reset by software.
- It sets the Framing Error bit (URC, bit 4) if the character is assembled without any stop bits. This bit remains set until cleared by software.

Overrun errors occur when characters are received faster than they are read. That is, when the UART has assembled a complete character before the CPU has read the current character, the UART sets the Overrun Error bit (URC, bit 3), and the character currently in the receive buffer is lost.

The overrun bit remains set until cleared by software.

## ADDRESS SPACE

The Super8 can access 64K bytes of program memory and 64K bytes of data memory. These spaces can be either combined or separate. If separate, they are

controlled by the  $\overline{DM}$  line (Port P3<sub>5</sub>), which selects data memory when Low and program memory when High.

Figure 16 on page 23 shows the system memory space.

## CPU Program Memory

Program memory occupies addresses 0 to 64K. External program memory, if present, is accessed by configuring Ports 0 and 1 as a memory interface.

The address/data lines are controlled by  $\overline{AS}$ ,  $\overline{DS}$  and R/ $\overline{W}$ .

The first 32 program memory bytes are reserved for interrupt vectors; the lowest address available for user programs is 32 (decimal). This value is automatically loaded into the program counter after a hardware reset.

## ROMless

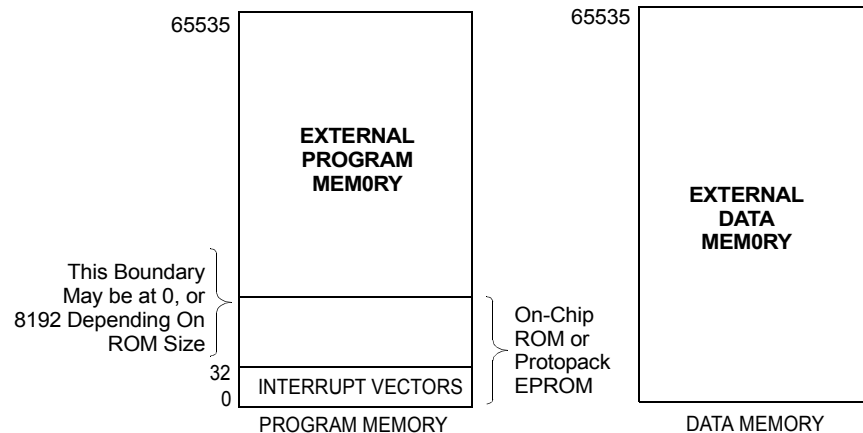
Port 0 can be configured to provide from 0 to 8 additional address lines. Port 1 is always used as an 8-bit multiplexed address/data port.

## ROM and Protopack

Port 1 is configured as multiplexed address/data or as I/O. When Port 1 is configured as address/data, Port 0 lines can be used as additional address lines, up to address 15. External program memory is mapped above internal program memory; that is, external program memory can occupy any space beginning at the top of the internal ROM space up to the 64K (16-bit address) limit.

## CPU Data Memory

The external CPU data memory space, if separated from program memory by the  $\overline{DM}$  optional output, can be mapped anywhere from 0 to 64K (full 16-bit address space). Data memory uses the same address/data bus (Port 1) and additional addresses (chosen from Port 0) as program memory. Data memory is distinguished from program memory by the DM pin (P3<sub>5</sub>), and by the fact that data memory can begin at address 0000H. This feature differs from the Z8.



**Figure 16. Program and Data Memory Address Spaces**

## INSTRUCTION SET

The Super8 instruction set is designed to handle its large register set. The instruction set provides a full complement of 8-bit arithmetic and logical operations, including multiply and divide. It supports BCD operations using a decimal adjustment of binary values, and it supports incrementing and decrementing 16-bit quantities for addresses and counters.

It provides extensive bit manipulation, and rotate and shift operations, and it requires no special I/O instructions-the I/O ports are mapped into the register file.

### Instruction Pointer

A special register called the Instruction Pointer (IP) provides hardware support for threaded-code languages. It consists of register-pair R218 and R219, and it contains, memory addresses. The MSB is R218.

Threaded-code languages deal with an imaginary higher-level machine within the existing hardware machine. The IP acts like the PC for that machine. The command NEXT passes control to or from the hardware machine to the imaginary machine, and the commands ENTER and EXIT are imaginary machine equivalents of (real machine) CALLS and RETURNS.

If the commands NEXT, ENTER, and EXIT are not used, the IP can be used by the fast interrupt processing, as described in the Interrupts section.

## Flag Register

The Flag register (FLAGS) contains eight bits that describe the current status of the Super8. Four of these can be tested and used with conditional jump instructions; two others are used for BCD- arithmetic. FLAGS also contains the Bank Address bit and the Fast Interrupt Status bit.

The flag bits can be set and reset by instructions.



**Caution:** Do not specify FLAGS as the destination of an instruction that normally affects the flag bits or the result is unspecified.

The following paragraphs describe each flag bit:

**Bank Address.** This bit is used to select one of the register banks (0 or 1) between (decimal) addresses 224 and 255. It is cleared by the SBO instruction and set by the SB1 instruction.

**Fast Interrupt Status.** This bit is set during a fast interrupt cycle and reset during the IRET following interrupt servicing. When set, this bit inhibits all interrupts and causes the fast interrupt return to be executed when the IRET instruction is fetched.

**Half-Carry.** This bit is set to 1 whenever an addition generates a carry out of bit 3, or when a subtraction borrows out of bit 4. This bit is used by the Decimal Adjust (DA) instruction to convert the binary result of a previous addition or subtraction into the correct decimal (BCD) result. This flag, and the Decimal Adjust flag, are not usually accessed by users.

**Decimal Adjust.** This bit is used to specify what type of instruction was executed last during BCD operations, so a subsequent Decimal Adjust operation can function correctly. This bit is not usually accessible to programmers, and cannot be used as a test condition.

**Overflow Flag.** This flag is set to 1 when the result of a twos-complement operation was greater than 127 or less than -128. It is also cleared to 0 during logical operations.

**Sign Flag.** Following arithmetic, logical, rotate, or shift operations, this bit identifies the state of the MSB of the result. A 0 indicates a positive number and a 1 indicates a negative number.

**Zero Flag.** For arithmetic and logical operations, this flag is set to 1 if the result of the operation is zero.

For operations that test bits in a register, the zero bit is set to 1 if the result is zero.

For rotate and, shift operations, this bit is set to 1 if the result is zero.

**Carry Flag.** This flag is set to 1 if the result from an arithmetic operation generates a carry out of, or a borrow into, bit 7.

After rotate and shift operations, it contains the last value shifted out of the specified register.

It can be set, cleared, or complemented by instructions.

## Condition Codes

The flags C, Z, S, and V are used to control the operation of conditional jump instructions.

The opcode of a conditional jump contains a 4-bit field called the condition code (cc). This specifies under which conditions it is to execute the jump. For example, a conditional jump with the condition code for "equal" after a compare operation only jumps if the two operands are equal.

The condition codes and their meanings are given in Condition Codes and Meanings.

## Addressing Modes

All operands except for immediate data and condition codes are expressed as register addresses, program memory addresses, or data memory addresses. The addressing modes and their designations are:

- Register (R)
- Indirect Register (IR)
- Indexed (X)
- Direct (DA)
- Relative (RA)
- Immediate (IM)
- Indirect (IA)

**Table 18. Condition Codes and Meanings**

| Binary            | Mnemonic | Flags | Meaning      |
|-------------------|----------|-------|--------------|
| 0000              | F        | -     | Always false |
| 1000              | -        | -     | Always true  |
| 0111 <sup>1</sup> | C        | C = 1 | Carry        |
| 1111 <sup>1</sup> | NC       | C = 0 | No carry     |

Table 18. Condition Codes and Meanings

| Binary            | Mnemonic | Flags                | Meaning                        |
|-------------------|----------|----------------------|--------------------------------|
| 0110 <sup>1</sup> | Z        | Z = 1                | Zero                           |
| 1110 <sup>1</sup> | NZ       | Z = 0                | Not zero                       |
| 1101              | PL       | S = 0                | Plus                           |
| 0101              | MI       | S = 1                | Minus                          |
| 0100              | OV       | V = 1                | Overflow                       |
| 1100              | NOV      | V = 0                | No overflow                    |
| 0110 <sup>1</sup> | EQ       | Z = 1                | Equal                          |
| 1110 <sup>1</sup> | NE       | Z = 0                | Not equal                      |
| 1001              | GE       | (S XOR V)= 0         | Greater than or equal          |
| 0001              | LT       | (S XOR V)= 1         | Less than                      |
| 1010              | GT       | (Z OR (S XOR V))= 0  | Greater than                   |
| 0010              | LE       | (Z OR (S XOR V))= 1  | Less than or equal             |
| 1111 <sup>1</sup> | UGE      | C=0                  | Unsigned greater than or equal |
| 0111 <sup>1</sup> | ULT      | C=1                  | Unsigned less than             |
| 1011              | UGT      | (C = 0 AND Z = 0)= 1 | Unsigned greater than          |
| 0011              | ULE      | (C OR Z)= 1          | Unsigned less than or equal    |

1. Has condition codes that relate to two different mnemonics but test the same flags. For example, Z and EQ are both True if the Zero flag is set, but after an ADD instruction, Z would probably be used, while after a CP instruction, EQ would probably be used.

Registers can be addressed by an 8-bit address in the range of 0 to 255. Working, registers can also be addressed using 4-bit addresses, where five bits contained in a register pointer (R218 or R219) are concatenated with three bits from the 4-bit address to form an 8-bit address.

Registers can be used in pairs to generate 16-bit program or data memory addresses.

### Notation and Encoding

The instruction set notations are described in Table 5.



## Functional Summary of Commands

Figure 17 shows the formats followed by a quick reference guide to the commands.

**Table 19. Instruction Set Notations**

| Notation | Meaning                                                                                   | Notation | Meaning                                 |
|----------|-------------------------------------------------------------------------------------------|----------|-----------------------------------------|
| cc       | Condition code (see Table 4)                                                              |          | DA Direct address (between 0 and 65535) |
| r        | Working register (between 0 and 15)                                                       |          | RA Relative address                     |
| rb       | Bit of working register                                                                   | IM       | Immediate                               |
| r0       | Bit 0 of working register                                                                 | IML      | Immediate long                          |
| R        | Register or working register                                                              |          | dst Destination operand                 |
| RR       | Register pair or working register pair (Register always start on an even-number boundary) | pairs    | src Source operand                      |
|          |                                                                                           | @        | Indirect                                |
| IA       | Indirect address                                                                          | SP       | Stack                                   |
| Ir       | Indirect working register                                                                 | PC       | Program                                 |
| IR       | Indirect register or indirect working register                                            |          | IP                                      |
| Irr      | Indirect working register pair                                                            | FLAGS    | Flags                                   |
| IRR      | Indirect register pair or indirect working register                                       | pair     | RP                                      |
| X        | Indexed                                                                                   | #        | Immediate                               |
| XS       | Indexed, short offset                                                                     | %        | Hexadecimal                             |
| XL       | Indexed, long offset                                                                      | OPC      | Opcode                                  |

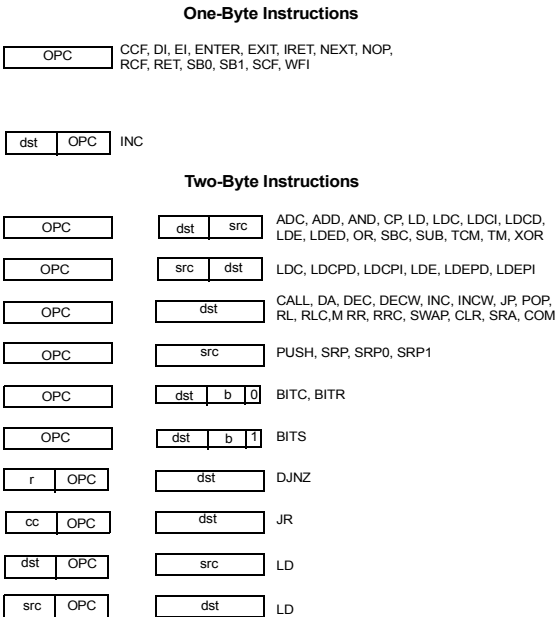


Figure 17.Instruction Formats

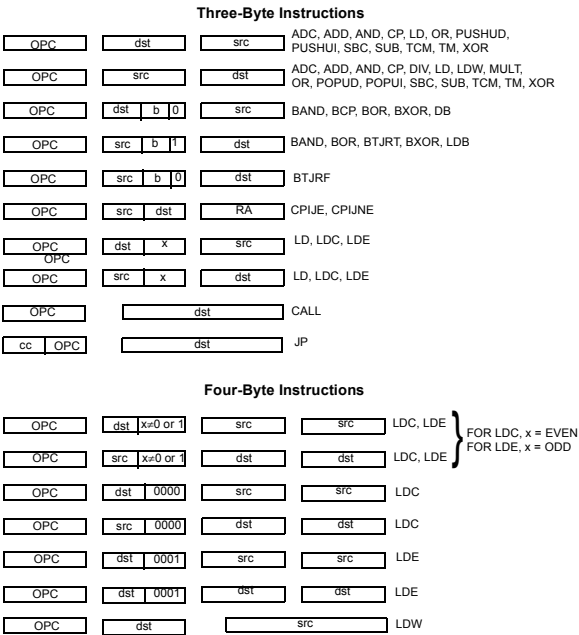


Figure 18.Instruction Formats (Continued)

## INSTRUCTION SUMMARY

Table 20. Instruction Summary

| Instruction<br>and Operation               | Address Mode      |     | Opcode<br>Byte (Hex) | Flags Affected |   |   |   |   |   |
|--------------------------------------------|-------------------|-----|----------------------|----------------|---|---|---|---|---|
|                                            | dst               | src |                      | C              | Z | S | V | D | H |
| <b>ADC</b> dst, src<br>dst ← dst + src + C | Note <sup>1</sup> |     | 1[]                  | *              | * | * | - | 0 | * |
| <b>ADD</b> dst, src<br>dst ← dst + src     | Note <sup>1</sup> |     | 0[]                  | *              | * | * | * | 0 | * |
| <b>AND</b> dst, src<br>dst ← dst AND src   | Note <sup>1</sup> |     | 5[]                  | -              | * | * | 0 | - | - |
| <b>BAND</b> dst, src<br>dst ← dst AND src  | r0                | Rb  | 67                   | -              | * | 0 | U | - | - |
|                                            | Rb                | r0  | 67                   |                |   |   |   |   |   |
| <b>BCP</b> dst, src<br>dst – src           | r0                | Rb  | 17                   | -              | * | 0 | U | - | - |
| <b>BITC</b> dst<br>dst ← NOT dst           | rb                |     | 57                   | -              | * | 0 | U | - | - |
| <b>BITR</b> dst<br>dst ← 0                 | rb                |     | 77                   | -              | - | - | - | - | - |
| <b>BITS</b> dst<br>dst ← 1                 | rb                |     | 77                   | -              | - | - | - | - | - |
| <b>BOR</b> dst, src<br>dst ← dst OR src    | r0                | Rb  | 07                   | -              | * | 0 | U | - | - |
|                                            | Rb                | r0  |                      |                |   |   |   |   |   |
| <b>BTJRF</b><br>If src = 0, PC = PC + dst  | RA                | rB  | 37                   | -              | - | - | - | - | - |
| <b>BTJRT</b><br>If src = '1, PC = PC + dst | RA                | rB  | 37                   | -              | - | - | - | - | - |
| <b>BXOR</b> dst, src<br>dst ← dst XOR src  | r0                | Rb  | 27                   | -              | * | 0 | U | - | - |
|                                            | Rb                | r0  | 27                   |                |   |   |   |   |   |
| <b>CALL</b> dst                            | DA                |     | F6                   | -              | - | - | - | - | - |

Table 20. Instruction Summary (Continued)

| Instruction<br>and Operation                                          | Address Mode      |     | Opcode<br>Byte (Hex) | Flags Affected |   |   |   |   |   |
|-----------------------------------------------------------------------|-------------------|-----|----------------------|----------------|---|---|---|---|---|
|                                                                       | dst               | src |                      | C              | Z | S | V | D | H |
| SP←SP-2                                                               | IRR               |     | F4                   |                |   |   |   |   |   |
| @SP←PC,<br>PC←dst                                                     | IA                |     | D4                   |                |   |   |   |   |   |
| <b>CCF</b><br>C = NOT C                                               |                   |     | EF                   | *              | - | - | - | - | - |
| <b>CLR</b> dst                                                        | R                 |     | B0                   | -              | - | - | - | - | - |
| dst←0                                                                 | IR                |     | B1                   |                |   |   |   |   |   |
| <b>COM</b> dst                                                        | R                 |     | 60                   | -              | * | * | 0 | - | - |
| dst←NOT dst                                                           | IR                |     | 61                   |                |   |   |   |   |   |
| <b>CP</b> dst, src<br>dst - src                                       | Note <sup>1</sup> |     | A[ ]                 | *              | * | * | * | - | - |
| <b>CPIJE</b><br>If dst – src = 0,<br>then<br>PC←PC + RA<br>Ir←Ir + 1  | r                 | Ir  | C2                   | -              | - | - | - | - | - |
| <b>CPIJNE</b><br>If dst – src = 0,<br>then<br>PC←PC + RA<br>Ir←Ir + 1 | r                 | Ir  | D2                   | -              | - | - | - | - | - |
| <b>DA</b> dst                                                         | R                 |     | 40                   | *              | * | * | U | - | - |
| dst←DA dst                                                            | IR                |     | 41                   |                |   |   |   |   |   |
| <b>DEC</b> dst                                                        | R                 |     | 00                   | -              | * | * | * | - | - |
| dst←dst -1                                                            | IR                |     | 01                   |                |   |   |   |   |   |
| <b>DECW</b> dst                                                       | RR                |     | 80                   | -              | * | * | * | - | - |
| dst←dst-1                                                             | IR                |     | 81                   |                |   |   |   |   |   |
| <b>DI</b>                                                             |                   |     | 8F                   | -              | - | - | - | - | - |

Table 20. Instruction Summary (Continued)

| Instruction<br>and Operation | Address Mode |     | Opcode<br>Byte (Hex) | Flags Affected |   |   |   |   |   |
|------------------------------|--------------|-----|----------------------|----------------|---|---|---|---|---|
|                              | dst          | src |                      | C              | Z | S | V | D | H |
| SMR(0)←0                     |              |     |                      |                |   |   |   |   |   |
| DIV dst, src                 |              |     |                      |                |   |   |   |   |   |
| dst ÷ src                    | RR           | R   | 94                   | *              | * | * | * | - | - |
| dst<br>(Upper)←Quotient      | RR           | IR  | 95                   |                |   |   |   |   |   |
| dst<br>(Lower)←Remainder     | RR           | IM  | 96                   |                |   |   |   |   |   |
| DJNZ r, dst                  | RA           | r   | rA                   | -              | - | - | - | - | - |
| r←r - 1<br>(r = 0 to F)      |              |     |                      |                |   |   |   |   |   |
| if r = 0                     |              |     |                      |                |   |   |   |   |   |
| PC←PC + dst                  |              |     |                      |                |   |   |   |   |   |
| EI                           |              |     | 9F                   | -              | - | - | - | - | - |
| SMR(0)←1                     |              |     |                      |                |   |   |   |   |   |
| ENTER                        |              |     | 1F                   | -              | - | - | - | - | - |
| SP←SP - 2                    |              |     |                      |                |   |   |   |   |   |
| @SP←IP                       |              |     |                      |                |   |   |   |   |   |
| IP←PC                        |              |     |                      |                |   |   |   |   |   |
| PC←@IP                       |              |     |                      |                |   |   |   |   |   |
| IP←IP + 2                    |              |     |                      |                |   |   |   |   |   |
| EXIT                         |              |     | 2F                   | -              | - | - | - | - | - |
| IP←@SP                       |              |     |                      |                |   |   |   |   |   |
| SP←SP + 2                    |              |     |                      |                |   |   |   |   |   |
| PC←@IP                       |              |     |                      |                |   |   |   |   |   |
| IP←IP + 2                    |              |     |                      |                |   |   |   |   |   |
| INC dst                      | r            |     | rE                   | -              | * | * | * | - | - |
| dst←dst + 1<br>(r = 0 to F)  |              |     |                      |                |   |   |   |   |   |
|                              | R            |     | 20                   |                |   |   |   |   |   |
|                              | IR           |     | 21                   |                |   |   |   |   |   |

Table 20. Instruction Summary (Continued)

| Instruction<br>and Operation | Address Mode |     | Opcode<br>Byte (Hex) | Flags Affected               |   |   |   |   |   |
|------------------------------|--------------|-----|----------------------|------------------------------|---|---|---|---|---|
|                              | dst          | src |                      | C                            | Z | S | V | D | H |
| <b>INCW</b> dst              | RR           |     | A0                   | -                            | * | * | * | - | - |
| dst ← 1 + dst                | IR           |     | A1                   |                              |   |   |   |   |   |
| <b>IRET</b> (Fast)           |              |     | BF                   | Restored to before interrupt |   |   |   |   |   |
| PC ← IP                      |              |     |                      |                              |   |   |   |   |   |
| FLAG ← FLAG'                 |              |     |                      |                              |   |   |   |   |   |
| FIS ← 0                      |              |     |                      |                              |   |   |   |   |   |
| <b>IRET</b> (Normal)         |              |     | BF                   | Restored to before interrupt |   |   |   |   |   |
| FLAGS ← @SP; SP ← SP + 1     |              |     |                      |                              |   |   |   |   |   |
| PC ← @SP; SP ← SP + 2;       |              |     |                      |                              |   |   |   |   |   |
| SMR(0) ← 1                   |              |     |                      |                              |   |   |   |   |   |
| <b>JP</b> cc, dst            | DA           |     | ccD                  | -                            | - | - | - | - | - |
| if cc is true,               |              |     | (cc = 0 to F)        |                              |   |   |   |   |   |
| PC ← dst                     | IRR          |     | 30                   |                              |   |   |   |   |   |
| <b>JR</b> cc, dst            | RA           |     | ccB                  | -                            | - | - | - | - | - |
| if cc is true,               |              |     |                      |                              |   |   |   |   |   |
| PC ← PC + d                  |              |     | (cc = 0 to F)        |                              |   |   |   |   |   |
| <b>LD</b> dst, src           | r            | IM  | rC                   | -                            | - | - | - | - | - |
| dst ← src                    | r            | R   | r8                   |                              |   |   |   |   |   |
|                              | R            | r   | r9                   |                              |   |   |   |   |   |
|                              |              |     | (r = 0 to F)         |                              |   |   |   |   |   |
|                              | r            | IR  | C7                   |                              |   |   |   |   |   |
|                              | IR           | r   | D7                   |                              |   |   |   |   |   |
|                              | R            | R   | E4                   |                              |   |   |   |   |   |
|                              | R            | IR  | E5                   |                              |   |   |   |   |   |
|                              | R            | IM  | E6                   |                              |   |   |   |   |   |
|                              | IR           | IM  | D6                   |                              |   |   |   |   |   |
|                              | IR           | R   | F5                   |                              |   |   |   |   |   |
|                              | r            | x   | 87                   |                              |   |   |   |   |   |
|                              | x            | r   | 97                   |                              |   |   |   |   |   |

Table 20. Instruction Summary (Continued)

| Instruction<br>and Operation   | Address Mode |     | Opcode<br>Byte (Hex) | Flags Affected |   |   |   |   |   |
|--------------------------------|--------------|-----|----------------------|----------------|---|---|---|---|---|
|                                | dst          | src |                      | C              | Z | S | V | D | H |
| <b>LDB</b> dst, src            | r0           | Rb  | 47                   | -              | - | - | - | - | - |
| dst←src                        | Rb           | R0  | 47                   |                |   |   |   |   |   |
| <b>LDC/LDE</b>                 | r            | lrr | C3                   | -              | - | - | - | - | - |
| dst←src                        | lrr          | r   | D3                   |                |   |   |   |   |   |
|                                | r            | xs  | E7                   |                |   |   |   |   |   |
|                                | xs           | r   | F7                   |                |   |   |   |   |   |
|                                | r            | x1  | A7                   |                |   |   |   |   |   |
|                                | x1           | r   | B7                   |                |   |   |   |   |   |
|                                | r            | DA  | A7                   |                |   |   |   |   |   |
|                                | DA           | r   | B7                   |                |   |   |   |   |   |
| <b>LDCD/LDED</b>               | r            | lrr | E2                   | -              | - | - | - | - | - |
| dst, src                       |              |     |                      |                |   |   |   |   |   |
| dst←src                        |              |     |                      |                |   |   |   |   |   |
| rr←rr - 1                      |              |     |                      |                |   |   |   |   |   |
| <b>LDEI/LDCI</b> dst, src      | r            | lrr | E3                   | -              | - | - | - | - | - |
| dst←src                        |              |     |                      |                |   |   |   |   |   |
| rr←rr + 1                      |              |     |                      |                |   |   |   |   |   |
| <b>LDCPD/LDEPD</b>             |              |     |                      |                |   |   |   |   |   |
| dst, src                       |              |     |                      |                |   |   |   |   |   |
| rr←rr - 1                      | lrr          | r   | F2                   | -              | - | - | - | - | - |
| dst←src                        |              |     |                      |                |   |   |   |   |   |
| <b>LDCPI/LDEPI</b> dst,<br>src |              |     |                      |                |   |   |   |   |   |
| rr←rr + 1                      | lrr          | r   | F3                   | -              | - | - | - | - | - |
| dst←src                        |              |     |                      |                |   |   |   |   |   |
| <b>LDW</b> dst, src            | RR           | R   | C4                   | -              | - | - | - | - | - |
| dst←src                        | RR           | IR  | C5                   |                |   |   |   |   |   |
|                                | RR           | IM  | C6                   |                |   |   |   |   |   |
| <b>MULT</b> dst, src           | RR           | R   | 84                   | -              | - | - | - | - | - |

Table 20. Instruction Summary (Continued)

| Instruction<br>and Operation | Address Mode      |     | Opcode<br>Byte (Hex) | Flags Affected |   |   |   |   |   |
|------------------------------|-------------------|-----|----------------------|----------------|---|---|---|---|---|
|                              | dst               | src |                      | C              | Z | S | V | D | H |
| dst←src                      | RR                | IR  | 85                   |                |   |   |   |   |   |
|                              | RR                | IM  | 86                   |                |   |   |   |   |   |
| <b>NEXT</b>                  |                   |     | 0F                   | -              | - | - | - | - | - |
| PC←@IP                       |                   |     |                      |                |   |   |   |   |   |
| IP←IP + 2                    |                   |     |                      |                |   |   |   |   |   |
| <b>NOP</b>                   |                   |     | FF                   | -              | - | - | - | - | - |
| <b>OR</b> dst, src           | Note <sup>1</sup> |     | 4[ ]                 | -              | * | * | 0 | - | - |
| dst←dst OR src               |                   |     |                      |                |   |   |   |   |   |
| <b>POP</b> dst               |                   | R   | 50                   | -              | - | - | - | - | - |
| dst←@SP;                     |                   | IR  | 51                   |                |   |   |   |   |   |
| SP←SP + 1                    |                   |     |                      |                |   |   |   |   |   |
| <b>POPUD</b> dst, src        | R                 | IR  | 92                   | -              | - | - | - | - | - |
| dst←src                      |                   |     |                      |                |   |   |   |   |   |
| IR←IR - 1                    |                   |     |                      |                |   |   |   |   |   |
| <b>POPUI</b> dst, src        | R                 | IR  | 93                   | -              | - | - | - | - | - |
| dst←src                      |                   |     |                      |                |   |   |   |   |   |
| IR←IR + 1                    |                   |     |                      |                |   |   |   |   |   |
| <b>PUSH</b> src              |                   | R   | 70                   | -              | - | - | - | - | - |
| SP←SP - 1; @SP←src           |                   | IR  | 71                   |                |   |   |   |   |   |
| <b>PUSHUD</b> dst, src       | IR                | R   | 82                   | -              | - | - | - | - | - |
| IR←IR - 1                    |                   |     |                      |                |   |   |   |   |   |
| dst←src                      |                   |     |                      |                |   |   |   |   |   |
| <b>PUSHUI</b> dst, src       | IR                | R   | 83                   | -              | - | - | - | - | - |
| IR←IR + 1                    |                   |     |                      |                |   |   |   |   |   |
| dst←src                      |                   |     |                      |                |   |   |   |   |   |
| <b>RCF</b>                   |                   |     | CF                   | 0              | - | - | - | - | - |
| C←0                          |                   |     |                      |                |   |   |   |   |   |
| <b>RET</b>                   |                   |     | AF                   | -              | - | - | - | - | - |



Table 20. Instruction Summary (Continued)

| Instruction<br>and Operation | Address Mode      |     | Opcode<br>Byte (Hex) | Flags Affected |   |   |   |   |   |
|------------------------------|-------------------|-----|----------------------|----------------|---|---|---|---|---|
|                              | dst               | src |                      | C              | Z | S | V | D | H |
| PC←@SP; SP←SP + 2            |                   |     |                      |                |   |   |   |   |   |
| <b>RL</b> dst                | R                 |     | 90                   | *              | * | * | * | - | - |
| C←dst(7)                     | IR                |     | 91                   |                |   |   |   |   |   |
| dst(0)←dst(7)                |                   |     |                      |                |   |   |   |   |   |
| dst(N + 1)←dst(N)            |                   |     |                      |                |   |   |   |   |   |
| N = 0 to 6                   |                   |     |                      |                |   |   |   |   |   |
| <b>RLC</b> dst               | R                 |     | 10                   | *              | * | * | * | - | - |
| dst(0)←C                     | IR                |     | 11                   |                |   |   |   |   |   |
| C←dst(7)                     |                   |     |                      |                |   |   |   |   |   |
| dst(N + 1)←dst(N)            |                   |     |                      |                |   |   |   |   |   |
| N = 0 to 6                   |                   |     |                      |                |   |   |   |   |   |
| <b>RR</b> dst                | R                 |     | E0                   | *              | * | * | * | - | - |
| C←dst(0)                     | IR                |     | E1                   |                |   |   |   |   |   |
| dst(7)←dst(0)                |                   |     |                      |                |   |   |   |   |   |
| dst(N)←dst(N + 1 )           |                   |     |                      |                |   |   |   |   |   |
| N = 0 to 6                   |                   |     |                      |                |   |   |   |   |   |
| <b>RRC</b> dst               | R                 |     | C0                   | *              | * | * | * | - | - |
| C←dst(0)                     | IR                |     | C1                   |                |   |   |   |   |   |
| dst(7)←C                     |                   |     |                      |                |   |   |   |   |   |
| dst(N)←dst(N + 1 )           |                   |     |                      |                |   |   |   |   |   |
| N = 0 to 6                   |                   |     |                      |                |   |   |   |   |   |
| <b>SB0</b>                   |                   |     | 4F                   | -              | - | - | - | - | - |
| BANK←0                       |                   |     |                      |                |   |   |   |   |   |
| <b>SB1</b>                   |                   |     | 5F                   | -              | - | - | - | - | - |
| BANK←1                       |                   |     |                      |                |   |   |   |   |   |
| <b>SBC</b> dst, src          | Note <sup>1</sup> |     | 3[ ]                 | *              | * | * | * | 1 | * |
| dst←dst – src – C            |                   |     |                      |                |   |   |   |   |   |
| <b>SCF</b>                   |                   |     | DF                   | 1              | - | - | - | - | - |

Table 20. Instruction Summary (Continued)

| Instruction and Operation | Address Mode      |     | Opcode Byte (Hex) | Flags Affected |   |   |   |   |   |
|---------------------------|-------------------|-----|-------------------|----------------|---|---|---|---|---|
|                           | dst               | src |                   | C              | Z | S | V | D | H |
| C←1                       |                   |     |                   |                |   |   |   |   |   |
| SRA dst                   | R                 |     | D0                | *              | * | * | 0 | - | - |
| dst(7)←dst(7)             | IR                |     | D1                |                |   |   |   |   |   |
| C←dst(0)                  |                   |     |                   |                |   |   |   |   |   |
| dst(N)←dst(N + 1 )        |                   |     |                   |                |   |   |   |   |   |
| N = 0 to 6                |                   |     |                   |                |   |   |   |   |   |
| SRP src                   |                   | IM  | 31                | -              | - | - | - | - | - |
| RP0←IM                    |                   |     |                   |                |   |   |   |   |   |
| RP0←IM + 8                |                   |     |                   |                |   |   |   |   |   |
| SRP0                      |                   | IM  | 31                | -              | - | - | - | - | - |
| RP0←IM                    |                   |     |                   |                |   |   |   |   |   |
| SRP1                      |                   | IM  | 31                | -              | - | - | - | - | - |
| RP1←IM                    |                   |     |                   |                |   |   |   |   |   |
| SUB dst, src              | Note <sup>1</sup> |     | 2[ ]              | *              | * | * | * | 1 | * |
| dst←dst – src             |                   |     |                   |                |   |   |   |   |   |
| SWAP dst                  | R                 |     | F0                | -              | * | * | U | - | - |
| dst(0-3)↔dst(4-7)         | IR                |     | F1                |                |   |   |   |   |   |
| TCM dst, src              | Note <sup>1</sup> |     | 6[ ]              | -              | * | * | 0 | - | - |
| (NOT dst) AND src         |                   |     |                   |                |   |   |   |   |   |
| TM dst, src               | Note <sup>1</sup> |     | 7[ ]              | -              | * | * | 0 | - | - |
| dst AND src               |                   |     |                   |                |   |   |   |   |   |
| WFI                       |                   |     | 3F                | -              | - | - | - | - | - |
| XOR dst, src              | Note <sup>1</sup> |     | B[ ]              | -              | * | * | 0 | - | - |
| dst←dst XOR src           |                   |     |                   |                |   |   |   |   |   |

1. These instructions have an identical set of addressing modes, which are encoded for brevity. The first opcode nibble identifies the command, and is found in the table above. The second nibble, represented by a [ ], defines the addressing mode as shown in Table 6.

**Table 21. Second Nibble**

| Addr Mode |     | Lower<br>Opcode<br>Nibble <sup>1</sup> |
|-----------|-----|----------------------------------------|
| dst       | src |                                        |
| r         | r   | [2]                                    |
| r         | lr  | [3]                                    |
| R         | R   | [4]                                    |
| R         | IR  | [5]                                    |
| R         | IM  | [6]                                    |

- For example, to use an opcode represented as x[ ] with an "RR" addressing mode, use the opcode "x4."  
0= Cleared to Zero  
1= Set to One  
-= Unaffected  
\*= Set or reset, depending on result of operation.  
U= Undefined

## SUPER-8 OPCODE MAP

|                    |   | Lower Nibble (Hex)              |                                  |                                                      |                                                 |                                                 |                                                  |                                    |                                                              |                                           |                                           |                                     |                      |                               |                      |                            |            |              |  |  |  |
|--------------------|---|---------------------------------|----------------------------------|------------------------------------------------------|-------------------------------------------------|-------------------------------------------------|--------------------------------------------------|------------------------------------|--------------------------------------------------------------|-------------------------------------------|-------------------------------------------|-------------------------------------|----------------------|-------------------------------|----------------------|----------------------------|------------|--------------|--|--|--|
|                    |   | 0                               | 1                                | 2                                                    | 3                                               | 4                                               | 5                                                | 6                                  | 7                                                            | 8                                         | 9                                         | A                                   | B                    | C                             | D                    | E                          | F          |              |  |  |  |
| Upper Nibble (Hex) | 0 | 6<br>DEC<br>R <sub>1</sub>      | 6<br>DEC<br>IR <sub>1</sub>      | 6<br>ADD<br>r <sub>1</sub> ,f <sub>2</sub>           | 6<br>ADD<br>r <sub>1</sub> ,f <sub>2</sub>      | 10<br>ADD<br>R <sub>2</sub> ,R <sub>1</sub>     | 10<br>ADD<br>IR <sub>2</sub> ,R <sub>1</sub>     | 10<br>ADD<br>R <sub>1</sub> ,IM    | 10<br>BOR*<br>r <sub>0</sub> ,R <sub>b</sub>                 | 6<br>LD<br>r <sub>1</sub> ,R <sub>2</sub> | 6<br>LD<br>r <sub>2</sub> ,R <sub>1</sub> | 12/10<br>DJNZ<br>r <sub>1</sub> ,RA | 12/10<br>JR<br>cc,RA | 6<br>LD<br>r <sub>1</sub> ,RM | 12/10<br>JP<br>cc,DA | 6<br>INC<br>r <sub>1</sub> | 14<br>NEXT |              |  |  |  |
|                    | 1 | 6<br>RLC<br>R <sub>1</sub>      | 6<br>RLC<br>IR <sub>1</sub>      | 6<br>ADC<br>r <sub>1</sub> ,f <sub>2</sub>           | 6<br>ADC<br>r <sub>1</sub> ,f <sub>2</sub>      | 10<br>ADC<br>R <sub>2</sub> ,R <sub>1</sub>     | 10<br>ADC<br>IR <sub>2</sub> ,R <sub>1</sub>     | 10<br>ADC<br>R <sub>1</sub> ,IM    | 10<br>BCP<br>r <sub>1</sub> ,b,R <sub>2</sub>                |                                           |                                           |                                     |                      |                               |                      |                            |            | 20<br>ENTER  |  |  |  |
|                    | 2 | 6<br>INC<br>R <sub>1</sub>      | 6<br>INC<br>IR <sub>1</sub>      | 6<br>SUB<br>r <sub>1</sub> ,f <sub>2</sub>           | 6<br>SUB<br>r <sub>1</sub> ,f <sub>2</sub>      | 10<br>SUB<br>R <sub>2</sub> ,R <sub>1</sub>     | 10<br>SUB<br>IR <sub>2</sub> ,R <sub>1</sub>     | 10<br>SUB<br>R <sub>1</sub> ,IM    | 10<br>BXOR*<br>r <sub>0</sub> ,R <sub>b</sub>                |                                           |                                           |                                     |                      |                               |                      |                            |            | 22<br>EXIT   |  |  |  |
|                    | 3 | 10<br>JP<br>IRR <sub>1</sub>    | NOTE<br>C                        | 6<br>SBC<br>r <sub>1</sub> ,f <sub>2</sub>           | 6<br>SBC<br>r <sub>1</sub> ,f <sub>2</sub>      | 10<br>SBC<br>R <sub>2</sub> ,R <sub>1</sub>     | 10<br>SBC<br>IR <sub>2</sub> ,R <sub>1</sub>     | 10<br>SBC<br>R <sub>1</sub> ,IM    | NOTE<br>A                                                    |                                           |                                           |                                     |                      |                               |                      |                            |            | 6<br>WFI     |  |  |  |
|                    | 4 | 6<br>DA<br>R <sub>1</sub>       | 6<br>DA<br>IR <sub>1</sub>       | 6<br>OR<br>r <sub>1</sub> ,f <sub>2</sub>            | 6<br>OR<br>r <sub>1</sub> ,f <sub>2</sub>       | 10<br>OR<br>R <sub>2</sub> ,R <sub>1</sub>      | 10<br>OR<br>IR <sub>2</sub> ,R <sub>1</sub>      | 10<br>OR<br>R <sub>1</sub> ,IM     | 10<br>LDB*<br>r <sub>0</sub> ,R <sub>b</sub>                 |                                           |                                           |                                     |                      |                               |                      |                            |            | 6<br>SBO     |  |  |  |
|                    | 5 | 10<br>POP<br>R <sub>1</sub>     | 10<br>POP<br>IR <sub>1</sub>     | 6<br>AND<br>r <sub>1</sub> ,f <sub>2</sub>           | 6<br>AND<br>r <sub>1</sub> ,f <sub>2</sub>      | 10<br>AND<br>R <sub>2</sub> ,R <sub>1</sub>     | 10<br>AND<br>IR <sub>2</sub> ,R <sub>1</sub>     | 10<br>AND<br>R <sub>1</sub> ,IM    | 10<br>BITC<br>r <sub>1</sub> ,b                              |                                           |                                           |                                     |                      |                               |                      |                            |            | 6<br>SBI     |  |  |  |
|                    | 6 | 6<br>COM<br>R <sub>1</sub>      | 6<br>COM<br>IR <sub>1</sub>      | 6<br>TCM<br>r <sub>1</sub> ,f <sub>2</sub>           | 6<br>TCM<br>r <sub>1</sub> ,f <sub>2</sub>      | 10<br>TCM<br>R <sub>2</sub> ,R <sub>1</sub>     | 10<br>TCM<br>IR <sub>2</sub> ,R <sub>1</sub>     | 10<br>TCM<br>R <sub>1</sub> ,IM    | BAND*<br>r <sub>0</sub> ,R <sub>b</sub>                      |                                           |                                           |                                     |                      |                               |                      |                            |            |              |  |  |  |
|                    | 7 | 10/12<br>PUSH<br>R <sub>2</sub> | 12/14<br>PUSH<br>IR <sub>2</sub> | 6<br>TM<br>r <sub>1</sub> ,f <sub>2</sub>            | 6<br>TM<br>r <sub>1</sub> ,f <sub>2</sub>       | 10<br>TM<br>R <sub>2</sub> ,R <sub>1</sub>      | 10<br>TM<br>IR <sub>2</sub> ,R <sub>1</sub>      | 10<br>TM<br>R <sub>1</sub> ,IM     | NOTE<br>B                                                    |                                           |                                           |                                     |                      |                               |                      |                            |            |              |  |  |  |
|                    | 8 | 10<br>DECW<br>IRR <sub>1</sub>  | 10<br>DECW<br>IR <sub>1</sub>    | 10<br>PUSHUD<br>IR <sub>1</sub> ,R <sub>2</sub>      | 10<br>PUSHUI<br>IR <sub>2</sub> ,R <sub>1</sub> | 24<br>MULT<br>R <sub>2</sub> ,RR <sub>1</sub>   | 24<br>MULT<br>IR <sub>2</sub> ,RR <sub>1</sub>   | 24<br>MULT<br>IM,RR <sub>1</sub>   | 10<br>LD<br>r <sub>1</sub> ,x,f <sub>2</sub>                 |                                           |                                           |                                     |                      |                               |                      |                            |            | 6<br>DI      |  |  |  |
|                    | 9 | 6<br>RL<br>R <sub>1</sub>       | 6<br>RL<br>IR <sub>1</sub>       | 6<br>POPUD<br>IR <sub>2</sub> ,R <sub>1</sub>        | 6<br>POPUI<br>R <sub>2</sub> ,RR <sub>1</sub>   | 28/12<br>DIV<br>R <sub>2</sub> ,RR <sub>1</sub> | 28/12<br>DIV<br>IR <sub>2</sub> ,RR <sub>1</sub> | 28/12<br>DIV<br>IM,RR <sub>1</sub> | 10<br>LD<br>r <sub>2</sub> ,x,r <sub>1</sub>                 |                                           |                                           |                                     |                      |                               |                      |                            |            | 6<br>EI      |  |  |  |
|                    | A | 10<br>INCW<br>RR <sub>1</sub>   | 10<br>INCW<br>IR <sub>1</sub>    | 6<br>CP<br>r <sub>1</sub> ,f <sub>2</sub>            | 6<br>CP<br>r <sub>1</sub> ,f <sub>2</sub>       | 10<br>CP<br>R <sub>2</sub> ,R <sub>1</sub>      | 10<br>CP<br>IR <sub>2</sub> ,R <sub>1</sub>      | 10<br>CP<br>R <sub>1</sub> ,IM     | NOTE<br>D                                                    |                                           |                                           |                                     |                      |                               |                      |                            |            | 14<br>RET    |  |  |  |
|                    | B | 6<br>CLR<br>R <sub>1</sub>      | 6<br>CLR<br>IR <sub>1</sub>      | 6<br>XOR<br>r <sub>1</sub> ,f <sub>2</sub>           | 6<br>XOR<br>r <sub>1</sub> ,f <sub>2</sub>      | 10<br>XOR<br>R <sub>2</sub> ,R <sub>1</sub>     | 10<br>XOR<br>IR <sub>2</sub> ,R <sub>1</sub>     | 10<br>XOR<br>R <sub>1</sub> ,IM    | NOTE<br>E                                                    |                                           |                                           |                                     |                      |                               |                      |                            |            | 16/6<br>IRET |  |  |  |
|                    | C | 6<br>RRC<br>IRR <sub>1</sub>    | 6<br>RRC<br>IR <sub>1</sub>      | 16/18<br>CPIJE<br>r <sub>1</sub> ,f <sub>2</sub> ,RA | 12<br>LDC*<br>r <sub>1</sub> ,f <sub>2</sub>    | 10<br>LDW<br>RR <sub>2</sub> ,RR <sub>1</sub>   | 10<br>LDW<br>IR <sub>2</sub> ,RR <sub>1</sub>    | 12<br>LDW<br>RR <sub>1</sub> ,IML  | 6<br>LD<br>r <sub>1</sub> ,f <sub>2</sub>                    |                                           |                                           |                                     |                      |                               |                      |                            |            | 6<br>RCF     |  |  |  |
|                    | D | 6<br>SRA<br>R <sub>1</sub>      | 6<br>SRA<br>IR <sub>1</sub>      | 16/18<br>CPUNE<br>r <sub>1</sub> ,f <sub>2</sub> ,RA | 12<br>LDC*<br>r <sub>2</sub> ,f <sub>1</sub>    | 20<br>CALL<br>IA <sub>1</sub>                   |                                                  | 10<br>LD<br>IR <sub>1</sub> ,IM    | 6<br>LD<br>r <sub>1</sub> ,f <sub>2</sub>                    |                                           |                                           |                                     |                      |                               |                      |                            |            | 6<br>SCF     |  |  |  |
|                    | E | 6<br>RR<br>R <sub>1</sub>       | 6<br>RR<br>IR <sub>1</sub>       | 16<br>LDCD*<br>r <sub>1</sub> ,f <sub>2</sub>        | 16<br>LDCI*<br>r <sub>1</sub> ,f <sub>2</sub>   | 10<br>LD<br>R <sub>2</sub> ,R <sub>1</sub>      | 10<br>LD<br>IR <sub>2</sub> ,R <sub>1</sub>      | 10<br>LD<br>R <sub>1</sub> ,IM     | 18<br>LDC*<br>f <sub>1</sub> ,f <sub>2</sub> ,x <sub>s</sub> |                                           |                                           |                                     |                      |                               |                      |                            |            | 6<br>CCF     |  |  |  |
|                    | F | 8<br>SWAP<br>R <sub>1</sub>     | 8<br>SWAP<br>IR <sub>1</sub>     | 16<br>LDCPD*<br>r <sub>2</sub> ,f <sub>1</sub>       | 16<br>LDCPI*<br>r <sub>2</sub> ,f <sub>1</sub>  | 18<br>CALL<br>IRR <sub>1</sub>                  | 18<br>CALL<br>R <sub>2</sub> ,IR <sub>1</sub>    | 18<br>CALL<br>DA <sub>1</sub>      | 18<br>LDC*<br>f <sub>1</sub> ,f <sub>2</sub> ,x <sub>s</sub> |                                           |                                           |                                     |                      |                               |                      |                            |            | 6<br>NOP     |  |  |  |

Note A

|                                        |                                        |
|----------------------------------------|----------------------------------------|
| 16/18<br>BTJRF<br>r <sub>2</sub> ,b,RA | 16/18<br>BTJRT<br>r <sub>2</sub> ,b,RA |
|----------------------------------------|----------------------------------------|

Note B

|                                |                                |
|--------------------------------|--------------------------------|
| 8<br>BITR<br>r <sub>1</sub> ,b | 8<br>BITS<br>r <sub>1</sub> ,b |
|--------------------------------|--------------------------------|

Note C

|                |                 |                 |
|----------------|-----------------|-----------------|
| 6<br>SRP<br>IM | 6<br>SRP0<br>IM | 6<br>SRP1<br>IM |
|----------------|-----------------|-----------------|

Legend:

r = 4-bit address  
R = 8-bit address  
b = bit number  
R<sub>1</sub> or r<sub>1</sub> = dsts address  
R<sub>2</sub> or r<sub>2</sub> = src address

\*Examples:

BORr<sub>0</sub>,R<sub>2</sub>  
is BORr<sub>1</sub>,b,R<sub>2</sub>  
or BORr<sub>2</sub>,b,R<sub>1</sub>  
LDCr<sub>1</sub>,f<sub>2</sub>  
is LDCr<sub>1</sub>,f<sub>2</sub> = program  
or LDEr<sub>1</sub>,f<sub>2</sub> = data

Sequence:

Opcode, first, second, third operands

NOTE: The blank areas are not defined.

Figure 19.Opcode Map

## INSTRUCTIONS

Table 22. Super8 Instructions

| Mnemonic                       | Operands | Instruction                            |
|--------------------------------|----------|----------------------------------------|
| <b>Load Instructions</b>       |          |                                        |
| CLR                            | dst      | Clear                                  |
| LD                             | dst, src | Load                                   |
| LDB                            | dst, src | Load bit                               |
| LDC                            | dst, src | Load program memory                    |
| LDE                            | dst, src | Load data memory                       |
| LDCD                           | dst, src | Load program memory and decrement      |
| LDED                           | dst, src | Load data memory and decrement         |
| LDCI                           | dst, src | Load program memory and increment      |
| LDEI                           | dst, src | Load data memory and increment         |
| LDCPD                          | dst, src | Load program memory with pre-decrement |
| LDEPD                          | dst, src | Load data memory with pre-decrement    |
| LDCPI                          | dst, src | Load program memory with pre-increment |
| LDEPI                          | dst, src | Load data memory with pre-increment    |
| LDW                            | dst, src | Load word                              |
| POP                            | dst      | Pop stack                              |
| POPUD                          | dst, src | Pop user stack (decrement)             |
| POPUI                          | dst, src | Pop user stack (increment)             |
| PUSH                           | src      | Push stack                             |
| PUSHUD                         | dst, src | Push user stack (decrement)            |
| PUSHUI                         | dst, src | Push user stack (increment)            |
| <b>Arithmetic Instructions</b> |          |                                        |
| ADC                            | dst, src | Add with carry                         |
| ADD                            | dst, src | Add                                    |
| CP                             | dst, src | Compare                                |
| DA                             | dst      | Decimal adjust                         |
| DEC                            | dst      | Decrement                              |

Table 22. Super8 Instructions (Continued)

| Mnemonic                            | Operands | Instruction                              |
|-------------------------------------|----------|------------------------------------------|
| DECW                                | dst      | Decrement word                           |
| DIV                                 | dst, src | Divide                                   |
| INC                                 | dst      | Increment                                |
| INCW                                | dst      | Increment word                           |
| MULT                                | dst, src | Multiply                                 |
| SBC                                 | dst, src | Subtract with carry                      |
| SUB                                 | dst, src | Subtract                                 |
| <b>Logical Instructions</b>         |          |                                          |
| AND                                 | dst, src | Logical AND                              |
| COM                                 | dst      | Complement                               |
| OR                                  | dst, src | Logical OR                               |
| XOR                                 | dst, src | Logical exclusive                        |
| <b>Program Control Instructions</b> |          |                                          |
| BTJRT                               | dst, src | Bit test jump relative on True           |
| BTJRF                               | dst, src | Bit test jump relative on False          |
| CALL                                | dst      | Call procedure                           |
| CPIJE                               | dst, src | Compare, increment and jump on equal     |
| CPIJNE                              | dst,src  | Compare, increment and jump on non-equal |
| DJNZ                                | r, dst   | Decrement and jump on non-zero           |
| ENTER                               | Enter    |                                          |
| EXIT                                | Exit     |                                          |
| IRET                                |          | Return from interrupt                    |
| JP                                  | cc, dst  | Jump on condition code                   |
| JP                                  | dst      | Jump unconditional                       |
| JR                                  | cc, dst  | Jump relative on condition code          |
| JR                                  | dst      | Jump relative unconditional              |
| NEXT                                |          | Next                                     |
| RET                                 |          | Return                                   |
| WFI                                 |          | Wait for interrupt                       |

Table 22. Super8 Instructions (Continued)

| Mnemonic                             | Operands | Instruction                |
|--------------------------------------|----------|----------------------------|
| <b>Bit Manipulation Instructions</b> |          |                            |
| BAND                                 | dst,src  | Bit AND                    |
| BCP                                  | dst, src | Bit compare                |
| BITC                                 | dst      | Bit complement             |
| BITR                                 | dst      | Bit reset                  |
| BITS                                 | dst      | Bit set                    |
| BOR                                  | dst, src | Bit OR                     |
| BXOR                                 | dst, src | Bit exclusive OR           |
| TCM                                  | dst, src | Test complement under mask |
| TM                                   | dst, src | Test under mask            |
| <b>Rotate and Shift Instructions</b> |          |                            |
| RL                                   | dst      | Rotate left                |
| RLC                                  | dst      | Rotate left through carry  |
| RR                                   | dst      | Rotate right               |
| RRC                                  | dst      | Rotate right through carry |
| SRA                                  | dst      | Shift right arithmetic     |
| SWAP                                 | dst      | Swap nibbles               |
| <b>CPU Control Instructions</b>      |          |                            |
| CCF                                  |          | Complement carry flag      |
| DI                                   |          | Disable interrupts         |
| EI                                   |          | Enable interrupts          |
| NOP                                  |          | Do nothing                 |
| RCF                                  |          | Reset carry flag           |
| SB0                                  |          | Set bank 0                 |
| SB1                                  |          | Set bank 1                 |
| SCF                                  |          | Set carry flag             |
| SRP                                  | src      | Set register pointers      |
| SRP0                                 | src      | Set register pointer zero  |

Table 22. Super8 Instructions (Continued)

| Mnemonic | Operands | Instruction              |
|----------|----------|--------------------------|
| SRP1     | src      | Set register pointer one |

## INTERRUPTS

The Super8 interrupt structure contains 8 levels of interrupt, 16 vectors, and 27 sources.

Interrupt priority is assigned by level, controlled by the Interrupt Priority register (IPR). Each level is masked (or enabled) according to the bits in the Interrupt Mask register (IMR), and the entire interrupt structure can be disabled by clearing a bit in the System Mode register (R222).

The three major components of the interrupt structure are sources, vectors, and levels. These are shown in Figure 20 and discussed in the following paragraphs.

### Sources

A source is anything that generates an interrupt. This can be internal or external to the Super8 MCU. Internal sources are hardwired to a particular vector and level, while external sources can be assigned to various external events.

### Vectors

The 16 vectors are divided unequally among the eight levels. For example, vector 12 belongs to level 2, while level 3 contains vectors 0, 2, 4, and 6.

The vector number is used to generate the address of a particular interrupt servicing routine; therefore all interrupts using the same vector must use the same interrupt handling routine.

### Levels

Levels provide the top level of priority assignment. While the sources and vectors are hardwired within each level, the priorities of the levels can be changed by using the Interrupt Priority register (see Figure 15 for bit details).

If more than one interrupt source is active, the source from the highest priority level is serviced first. If both sources are from the same level, the source with the lowest vector has priority. For example, if the UART Receive Data bit and UART Parity Error bit are both active, the UART Parity Error bit is serviced first because it is vector 16, and UART receive data is vector 20.



The levels are shown in Figure 20.

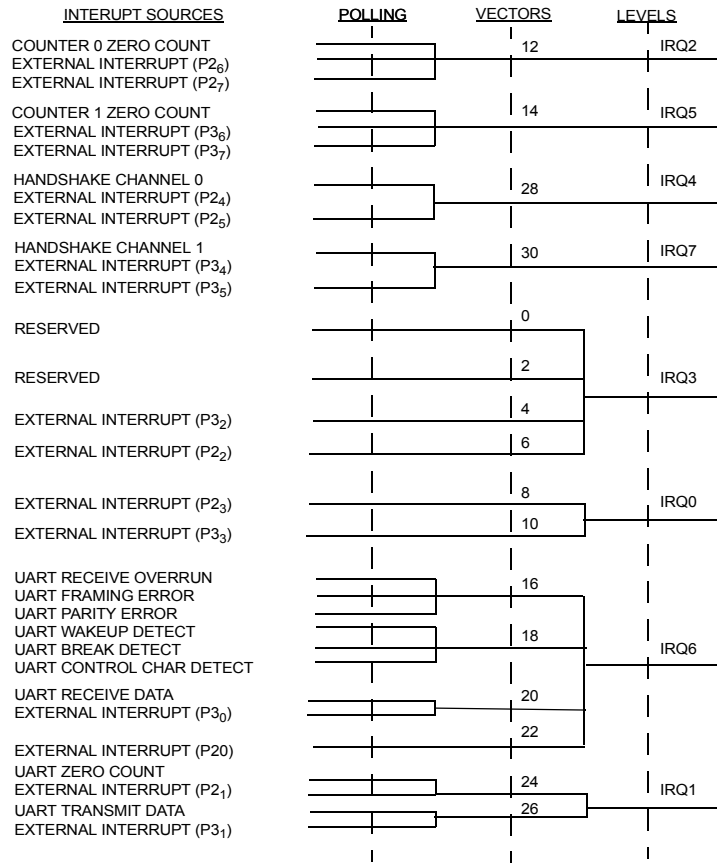


Figure 20. Interrupt Levels and Vectors

## Enables

- Interrupts can be enabled or disabled as follows:
- Interrupt enable/disable. The entire interrupt structure can be enabled or disabled by setting bit 0 in the System Mode register (R222).
- Level enable. Each level can be enabled or disabled by setting the appropriate bit in the Interrupt Mask register (R221).
- Level priority. The priority of each level can be controlled by the values in the Interrupt Priority register (R255, Bank 0).
- Source enable/disable. Each interrupt source can be enabled or disabled in the sources' Mode and Control register.

## Service Routines

Before an interrupt request can be granted, a) interrupts must be enabled, b) the level must be enabled, c) it must be the highest priority interrupting level, d) it must be enabled at the interrupting source, and e) it must have the highest priority within the level.

If all this occurs, an interrupt request is granted.

The Super8 then enters an interrupt machine cycle that completes the following sequence:

- It resets the Interrupt Enable bit to disable all subsequent interrupts.
- It saves the Program Counter and status flags on the stack.
- It branches to the address contained within the vector location for the interrupt.
- It passes control to the interrupt servicing routine.

When the interrupt servicing routine has serviced the interrupt, it should issue an interrupt return (IRET) instruction. This restores the Program Counter and status flags and sets the Interrupt Enable bit in the System Mode register.

## Fast Interrupt Processing

The Super8 provides a feature called fast interrupt processing, which completes the interrupt servicing in 6 clock periods instead of the usual 22.

Two hardware registers support fast interrupts. The Instruction Pointer (IP) holds the starting address of the service routine, and saves the PC value when a fast interrupt occurs. A dedicated register, FLAG', saves the contents of the FLAGS register when a fast interrupt occurs.

To use this feature, load the address of the service routine in the Instruction Pointer, load the level number into the Fast Interrupt Select field, and turn on the Fast Interrupt Enable bit in the System Mode register.

When an interrupt occurs in the level selected for fast interrupt processing, the following occurs:

- The contents of the Instruction Pointer and Program Counter are swapped.
- The contents of the Flag register are copied into FLAG'.
- The Fast Interrupt Status Bit in FLAGS is set.
- The interrupt is serviced.
- When IRET is issued after the interrupt service routine is completed, the Instruction Pointer and Program Counter are swapped again.

- The contents of FLAG' are copied back into the Flag register
- The Fast Interrupt Status bit in FLAGS is cleared.

The interrupt servicing routine selected for fast processing should be written so that the location after the IRET instruction is the entry point the next time the (same) routine is used.

## Level or Edge Triggered

Because internal interrupt requests are levels and interrupt requests from the outside are (usually) edges, the hardware for external interrupts uses edge-triggered flip-flops to convert the edges to levels.

The level-activated system requires that interrupt-serving software perform some action to remove the interrupting source. The action involved in serving the interrupt may remove the source, or the software may have to actually reset the flip-flops by writing to the corresponding Interrupt Pending register.

## STACK OPERATION

The Super8 architecture supports stack operations in the register file or in data memory. Bit 1 in the external Memory Timing register (R254 bank 0) selects between the two.

Register pair 216-217 forms the Stack Pointer used for all stack operations. R216 is the MSB and R217 is the LSB.

The Stack Pointer always points to data stored on the top of the stack. The address is decremented prior to a PUSH and incremented after a POP.

The stack is also used as a return stack for CALLs and interrupts. During a CALL, the contents of the PC are saved on the stack, to be restored later. Interrupts cause the contents of the PC and FLAGS to be saved on the stack, for recovery by IRET when the interrupt is finished.

When the Super8 is configured for an internal stack (using the register file), R217 contains the Stack Pointer. R216 may be used as a general-purpose register, but its contents are changed if an overflow or underflow, occurs as the result of incrementing or decrementing the stack address during normal stack operations.

## User-Defined Stacks

The Super8 provides for user-defined stacks in both the register file and program or data memory. These can be made to increment or decrement on a push by the choice of opcodes. For example, to implement a stack that grows from low

addresses to high addresses in the register file, use PUSHUI and POPUD. For a stack that grows from high addresses to low addresses in data memory, use LDEI for pop and LDEPD for push.

## COUNTER/TIMERS

The Super8 has two identical independently programmable 16-bit counter/timers that can be cascaded to produce a single 32-bit counter. They can be used to count external events, or they can obtain their input internally. The internal input is obtained by dividing the crystal frequency by four.

The counter/timers can be set to count up or down, by software or external events. They can be set for single or continuous cycle counting, and they can be set with a bi-value option, where two preset time constants alternate in loading the counter each time it reaches zero. This can be used to produce an output pulse train with a variable duty cycle.

The counter/timers can also be programmed to capture the count value at an external event or generate an interrupt whenever the count reaches zero. They can be turned on and off in response to external events by using a gate and/or a trigger option. The gate option enables counts only when the gate line is Low; the trigger option turns on the counter after a transient High. The gate and trigger options used together cause the counter/timer to work in gate mode after initially being triggered.

The control and status register bits for the counter/timers are shown in Figure 7.

## DMA

The Super8 features an on-chip Direct Memory Access (DMA) channel to provide high bandwidth data transmission capabilities. The DMA channel can be used by the UART receiver, UART transmitter, or handshake channel 0. Data can be transferred between the peripheral and contiguous locations in either the register file or external data memory. A 16-bit count register determines the number of transactions to be performed; an interrupt can be generated when the count is exhausted. DMA transfers to or from the register file require six CPU clock cycles; DMA transfers to or from external memory take ten CPU clock cycles, excluding wait states.

## ABSOLUTE MAXIMUM RATINGS

|                                            |                          |
|--------------------------------------------|--------------------------|
| Voltage on all pins with respect to ground | -0.3 V to +7.0 V         |
| Ambient Operating Temperature              | See Ordering Information |
| Storage Temperature                        | -65 °C to + 150 °C       |

Stresses greater than these may cause permanent damage to the device. This is a stress rating only; operation of the device under conditions more severe than those listed for operating conditions may cause permanent damage to the device. Exposure to absolute maximum ratings for extended periods may also cause permanent damage.

## STANDARD TEST CONDITIONS

Figure 21 shows the setup for standard test conditions. All voltages are referenced to ground, and positive current flows into the reference pin.

Standard conditions are:

- $+4.75\text{ V} \leq V_{CC} \leq +5.25\text{ V}$
- $GND = 0\text{ V}$
- $0^{\circ}\text{C} \leq T_A \leq +70^{\circ}\text{C}$

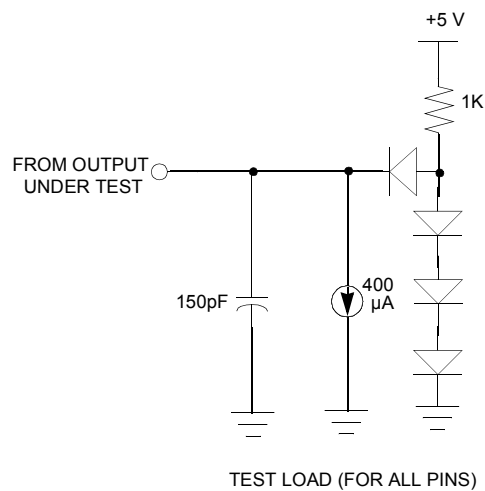


Figure 21. Standard Test Load

## DC CHARACTERISTICS

Table 23.DC Characteristics

| Symbol   | Parameter                | Min  | Max      | Unit    | Condition                          |
|----------|--------------------------|------|----------|---------|------------------------------------|
| $V_{CH}$ | Clock Input High Voltage | 3.8  | $V_{CC}$ | V       | Driven by External Clock Generator |
| $V_{CL}$ | Clock Input Low Voltage  | -0.3 | 0.8      | V       | Driven by External Clock Generator |
| $V_{IH}$ | Input High Voltage       | 2.2  | $V_{CC}$ | V       |                                    |
| $V_{IL}$ | Input Low Voltage        | -0.3 | 0.8      | V       |                                    |
| $V_{RH}$ | Reset Input High Voltage | 3.8  | $V_{CC}$ | V       |                                    |
| $V_{RL}$ | Reset Input Low Voltage  | -0.3 | 0.8      | V       |                                    |
| $V_{OH}$ | Output High Voltage      | 2.4  |          | V       | $I_{OH} = -400 \mu A$              |
| $V_{OL}$ | Output Low Voltage       |      | 0.4      | V       | $I_{OL} = +4.0 mA$                 |
| $I_{IL}$ | Input Leakage            | -10  | 10       | $\mu A$ |                                    |
| $I_{OL}$ | Output Leakage           | -10  | 10       | $\mu A$ |                                    |
| $I_{IR}$ | Reset Input Current      |      | -50      | $\mu A$ |                                    |
| $I_{CC}$ | VCC Supply Current       |      | 320      | mA      |                                    |

## INPUT HANDSHAKE TIMING

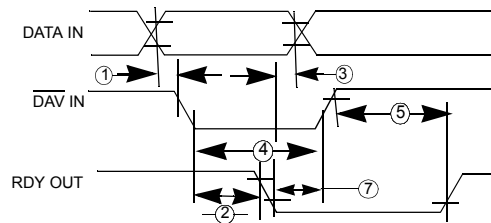


Figure 22.Fully Interlocked Mode

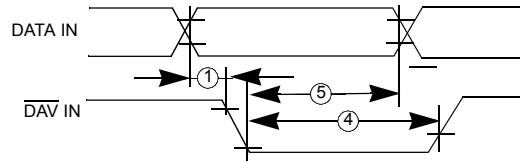


Figure 23.Strobed Mode

## AC CHARACTERISTICS (20 MHz)

### Input Handshake

Table 24.AC Characteristics (20 MHz) Input Handshake

| Number | Symbol       | Parameter                                                          | Min | Max | Notes <sup>1,2</sup> |
|--------|--------------|--------------------------------------------------------------------|-----|-----|----------------------|
| 1      | TsDI(DAV)    | Data In to Setup Time                                              | 0   |     |                      |
| 2      | TdDAVlf(RDY) | $\overline{\text{DAV}} \downarrow$ Input to RDY $\downarrow$ Delay |     | 200 | Note <sup>3</sup>    |
| 3      | ThDI(RDY)    | Data In Hold Time from RDY $\downarrow$                            | 0   |     |                      |
| 4      | TwDAV        | $\overline{\text{DAV}}$ In Width                                   | 45  |     |                      |
| 5      | ThDI(DAV)    | Data In Hold Time from $\overline{\text{DAV}} \downarrow$          | 130 |     |                      |
| 6      | TdDAV(RDY)   | $\overline{\text{DAV}} \uparrow$ Input to RDY $\uparrow$ Delay     |     | 100 | Note <sup>4</sup>    |
| 7      | TdRDYf(DAV)  | RDY $\downarrow$ Output to $\overline{\text{DAV}} \uparrow$ Delay  | 0   |     |                      |

1. Times are preliminary and subject to change.

2. Times given are in ns.

3. Standard Test Load

4. This time assumes user program reads data before  $\overline{\text{DAV}}$  Input goes high. RDY does not go high before data is read.

## OUTPUT HANDSHAKE TIMING

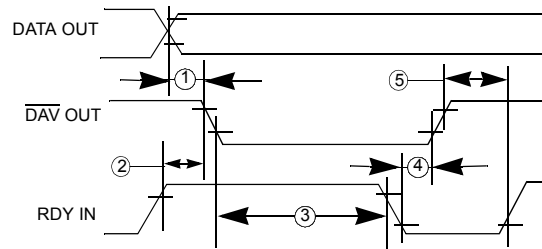


Figure 24.Fully Interlocked Mode

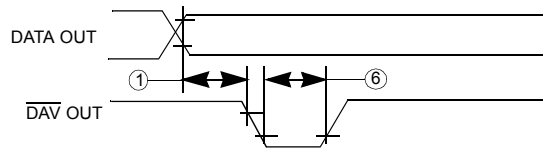


Figure 25.Strobed Mode

## AC CHARACTERISTICS (12 MHz, 20 MHz)

### Output Handshake

Table 25.AC Characteristics (12 MHz, 20 MHz) Output Handshake

| Number | Symbol                    | Parameter                                                             | Min | Max | Notes <sup>1,2</sup> |
|--------|---------------------------|-----------------------------------------------------------------------|-----|-----|----------------------|
| 1      | TdDO(DAV)                 | Data Out to $\overline{\text{DAV}}$ $\downarrow$ Delay                | 90  |     | Note <sup>3,4</sup>  |
| 2      | TdRDYr(DAV)               | RDY $\uparrow$ Input to $\overline{\text{DAV}}$ $\downarrow$ Delay    | 0   | 110 | Note <sup>3</sup>    |
| 3      | TdDAVO <sub>f</sub> (RDY) | $\overline{\text{DAV}}$ $\downarrow$ Output to RDY $\downarrow$ Delay | 0   |     |                      |
| 4      | TdRDY <sub>f</sub> (DAV)  | RDY $\downarrow$ Input to $\overline{\text{DAV}}$ $\uparrow$ Delay    | 0   | 110 | Note <sup>3</sup>    |
| 5      | TdDAVO <sub>r</sub> (RDY) | $\overline{\text{DAV}}$ $\uparrow$ Output to RDY $\uparrow$ Delay     | 0   |     |                      |
| 6      | TwDAVO                    | $\overline{\text{DAV}}$ Output Width                                  | 150 |     | Note <sup>4</sup>    |

1. Times are preliminary and subject to change.

2. Times given are in ns.

3. Standard Test Load

4. Time given is for zero value in Deskew Counter. For nonzero value of n where n = 1, 2, . . . 15 add 2 x n x TpC to the given time.



## AC CHARACTERISTICS (12 MHz)

### Read /Write

Table 26.AC Characteristics (12 MHz) Read/Write

| Number | Symbol      | Parameter                                                        | Normal Timing |     | Extended Timing |     | Notes <sup>1,2</sup> |
|--------|-------------|------------------------------------------------------------------|---------------|-----|-----------------|-----|----------------------|
|        |             |                                                                  | Min           | Max | Min             | Max |                      |
| 1      | TdA(AS)     | Address Valid to $\overline{AS}$ $\uparrow$ Delay                | 35            |     | 115             |     |                      |
| 2      | TdAS(A)     | $\overline{AS}$ $\uparrow$ to Address Float Delay                | 65            |     | 150             |     |                      |
| 3      | TdAS(DR)    | $\overline{AS}$ $\uparrow$ to Read Data Required Valid           |               | 270 |                 | 600 | Note <sup>3</sup>    |
| 4      | TWAS        | $\overline{AS}$ Low Width                                        | 65            |     | 150             |     |                      |
| 5      | TdA(DS)     | Address Float to $\overline{DS}$ $\downarrow$                    | 20            |     | 20              |     |                      |
| 6a     | TWDS(Read)  | $\overline{DS}$ (Read) Low Width                                 | 225           |     | 470             |     | Note <sup>3</sup>    |
| 6b     | TwDS(Write) | $\overline{DS}$ (Write) Low Width                                | 130           |     | 295             | 1   | Note <sup>3</sup>    |
| 7      | TdDS(DR)    | $\overline{DS}$ $\downarrow$ to Read Data Required Valid         |               | 180 |                 | 420 | Note <sup>3</sup>    |
| 8      | ThDS(DR)    | Read Data to $\overline{DS}$ $\uparrow$ Hold Time                | 0             |     | 0               |     |                      |
| 9      | TdDS(A)     | $\overline{DS}$ $\uparrow$ to Address Active Delay               | 50            |     | 135             |     |                      |
| 10     | TdDS(AS)    | $\overline{DS}$ $\uparrow$ to $\overline{AS}$ $\downarrow$ Delay | 60            |     | 145             |     |                      |
| 11     | TdDO(DS)    | Write Data Valid to $\overline{DS}$ (Write) $\downarrow$ Delay   | 35            |     | 115             |     |                      |
| 12     | TdAS(W)     | $\overline{AS}$ $\uparrow$ to Wait Delay                         |               | 220 |                 | 600 | Note <sup>4</sup>    |
| 13     | ThDS(W)     | $\overline{DS}$ $\uparrow$ to Wait Hold Time                     | 0             |     | 0               |     |                      |
| 14     | TdRW(AS)    | R/ $\overline{W}$ Valid to $\overline{AS}$ $\uparrow$ Delay      | 50            |     | 135             |     |                      |

1. All times are in ns and are for 12 MHz input frequency.

2. Timings are preliminary and subject to change

3.) Wait states add 167 ns to these times.

4.) Auto-wait states add 167 ns to this time..

## AC CHARACTERISTICS (20 MHz)

### Read /Write

Table 27.AC Characteristics (20 MHz) Read/Write

| Number | Symbol      | Parameter                                                      | Normal Timing |     | Extended Timing |     | Notes <sup>1,2</sup> |
|--------|-------------|----------------------------------------------------------------|---------------|-----|-----------------|-----|----------------------|
|        |             |                                                                | Min           | Max | Min             | Max |                      |
| 1      | TdA(AS)     | Address Valid to $\overline{AS}$ $\uparrow$ Delay              | 20            |     | 50              |     |                      |
| 2      | TdAS(A)     | $\overline{AS}$ $\uparrow$ to Address Float Delay              | 35            |     | 85              |     |                      |
| 3      | TdAS(DR)    | $\overline{AS}$ $\uparrow$ to Read Data Required Valid         |               | 150 |                 | 335 | Note <sup>3</sup>    |
| 4      | TWAS        | $\overline{AS}$ Low Width                                      | 35            |     | 85              |     |                      |
| 5      | TdA(DS)     | Address Float to $\overline{DS}$ $\downarrow$                  | 0             |     | 0               |     |                      |
| 6a     | TWDS(Read)  | $\overline{DS}$ (Read) Low Width                               | 125           |     | 275             |     | Note <sup>3</sup>    |
| 6b     | TWDS(Write) | $\overline{DS}$ (Write) Low Width                              | 65            |     | 165             |     | Note <sup>3</sup>    |
| 7      | TdDS(DR)    | $\overline{DS}$ $\downarrow$ to Read Data Required Valid       |               | 80  |                 | 225 | Note <sup>3</sup>    |
| 8      | ThDS(DR)    | Read Data to $\overline{DS}$ $\uparrow$ Hold Time              | 0             |     | 0               |     |                      |
| 9      | TdDS(A)     | $\overline{DS}$ $\uparrow$ to Address Active Delay             | 20            |     | 70              |     |                      |
| 10     | TdDS(AS)    | $\overline{DS}$ $\uparrow$ to $\overline{AS}$ 1 Delay          | 30            |     | 80              |     |                      |
| 11     | TdDO(DS)    | Write Data Valid to $\overline{DS}$ (Write) $\downarrow$ Delay | 10            |     | 50              |     |                      |
| 12     | TdAS(W)     | $\overline{AS}$ $\uparrow$ to Wait Delay                       |               | 90  |                 | 335 | Note <sup>4</sup>    |
| 13     | ThDS(W)     | $\overline{DS}$ $\uparrow$ to Wait Hold Time                   | 0             |     | 0               |     |                      |
| 14     | TdRW(AS)    | R/ $\overline{W}$ Valid to $\overline{AS}$ $\uparrow$ Delay    | 20            |     | 70              |     |                      |

1. All times are in ns and are for 20 MHz input frequency.

2. Timings are preliminary and subject to change.

3.) Wait states add 100 ns to these times.

4.) Auto-wait states add 100 ns to this time

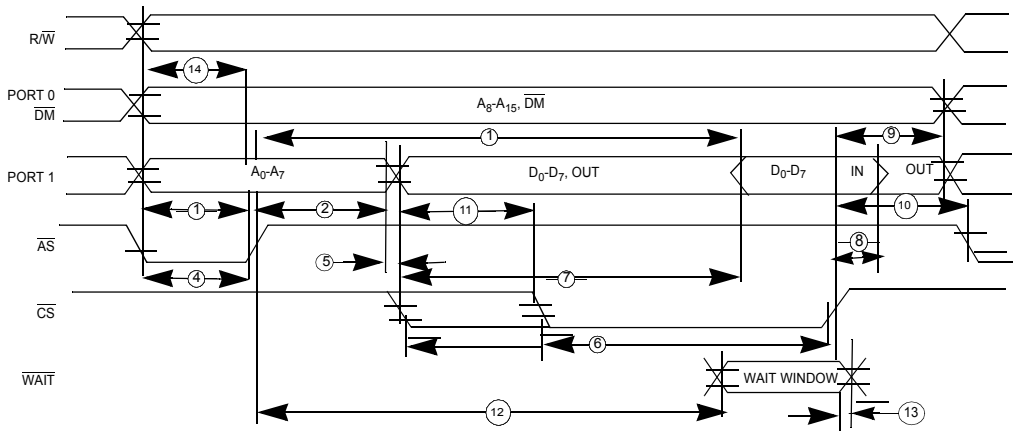


Figure 26.External Memory Read and Write Timing

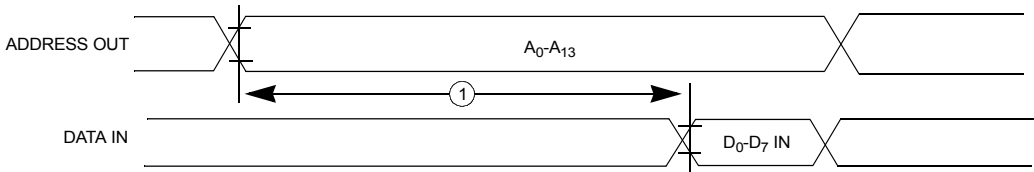


Figure 27.EPROM Read Timing

AC CHARACTERISTICS (20 MHz)

EPROM Read Cycle

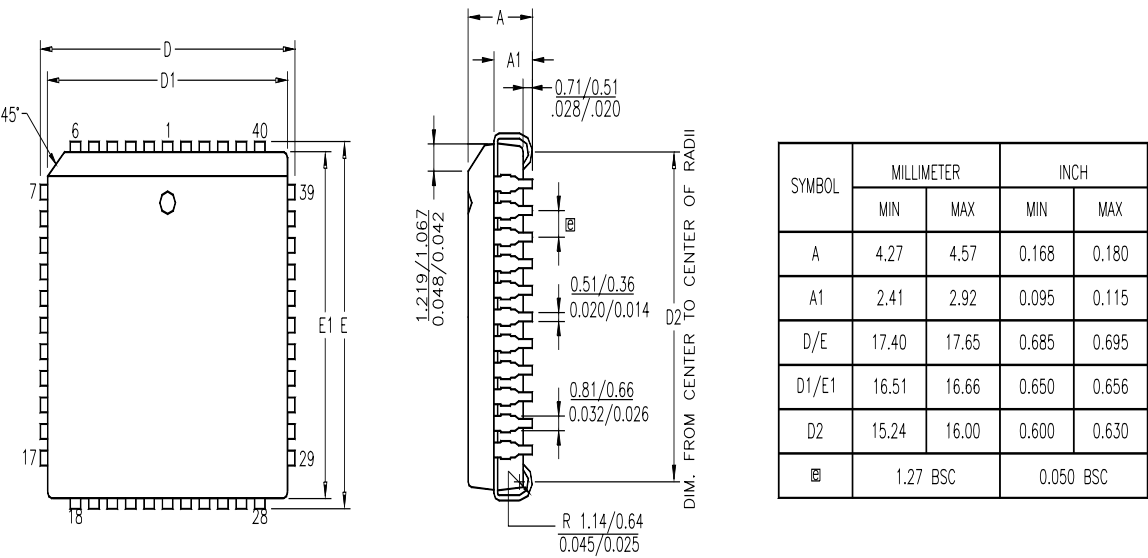
Example:

Table 28.AC Characteristics (20 MHz) EPROM Read Cycle

| Number | Symbol  | Parameter                                 | Min | Max | Notes <sup>1,2</sup> |
|--------|---------|-------------------------------------------|-----|-----|----------------------|
| 1      | TdA(DR) | Address Valid to Read Data Required Valid | 170 |     | Note <sup>3</sup>    |

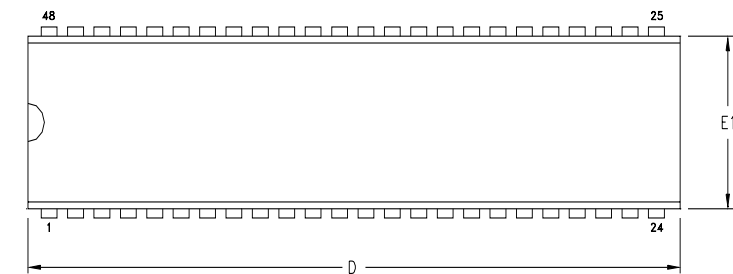
1. All times are in ns and are for 12 MHz input frequency.  
2. Timings are preliminary and subject to change.  
3.) Wait states add 167 ns to these times.

Packaging Information



- NOTES:  
1. CONTROLLING DIMENSION : INCH  
2. LEADS ARE COPLANAR WITHIN 0.004".  
3. DIMENSION :  $\frac{MM}{INCH}$

Figure 28.44-Pin PLCC



| SYMBOL | MILLIMETER |       | INCH     |       |
|--------|------------|-------|----------|-------|
|        | MIN        | MAX   | MIN      | MAX   |
| A1     | 0.38       | 0.81  | .015     | .032  |
| A2     | 3.68       | 4.19  | .145     | .165  |
| B      | 0.38       | 0.53  | .015     | .021  |
| B1     | 1.02       | 1.52  | .040     | .060  |
| C      | 0.23       | 0.38  | .009     | .015  |
| D      | 61.98      | 62.74 | 2.440    | 2.470 |
| E      | 15.24      | 15.75 | .600     | .620  |
| E1     | 13.72      | 14.22 | .540     | .560  |
| ⌀      | 2.54 BSC   |       | .100 BSC |       |
| eA     | 15.49      | 16.76 | .610     | .660  |
| L      | 3.18       | 3.81  | .125     | .150  |
| Q1     | 1.52       | 1.91  | .060     | .075  |
| S      | 1.52       | 2.29  | .060     | .090  |

CONTROLLING DIMENSIONS : INCH

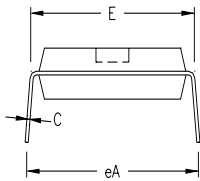
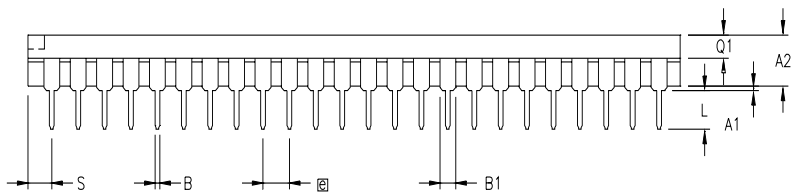


Figure 29.48-Pin DIP

# Mouser Electronics

Authorized Distributor

Click to View Pricing, Inventory, Delivery & Lifecycle Information:

[ZiLOG:](#)

[Z88C0020PSG](#) [Z88C0020VSG](#)